91 号 2015

# Xcell journal

SOLUTIONS FOR A PROGRAMMABLE WORLD

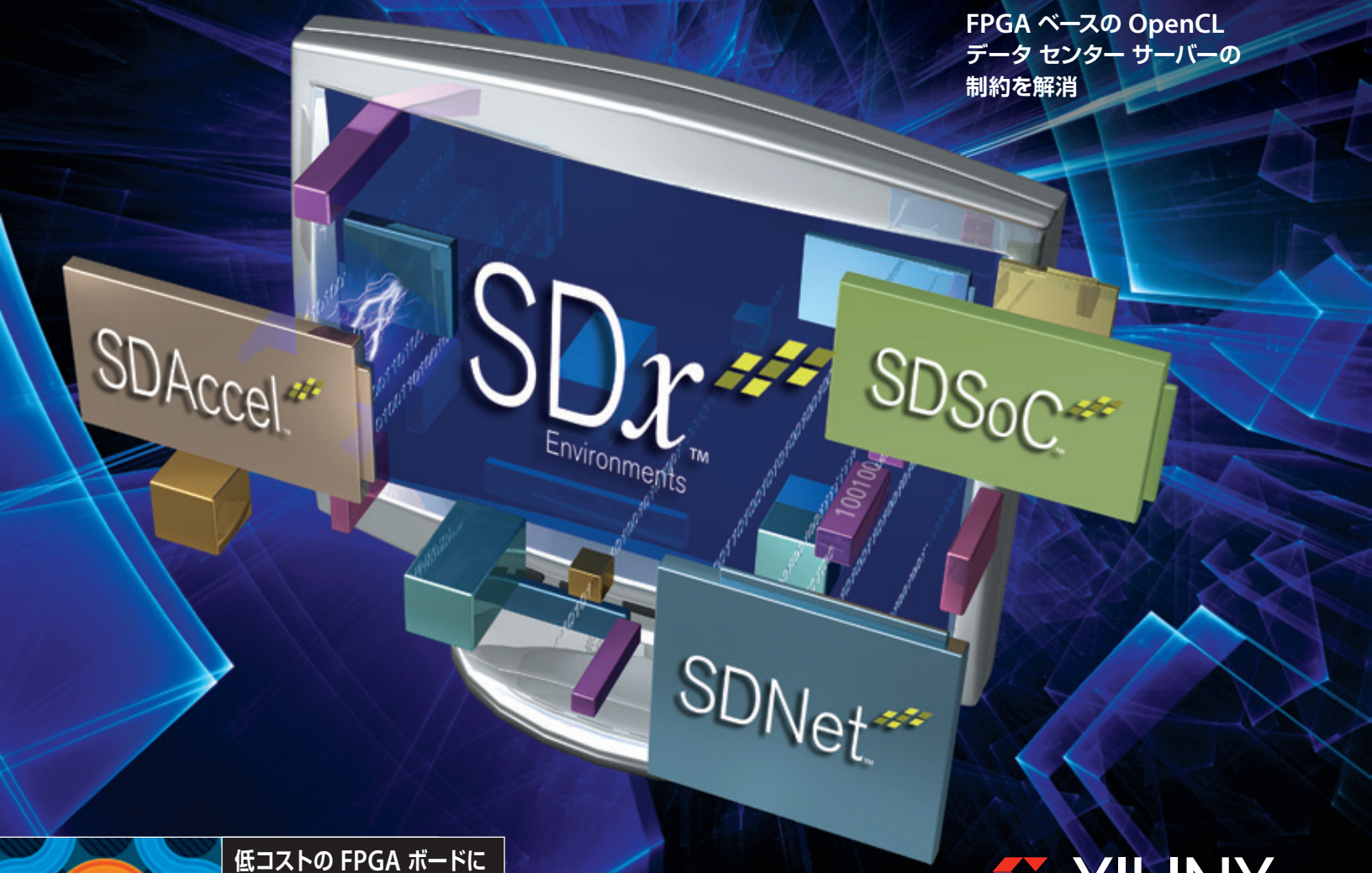
## All Programmable Abstraction: 慣れた言語でプログラマブル デバイスのプログラミングが容易に

FPGA のビデオ ウォーターマーキング用 OpenCL アプリケーションを最適化

周波数ドメインの基本を把握

プロ仕様ビデオ用の JPEG 2000 ネットワークを実現

FPGA ベースの OpenCL データ センター サーバーの制約を解消




低コストの FPGA ボードに インプリメントされる Oberon システム

ページ 28

XILINX
ALL PROGRAMMABLE™

japan.xilinx.com/xcell

Xcell journal

発行人 Mike Santarini
mike.santarini@xilinx.com
+1-408-626-5981

編集 Jacqueline Damian

アートディレクター Scott Blair

デザイン/制作 Teie, Gelwicks & Associates

日本語版統括 神保 直弘
naohiro.jinbo@xilinx.com

制作進行 周藤 智子
tomoko.suto@xilinx.com

日本語版 制作・広告 有限会社エイ・シー・シー

Xcell PUBLICATIONS

japan.xilinx.com/xcell/

Xcell Journal 日本語版 91 号

2015 年 6 月 30 日発行

Xilinx, Inc
2100 Logic Drive
San Jose, CA 95124-3400

**ザイリンクス株式会社**
〒 141-0032
東京都品川区大崎 1-2-2
アートヴィレッジ大崎セントラルタワー 4F

© 2015 Xilinx, Inc. All Right Reserved.

XILINX や、Xcell のロゴ、その他本書に記載の商標は、米国およびその他各国の Xilinx 社の登録商標です。ほかすべての名前は、各社の登録商標または商標です。

本書は、米国 Xilinx, Inc. が発行する英文季刊誌を、ザイリンクス株式会社が日本語に翻訳して発行したものです。

米国 Xilinx, Inc. およびザイリンクス株式会社は、本書に記載されたデータの使用に起因する第三者の特許権、他の権利、損害における一切の責任を負いません。

本書の一部または全部の無断転載、複写は、著作権法に基づき固く禁じます。

# 設計チームに生産性向上の基盤を提供

Xcell Journal 2015 年の 91 号へようこそ。本号の優れた記事の中でも、ザイリンクスが「All Programmable Abstraction」と呼んでいる戦略に関する 3 本の記事にご注目ください。All Programmable Abstraction という用語は、使い慣れたソフトウェア プログラミング モデルを FPGA デザインで使用可能にするために、ザイリンクスとアライアンス メンバー各社が提供する新たなタイプのハイレベル デザイン入力環境を指します。この開発環境では、設計チームの生産性が容易に向上し、ザイリンクスの All Programmable FPGA/SoC のプログラム経験のないユーザーでも、ハードウェア技術者の支援を受けることなくこれらのデバイスを利用できます。

カバー ストーリーでは、ハイレベル デザイン入力を向上させるザイリンクスの開発環境の進化について説明しています。All Programmable Abstraction 戦略は、2012 年にザイリンクスがリリースした Vivado 高位合成 (HLS) ツールと、MathWorks 社や National Instruments 社などのアライアンス メンバーによるサードパーティ製設計環境から始まりました。このイニシアチブは数年間で大きく成長し、先頃ザイリンクスは、新しい SDx 開発環境シリーズの 3 つの開発環境 (SDNet ™、SDAccel ™、SDSoC ™ ) を発表するにいたりました。

これらの新しい開発環境では、設計チーム全体の生産性が向上し、アーキテクチャ レベルで完成度の高いシステムを短期間で開発できます。これによって設計完了までの時間が短縮され、ソフトウェア開発を非常に有利にスタートできます。しかし、おそらくさらに素晴らしいことは、これらの環境では、新しいユーザーが利用できるシステムのリストにザイリンクスの FPGA と SoC が加わり、FPGA の専門家の支援を受けることなく革新的な新しいアプリケーションを開発できることです。最近の業界の試算によると、世界のソフトウェア技術者の数はハードウェア技術者の 10 倍にも上ります。したがって、SDx 環境では、ザイリンクスの All Programmable デバイスをベースにして革新的な製品を開発できるユーザーの幅が従来よりもはるかに広がります。

本号では、SDAccel 開発環境の 2 つの実践的な例も紹介しています。これらの記事は 36 ページと 40 ページに掲載されています。

また本号では、コンピューター エンジニアリング分野の生ける伝説となったニクラウス ヴィルト (Niklaus Wirth) 氏による高級言語の開発と Spartan®-3 FPGA プラットフォームへの移植について紹介しています。ご存じない方のために説明すると、ヴィルト教授は、プログラマをシステムの発明家に変えていく活動の中で Pascal 言語および複数の後継言語を開発した人物です。

ヴィルト教授は ( 現在は退職されていますが )、コンピューター サイエンスの次世代プロフェッショナルを生み出すようシステム プログラミング教育を支援するため、自著の『Project Oberon』を改訂しました。初版のレッスンでターゲットとしたプロセッサは、既に製造停止になっています。市販の製品には適切な代替品が見つからないため、ヴィルト教授は、低コストの ( したがって、学生向きの価格の ) Digilent 社製 Spartan-3 開発ボードを使用して独自のシステムを設計しました。

この記事でヴィルト教授は、新世代のイノベーターの意欲を引き出すことを願って、Spartan-3 ボードを使用して Oberon プログラミング言語を改良した設計体験について説明しています。

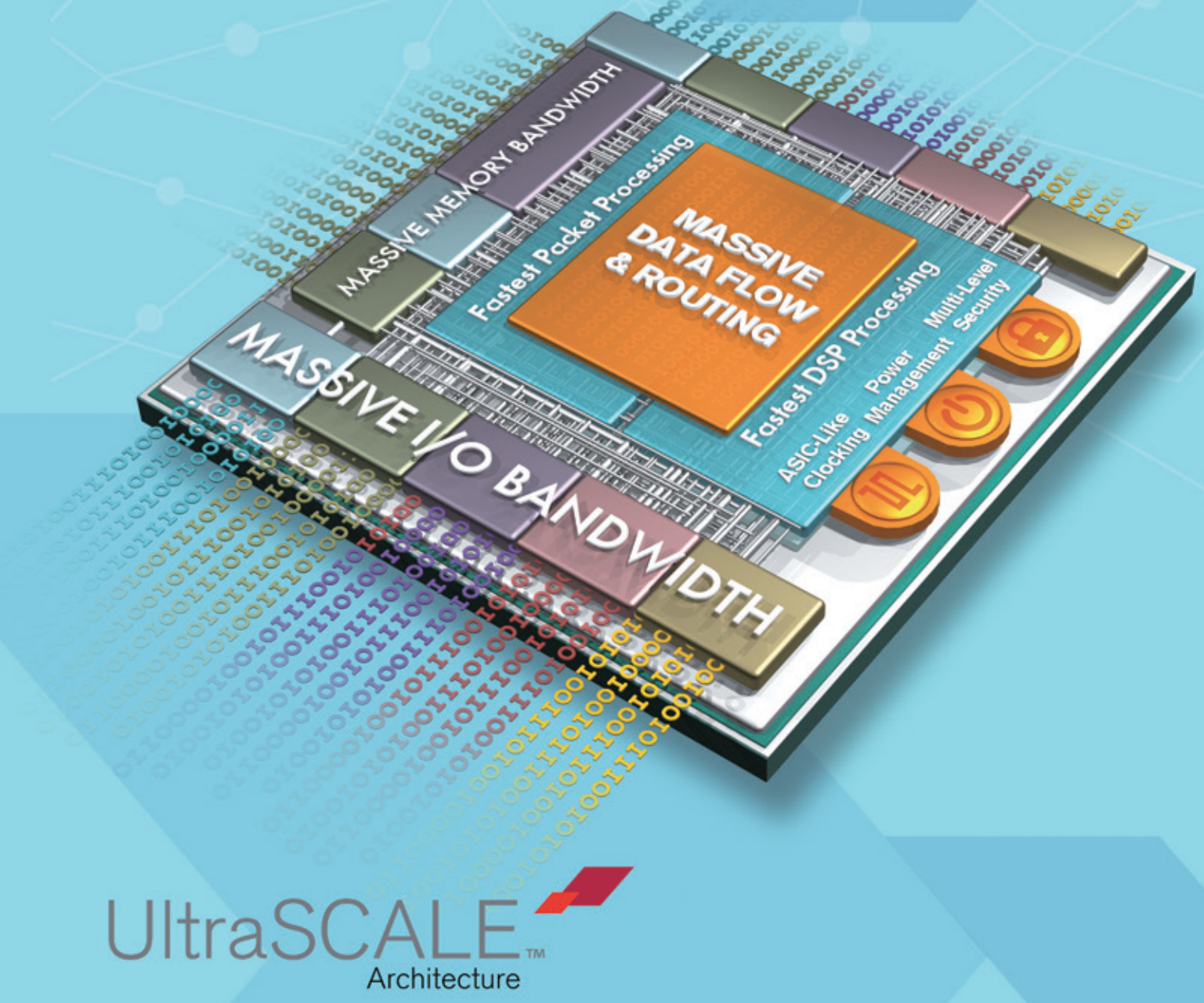
最後に、52 ページの優れたハウツー記事では、ザイリンクスの Daniel Michek が、Vivado ™ System Generator for DSP ツールを使用して最適化されたハードウェア プラットフォームを開発する方法について説明しています。

本号をお楽しみいただけたら幸いです。

Mike Santarini
発行人

# A Generation Ahead

## 業界初、ASIC クラスのプログラマブル アーキテクチャ




UltraSCALE™ Architecture

詳細はこちら

XILINX®
ALL PROGRAMMABLE

**ザイリンクス株式会社**
製品のお問い合わせは下記の販売代理店へどうぞ
■アヴネット(株) TEL (03) 5792-8210 EVAL-KITS-JP@avnet.co.jp ■(株)PALTEK TEL (045) 477-2001 info_pal@paltek.co.jp
©Copyright 2015 Xilinx, Inc. All rights reserved. ザイリンクスの名称およびロゴは、米国およびその他各国のザイリンクス社の登録商標および商標です。
ARMは、EUおよびその他の国におけるARM Limitedの登録商標です。他のすべての商標はそれぞれの所有者の財産です。

## VIEWPOINTS




## XCELLENCE BY DESIGN APPLICATION FEATURES






## Cover Story 6

All Programmable Abstraction:
慣れた言語で
プログラマブル デバイスの
プログラミングが容易に

## XCELLENCE BY DESIGN APPLICATION FEATURES





## THE XILINX XPERIENCE FEATURES



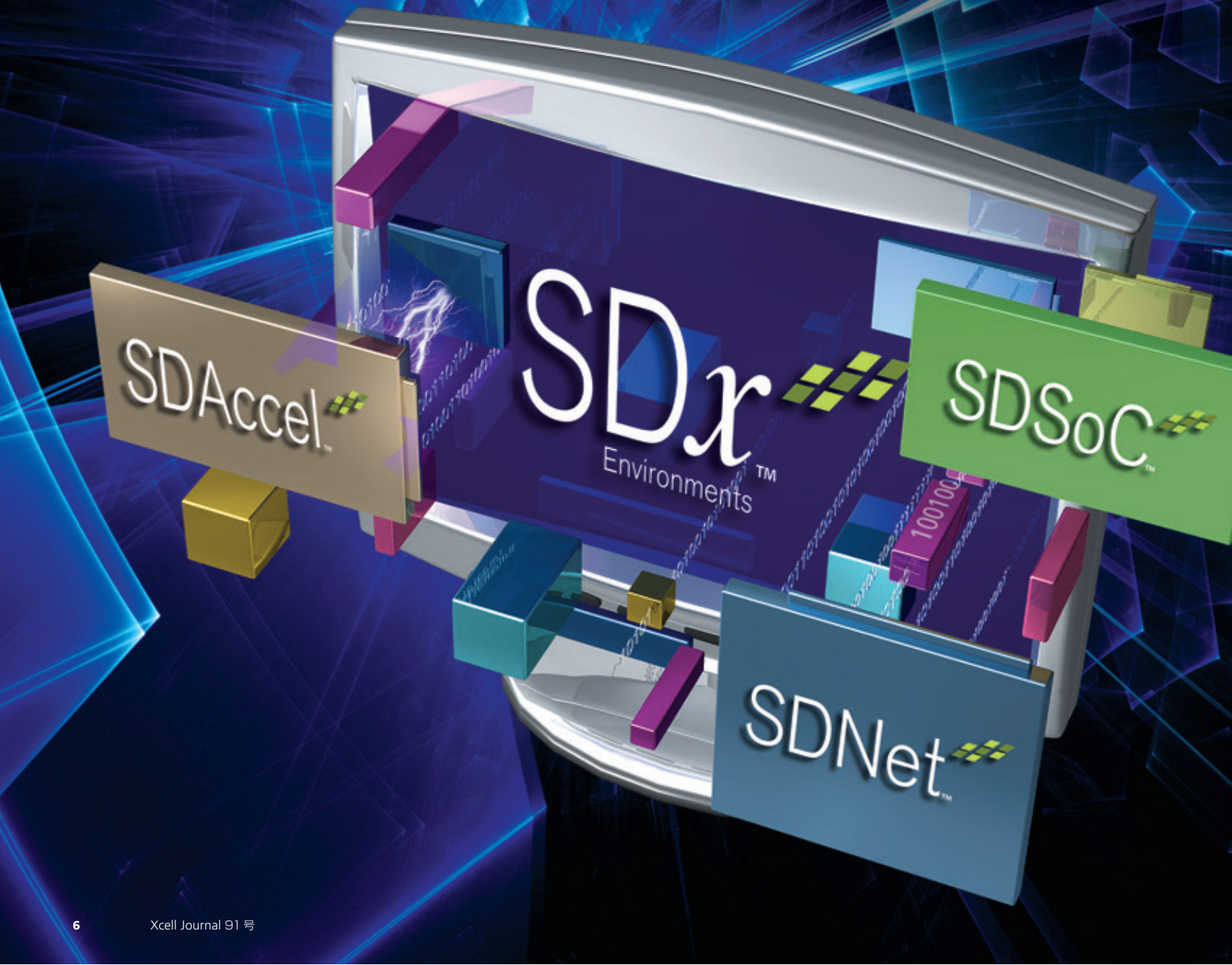
APEX® AWARDS FOR PUBLICATION EXCELLENCE
Excellence in Magazine & Journal Writing
2010, 201

APEX® AWARDS FOR PUBLICATION EXCELLENCE
Excellence in Magazine & Journal Design
2010, 2011, 2012

All Programmable Abstractions: Programming Your Way

# All Programmable Abstraction: 慣れた言語でプログラマブル デバイスのプログラミングが容易に

ザイリンクスの新しい SDx ソフトウェア定義環境は、Vivado IPI、HLS、および一般的なシステム レベル設計ツールを補完します。



**Mike Santarini**
Publisher, Xcell Journal
Xilinx, Inc.
*mike.santarini@xilinx.com*

先頃ザイリンクスは、設計チームの生産性をこれまでにないレベルに高め、All Programmable FPGA、SoC、3D IC のユーザー基盤を多くのソフトウェア技術者へと拡大する、SDx™ 開発環境ファミリの 2 つの新製品を公開しました。新しい SDAccel™ 開発環境では、FPGA 設計経験のないデータ センター機器のプログラマーが、OpenCL™、C、または C++ を使用して、データ センターおよびクラウド コンピューティングのインフラストラクチャ機器用にザイリンクスの FPGA をプログラムできます。この方法で開発された FPGA ベースの機器は、GPU/CPU ベースの機器よりもはるかに優れた単位ワット当たりの性能 ( パフォーマンス / ワット ) を発揮します。また今回、ザイリンクスは SDSoC™ 開発環境を公開しました。SDSoC™ 開発環境では、FPGA 設計未経験のソフトウェア開発者が、C または C++ を使用して、ザイリンクスおよびサードパーティ プラットフォーム開発者が提供する Zynq®-7000 All Programmable SoC および UltraScale+™ MPSoC プラットフォームをターゲットとするシステムを開発できます。

SDx 環境は、ザイリンクスの All Programmable Abstraction イニシアチブに基づく最新の成果です。このイニシアチブの目的は、ソフトウェア技術者およびシステム設計者が、各自の設計要件に合わせた開発環境で、ザイリンクスのデバイスを簡単にプログラムできるようにすることです。

ザイリンクスの企業戦略およびマーケティング グループのシニア バイス プレジデントであるスティーブ グレイザー (Steve Glaser) は次のように述べています。「SDNet、SDAccel、SDSoC の組み合わせは、システム技術者およびソフトウェア技術者に、使い慣れた CPU/GPU/ASSP 型のプログラミング環境を提供します。これらの技術者は、今回初めて、カスタマイズされたアクセラレーション、10 ～ 100 倍の単位ワット当たり性能、次世代スマート システムに必要なセキュリティと安全性を備えた「any-to-any」コネクティビティなど、All Programmable デバイス独自の利点を活用できるようになります。ザイリンクスは、これらの次世代システムのネットワーク接続機能、ソフトウェア定義機能、仮想化機能の強化を実現します。またこれらのシステムは、ソフトウェア ベースの分析と、広く普及したビデオ / エンベデッド ビジョンによって促進されるクラウド内での大量のコンピューティングをサポートする必要があります。そのためには、SDx ソフトウェア定義プログラミング環境と、新しい UltraScale FPGA および MPSoC を使用したヘテロジニアス マルチプロセッシングが必要になります。」

新しい SDAccel および SDSoC 環境は、2014 年春の SDNet™ 開発環境のリリース (Xcell Journal 日本語版 87 号のカバーストーリーを参照 ) に続いて提供されます。

新しい SDx 環境では、ソフトウェア技術者およびシステム設計者がザイリンクス デバイスの FPGA 部分をプログラムできるようになりますが、ハードウェア技術者を擁する設計チームにとっても、生産性の向上と最適化されたシステムへの迅速な設計収束が可能となり、time-to-market を短縮する効果をもたらします。実際に機能するシステム デザインを手に入れることにより、ハードウェア技術者は FPGA のレイアウトとパフォーマンスを最適化してシステムの効率を上げることに集中でき、他方ソフトウェア技術者はアプリケーション コードをさらに改良することができます。

## All Programmable Abstraction

ザイリンクスが 2008 年に新任 CEO のモーシェ ガブリエロフ (Moshe Gavrielov) の下で FPGA、3D IC、Zynq-7000 All Programmable SoC からなる 7 シリーズ All Programmable デバイス ファミリの計画に着手したときから、7 シリーズおよび将来の製品ラインの各デバイスの豊富な機能によって、顧客が最新で革新的製品の心臓部にザイリンクスのデバイスを利用できることは明らかでした。

これらの All Programmable デバイスは、ザイリンクスの初期のグルー ロジック FPGA よりもはるかに高度な機能を持ち、他のアーキテクチャでは達成できないシステム機能と最終製品の差別化を実現します。ザイリンクスの経営陣は、これらの新しいデバイスの価値を最大限に引き出して競合他社の優位に立つには、FPGA の専門家だけでなく、システム設計者やエンベデッド ソフトウェア開発者もザイリンクスの最新デバイス

をプログラミングできるようにするツールと手法を開発する必要性を理解していました。さらにザイリンクスは、主要な成長市場をターゲットとするソフトウェア技術者向けの設計環境を開発し、設計者が使い慣れたツールとフローに合わせてその環境を調整する必要がありました。また、MathWorks 社や National Instruments 社のような企業との連携を強化する必要もありました。これらの企業は、従来の FPGA ユーザー以外のユーザーがザイリンクス All Programmable デバイスの高い電力効率、性能、柔軟性を活用できる環境を既に確立していたからです。

すでにザイリンクスで実績を確立している顧客は、設計チームのすべてのメンバーに設計環境を提供することで、高い設計効率と time-to-market の短縮が保証されます。設計環境が十分に高度な機能を備えていれば、All Programmable FPGA および Zynq SoC のデザインは基本的に「民主化」され、FPGA を使った設計経験のない個々の設計者およびソフトウェア技術者でも、ハードウェア設計者の支援を受けずにこれらのデバイスをプログラムできるようになります。世界のソフトウェア技術者の数は、ハードウェア技術者の 10 倍に上ります。したがって、ザイリンクスと、このような開発環境 ( またはそれらの環境をサポートするハードウェア プラットフォーム ) を提供しているアライアンス プログラム パートナーは、ユーザー数と収益の両方を拡大できるでしょう。

こうした手法とそれに続く開発作業により、ソフトウェア技術者およびシステム設計者は、各自の設計要件に合わせた環境でザイリンクスのデバイスをプログラムできるようになります。ザイリンクスはこのような戦略を All Programmable Abstraction と呼んでいます ( 図 1)。

## Vivado HLS および IPI : デザイン アブストラクションの最初のステップ

デザイン アブストラクションの最初の重要なステップは、2011 年にザイリンクスが株式非公開の高位合成 (HLS) ツール ベンダーである AutoESL 社を買収したときに始まりました。この合併に続き、2012 年にザイリンクスは ISE® Design Suite および Vivado® Design Suite ツール フローに統合された HLS テクノロジを公開しました。ザイリンクスが AutoESL のテクノロジを選んだのは、Berkeley Design Automation 社による徹底的な調査によって、AutoESL の HLS ツールは EDA 業界で利用可能なすべての HLS ツールの中で最も使いやすく、最も高品質の結果が得られることが実証されたからです。このツールの起源が EDA 業界にあることは、このツールの利用モデルが、C および C++ のプログラミング経験とハードウェア記述言語 (Verilog および VHDL) およびチップ デザイン要件の実践的な知識を備えた、ASSP およびシステム オン チップ (SoC) システムの設計者と経験豊富な設計チームをターゲットとしていることを意味しています。

Vivado HLS では、こうした幅広いスキルを持つ設計者および設計チームは、C および C++ でアルゴリズムを作成する一方、作成したアルゴリズムを Vivado HLS を使ってコンパイルし、RTL IP ブロックに変換できます。次に FPGA 設計者は、その IP ( 知的設計資産 ) ブロックおよび ( 設計チームが作成またはライセンスを取得した ) 他の RTL ブロックを取り上げ、ザイリンクスの IP インテグレーター (IPI) ツールを使って IP からデザインをアセンブル ( 構築 ) します。次に、FPGA 設計者は、構築されたデザインを Vivado フローの中で取り上げ、HDL シミュレーション、タイミングおよび電力の最適化、配置配線を実行します。最終的に、FPGA 設計者はネットリスト / ビット ファイルを生成し、ターゲットとなる All Programmable デバイスのハードウェアをコンフィギュレーションします。デザインがプロセッサ (Zynq SoC またはソフト MPU コア ) を含んでいる場合、コンフィギュレーション済みのデバイスは、エンベデッド ソフトウェア技術者がプログラムできる状態になります。

エンベデッド ソフトウェア技術者の煩瑣なプログラミング作業を支援するため、ザイリンクスは、ザイリンクス ソフトウェア開発キット (SDK) と呼ばれる Eclipse ベースの統合設計環境を用意しています。この SDK は、ザイリンクスの 32 ビット MicroBlaze™ ソフト コアが組み込まれた Zynq SoC または FPGA をターゲットとするエディター、コンパイラ、デバッガー、ドライバー、およびライブラリで構成されます。10 年以上前に発表されたこの SDK は、ザイリンクスがデバイスに統合するプロセッサの変遷 ( ハード化された DSP スライスと 8/16/32 ビットのソフト コア MCU/MPU から、Virtex®-4 および Virtex®-5 FPGA のハード化された 32 ビット PowerPC®、Zynq SoC のハード化された 32 ビット ARM® プロセッサを経て、予定されている Zynq UltraScale+ MPSoC の 64 ビット

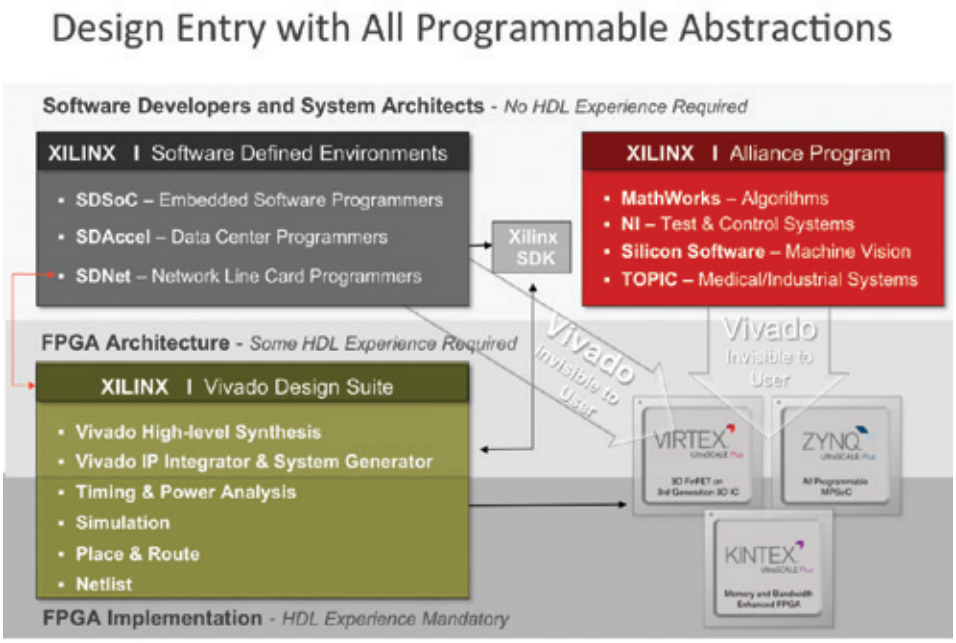



図 1 – ザイリンクスの新しい SDx 設計環境とアライアンス メンバーが提供する設計環境は、より多くの革新的な開発者がザイリンクスの All Programmable デバイスを利用して次世代製品の価値を高められるように支援します。

ARM プロセッサへ）とともに、飛躍的に進化してきました。

## アライアンス メンバーがザイリンクス デバイスの価値をより多くのユーザーに拡大

10 年以上にわたって、ザイリンクスは National Instruments 社および Math-Works 社との緊密な協力関係を維持してきました。これらの企業はいずれも、特定の対象ユーザーに合わせた独自のハイレベル開発環境を提供しています。

National Instruments 社（テキサス州オースティン）のハードウェア開発プラットフォームは、革新的な制御 / 試験システム開発者から熱心な支持を得ています。NI RIO プラットフォームには、ザイリンクスの FPGA と Zynq SoC が採用されています。National Instruments 社 の LabVIEW 開発環境は、Vivado Design Suite を内部で実行するユーザー フレンドリなグラフィックス ベースのプログラムであるため、National Instruments 社の顧客は FPGA デザインの詳しい知識を必要としません。RIO 製品の中核にザイリンクスのデバイスが搭載されていることすら知らないユーザーもいるほどです。ユーザーは単に LabVIEW 環境でシステムをプログラムし、NI 社のハードウェアによって開発中のデザインのパフォーマンスを高速化できます。

MathWorks 社（マサチューセッツ州ナティック）の場合、10 年以上前に同社の MATLAB®、Simulink®、HDL Coder、および Embedded Coder に FPGA のサポートが追加され、各製品の内部ではザイリンクスの ISE および Vivado ツールが完全に自動化されて実行されるようになりました。その結果、ユーザー（主に数学的アルゴリズムの開発者）は、FPGA ファブリック上で簡単にアルゴリズムを実行し、アルゴリズムの開発とパフォーマンスの飛躍的な高速化を実現できます。

ザイリンクスは、FPGA アーキテクチャレベルのツールである System Generator を 10 年以上前に ISE 開発環境に追加し、先頃 Vivado Design Suite にも追加しました。その目的は、FPGA の知識のある設計チームが、アルゴリズムの性能向上のためにさらにデザインを微調整できるようにすることです。こうした MathWorks 社とザイリンクスの技術の組み合わせにより、顧客企業は数千種類もの革新的な製品を開発してきました。

最近では、これ以外の企業もザイリンクスの環境に製品を提供し始めています。先頃ザイリンクスは、医療および産業市場に焦点を合わせたハイレベル開発環境を提供している TOPIC Embedded Systems 社および Silicon Software 社の 2 社の欧州企業を、新たにアライアンス プログラムに迎えました。

TOPIC Embedded Systems 社（アイントホーフェン、オランダ）は、DYPLO と呼ばれる独自の開発環境を提供しています（Xcell Journal 英語版 89 号を参照）。Zynq SoC と Zynq MPSoC（予定）をターゲットとするこの環境は、システムレベルのデザインに独自の手法を採用し、システム設計者がデザインのシステム表現を C または C++ で作成し、Zynq SoC のデュアルコア ARM Cortex™ -A9 プロセッシング システム上で実行できるようにします。ソフトウェアでの実行速度が遅すぎるデザインのセクションが見つかった場合、そのセクションを（Vivado HLS を内部で実行している）ウィンドウにドラッグ アンド ドロップすると、C コードが FPGA ロジックに変換され、Zynq SoC のプログラマブル ロジック内に配置されます。最適なシステム パフォーマンスが得られるまで、プロセッシング システムとプログラマブル ロジックの間でコード スニペットの移動を繰り返します。TOPIC 社の環境は、まず医療機器の開発に採用され、現在は産業用機器メーカーにも導入され始めています。

Silicon Software 社（マンハイム、ドイツ）は、ソフトウェア技術者によるザイリンクス All Programmable デバイスの活用を実現するアライアンス メンバー 4 社のうち最も新しく参加した企業です。同社の VisualApplets グラフィカル画像処理設計環境では、Zynq SoC プラットフォームをターゲットとするシステム設計者およびソフトウェア技術者が、ハードウェア技術者の支援を受けることなく、産業用マシン ビジョン アプリケーションの革新的な新製品を開発できます。2013 年の SPS Drives トレードショーのザイリンクス ブースで、Silicon Software 社は、VisualApplets 環境で開発した光学式検査システムのデモを行いました。このデモは、VisualApplets 環境を使って Zynq SoC のプロセッシング システムからデバイスの FPGA ロジックに画像処理タスクを移動すると、システム パフォーマンスが 10 倍に向上することを示しました。

## SDx による FPGA と SoC デザインの民主化

SDx ソフトウェア定義開発環境で、ザイリンクスは、主要市場のソフトウェア開発者およびシステム設計者を対象とする一連のハイレベル デザイン入力環境を構築しています。SDAccel と SDSoC はいずれも Vivado フロー全体を内部で自動的に実行するため、Vivado ツールを直接使用する必要はなく、ハードウェア技術者への引き継ぎも不要です。一方 SDNet は、Vivado HLS にアクセスせず、ライン カード設計者を対象とする独自の 2 段階利用モデルを使用します。

第 1 段階では、ライン カード設計者が、難解なマイクロコードの代わりに直感的な C 型の高級言語を使用して、ネットワーク ライン カードの要件設計と仕様作成を行います。SDNet 開発環境は、この仕様に基づいてデザインの RTL 版を生成します。このフローでは、ハードウェア技術者が RTL をターゲット FPGA にインプリメントする必要があります。

第 2 段階では、SDNet により、ネットワーク会社がプロトコルのテストと更新を高級言語で行えます。ライン カードが現場に導入された後でも、ハードウェア設計者の介入なしでカードの機能をアップグレードできます。このフローでは、各企業がカードの作成と更新を迅速に行えるため、ソフトウェア定義ネットワークに理想的な柔軟性を確保できます。

SDAccel と SDSoc の 2 つの新しい環境は、SDx の構想を新たなアプリケーションの領域に拡大します。

## SDACCEL によるデータ センターの単位ワット当たり性能の最適化

Data Center Journal 誌 2014 年 3 月号の記事によると、Google、Facebook、Amazon、Linked-In などの大手企業の中核をなす大規模データ センターの消費電力は、「世界の電力需要全体の 3% を超える規模となり、2 億トンの CO2 排出量に相当します。」この膨大な消費電力のため、データ センターの電気料金は年間 600 億ドル以上に上ります。この大きな消費電力は、大手

インターネット企業の収益の足かせとなるだけでなく、地球環境の面でも甚大なコストをもたらします。

クラウド コンピューティングとビッグ データ分析を利用する企業の増加、ビデオ / ストリーミング メディアの世界的な浸透、ワイヤレス ネットワークに参加し、5G 通信にアップグレードするユーザーの増加に伴って、数多くのデータ センターへの需要は急激に増大しています。Data Center Journal 誌の記事によると、現在の動向から予測されるデータ センターのトラフィック量は、2017 年までに年間 7.7 ゼタバイトに達します。データ センターの消費電力も、このまま放置すれば天文学的な数字に達しそうです。現在、この膨大な消費電力の主な原因となっているのは、ほとんどのデータ センター機器の基盤を構成しているインテルの x86 プロセッサです。現在の MPU は、満足のいく (ただし最適ではない) パフォーマンスを提供するために大量の電力を消費しています。

(世界的に人材の供給が豊富な) ソフトウェア技術者にとって、プログラミングが最も容易なデバイスはこれらの MPU です。データ センターの問題のうち性能に関わる部分を解決するために、多くの企業は、Graphics Processing Unit (GPU) を搭載した機器や GPU カードによってアクセラレートされた CPU システムを開発してきました。データ センター アプリケーションでは、GPU は CPU よりもはるかに優れたパフォーマンスを発揮しますが、残念なことに、はるかに大量の電力を消費します。その結果、パフォーマンスは極めて高速になりますが、消費電力が極端に悪化します。

パフォーマンスと消費電力の両面で最適解を得るために、多くの企業が FPGA 中心型の手法に移行し、FPGA と他のプロセッサを組み合わせて、データ センター機器のパフォーマンスを最大限に高めようとしています。

多くのデータ センター機器ベンダーは、ディスクリート FPGA とディスクリート CPU を組み合わせた場合、カード 1 枚当たりの消費電力のわずかな増加でパフォーマンスが飛躍的に向上し、単位ワット当たり性能が大幅に改善されることを実証しています。他方、1 つの SoC 上で x86 プロセッサ コアと FPGA ロジックを相互接続して組み合わせたチップを使えば、単位ワット当たりの性能がさらに改善されると考えているベンダーもあります。さらに、一部のベンダーの構想では、1 つの SoC 上に FPGA ロジックと ARM 64 ビット プロセッサ IP コアを統合することで、さらに消費電力を削減し、同程度に高性能なソリューションを実現できます。

データ センターへの FPGA の採用には、プログラミングの問題が主なハードルとなっていました。データ センター開発者は、通常は純粋なソフトウェア プログラミングの知識を背景として、x86 ベースのアーキテクチャのプログラミングに慣れ親しんでいます。最初のステップとして、開発者が CPU のプログラムからより高速な GPU へ移行しやすいように、業界ではオープンソースの OpenCL 言語を開発しました。この 2 年間で OpenCL はさらに進化し、FPGA をターゲットにできるようになりました。このような発展は、将来のデータ センター機器アーキテクチャとユビキタス ネットワークの新たな可能性を開くものです。

SDAccel 環境の発表により、ザイリンクスは、このようなプログラミングのギャップを解消し、ハードウェア技術者が設計に介入することなく、データ センター技術者が OpenCL、C、または C++ を使って FPGA ベースのプラットフォームをプログラムできる基盤を提供します。ザイリンクスのデザイン メソドロジ マーケティング担当シニア ディレクターであるトム フェイスト (Tom Feist) によると、OpenCL、C、C++ 用の SDAccel 開発環境では、データ センター機器のプログラマーは、CPU/GPU ベースのシステムに比べて単位ワット当たりの性能が最大 25 倍高い機器を開発できます。

フェイストによると、ザイリンクスの 20nm Kintex® UltraScale FPGA が動作するアクセラレーション カードと x86 CPU のペアをターゲットとする SDAccel フロー (図 2 の SDAccel 開発環境のデモを参照) では、ソフトウェア開発者は、アクセラレーションを必要とするアプリケーションを特定し、カーネルを OpenCL でコーディングおよび最適化し、アプリケーションを CPU 用にコンパイルして実行します。次に、サイクルしてカーネルの推定とデバッグを行い、サイクル アキュレートなモデルを作成します。次に、SDAccel を使ってコードをコンパイルし、FPGA 内に自動的にインプリメントします (Vivado ツールが内部で実行されているために可能)。次にアプリケーションを実行し、CPU のみで実行した場合と比較して、カードを使ってアクセラレートした場合のアプリケーションのパフォーマンスを検証できます。「開発者は、パフォーマンスと消費電力の最適なバランスが見つかり、CPU/GPU インプリメンテーションに比べて単位ワット当たり性能が最大 25 倍に達するまで、このサイクルを反復できます」とフェイストは述べています。

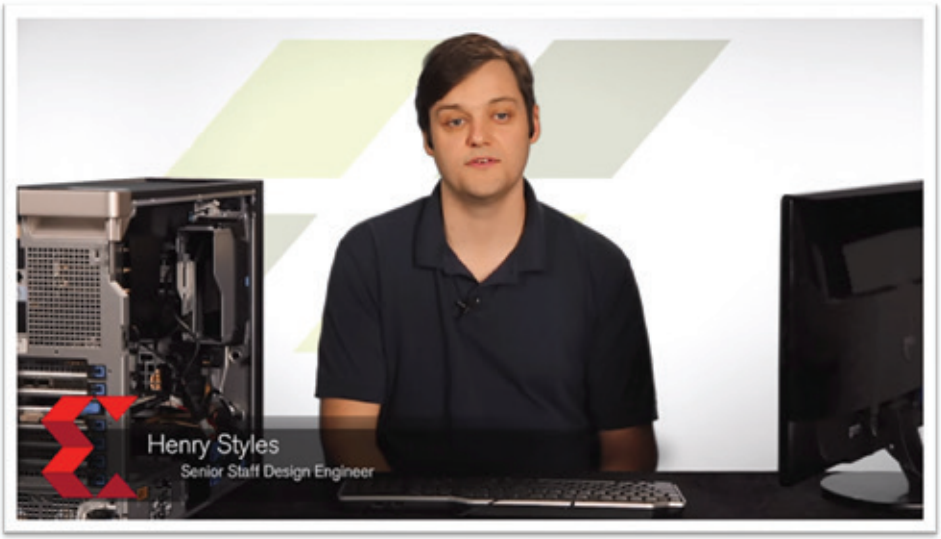



図 2 – この SDAccel 開発環境のデモでは、設計技術者のヘンリー スタイルズ (Henry Styles) が、SDAccel 開発環境を使用して、Alpha Data 社の ADM-PCIE-7V3 アクセラレータを搭載した標準 x86 64 ビット ワークステーションを利用したアクセラレーションを実現する方法について説明します。


図 3 – この SDSoC 開発環境のデモでは、プリンシパル エンジニアのジム ファン (Jim Hwang) が、SDSoC 環境を使用して、毎秒 60 フレームで動作する HD 1080pのライブ ビデオ ストリーム内で、動きを検出し、動きのエッジを挿入する簡単な画像処理パイプラインを作成します。

本号には、フェイストと同僚による SDAccel 環境の詳しい解説が掲載されています (36 ページを参照 )。もう 1 つの記事では、SDAccel の実際の動作を説明します (40 ページを参照 )。

## SDSoC による革新的なエンベデッド システム開発者の拡大

SDNet 環境では、ライン カード開発者が独自の「ソフト定義型」手法で次世代ネットワークを迅速に開発し、SDAccel 環境では、データ センター機器ベンダーが単位ワット当たり性能の非常に高い次世代データ センターを構築できます。一方、新しい SDSoC 開発環境は、ザイリンクスのユーザーに最も幅広い影響を与えそうです。なぜなら、SDSoC は膨大な数のエンベデッド システム設計チーム、特にザイリンクスの Zynq SoC 市場の大半にわたるソフトウェア設計技術者をターゲットとする環境だからです。SDSoC 環境では、ユーザーは C および C++ を使って、単に Zynq SoC ベースのハードウェア プラットフォームを動作させるエンベデッド システムのプロセッサをプログラムするだけでなく、ロジックをコンフィギュレーションできるようになります。

SDSoC プロダクト マネージャーのニック ナイ (Nick Ni) は次のように述べています。「ソフトウェア開発者は、ハードウェア技術者が介入することなくマザーボード、ASSP プラットフォーム、および ASIC をプログラムすることに慣れています。SDSoC 開発環境では、ASSP と同じ方法で Zynq SoC および MPSoC プラットフォームをプログラムできます。ただし、SDSoC のユニークな点は、Zynq SoC および Zynq UltraScale+ MPSoC プラットフォームをターゲットとして、Eclipse IDE 環境内で C または C++ を使って完成したシステム デザインを作成できることです。」

ナイによると、SDSoC 環境を使用する場合、エンベデッド ソフトウェア開発者は、C または C++ でデザインを作成し、デザインのどの部分が Zynq SoC のプロセッシング システム上で最適な動作をしていないかを確認するテストを実行します。疑わしいコードがハイライトされたら、そのコードを自動的に分割して Zynq SoC のプログラマブル ロジックに振り分けるように SDSoC 環境に指示し、システム パフォーマンスを向上させます。ナイによると、SDSoC 環境では、ボタンをクリックするだけでソフトウェアの機能を FPGA ロジックに移動できます。ハードウェア技術者が介入する必要はありません。SDSoC 内のコンパイラは、Vivado プロジェクト全体を生成し、ターゲットとするハードウェア プラットフォーム用のブート可能なソフトウェア イメージを生成します。

「Zynq SoC 用の SDSoC 環境は、基本的には、エンベデッド ソフトウェア技術者がシステム技術者の役割を果たすことを可能にします」とナイは述べています ( 図 3 の SDSoC 開発環境のデモを参照 )。

SDSoC 環境は、ユーザーが Zynq SoC のロジック内でアクセラレートするように指定した C/C++ コードを処理するマクロ コンパイラを利用します。SDSoC 環境は、Vivado Design Suite を内部で動作させて、このコードを自動的に IP ブロックに変換し、このブロックをデバイスのロジック内にコンフィギュレーションし、自動的にソフトウェアによるドライバーを生成します。

SDSoC 環境は、ZC702 および ZC706 などの Zynq All Programmable SoC ベースの開発ボード用と、ZedBoard、MicroZed、ZYBO およびビデオ / 画像開発キットなどのサードパーティ製特定市場向けプラットフォーム用に、ボード サポート パッケージ (BSP) を提供します。

「ザイリンクスは、Zynq SoC 搭載システムを開発するサードパーティ プラットフォーム開発会社の増加に対応して、今後数カ月にわたって SDSoC に BSP を追加していきます。SDSoC は、ザイリンクスだけでなく、Zynq SoC を利用したプラットフォームを開発する企業のユーザー基盤も拡大します」とナイは述べています。

ナイによると、SDSoC 環境の主な目標は、非常に多数のエンベデッド ソフトウェア設計者がザイリンクスの Zynq SoC を搭載したシステム全体を開発できるようにすることですが、従来の FPGA 設計経験のあるユーザーと設計チームにも、この環境は大きなメリットをもたらします。

「SDSoC 環境では、設計チームはシステム アーキテクチャを C および C++ で迅速に設計し、必要とする最適なパフォーマンスが得られるように、さまざまなコンフィギュレーションを試すことができます。FPGA 設計者を含む設計チームでは、FPGA 設計者が Vivado ツールを使用して IP ブロックとロジック レイアウトをさらに最適化できます」とナイは述べています。

アライアンス メンバーによる SDSoC 環境のサポートに関する最新情報は、本号の Xpedite のセクション ( 英語版 66 ページ ) を参照してください。ザイリンクスの All Programmable Abstraction の詳細は、http://japan.xilinx.com/products/design-tools/all-programmable-abstractions.html を参照してください。

# Enabling a JPEG 2000 Network for Professional Video

# プロ仕様ビデオ用のJPEG 2000 ネットワークを実現

**Jean-Marie Cloquet**
Manager, Image Processing Division
Barco Silex
*jean-marie.cloquet@barco.com*

# ザイリンクスと Barco Silex 社から JPEG 2000 ビデオ伝送の新しいリファレンス デザインがインターネット プロトコル ネットワークで提供

優れた画質を特長とする JPEG 2000 は、テレビ放送事業者のコントリビューション ネットワークのビデオ伝送など、高画質ビデオ圧縮の優れた標準として注目を浴びました。その結果、ビデオ機器メーカーは JPEG 2000 エンコーダーおよびデコーダーを各種の伝送ソリューションに組み込み始めましたが、これらのソリューションがサポートするインターフェイスは多様であり、独自規格に基づくプロトコルを採用している場合もあります。

機器間の相互運用性がないこの傾向が続くと、ビデオ サービス プロバイダーは、限られた少数のベンダーの製品に縛られることになります。この難題の解決策として、2013 年 4 月、インターネット プロトコル (IP) ネットワーク上のビデオ伝送に関する Video Services Forum (VSF) の TR-01 勧告 (相互運用可能な機器の仕様 ) が公開されました。ザイリンクス アライアンス プログラムの認定メンバーである Barco Silex 社とザイリンクスは、素早く提携を結ぶと、相互運用性の実現に向けた取り組みに着手しました。

このたび Barco Silex 社は、VSF TR-01 勧告のリファレンス インプリメンテーションを完成しました。この JPEG 2000 対応マルチチャネル Video-over-IP ソリューションは、先頃ザイリンクスの Web サイトで公開されました。このソリューションはザイリンクスと Barco Silex 社が開発した IP ( 知的設計資産 ) コアをベースにしており、放送機器メーカーはこのデザインを簡単にカスタマイズして自社の機器に組み込むことができます。Barco Silex 社のこの取り組みは、米国テレビ芸術科学アカデミーの 2014 年度テクノロジ & エンジニアリング エミー賞を受賞しました ( 図 1)。

## 優れたビデオ圧縮方式の探求

JPEG 2000 は古い JPEG 規格に代わるもので、JPEG や MPEG などの他の一般的なフォーマットに比べて多くの利点を備えています。2004 年には、JPEG 2000 は、ハリウッドの映画産業が後援する Digital Cinema Initiatives (DCI) 仕様を通じて、デジタル シネマの画像圧縮方式の事実上の標準になっていました。視覚情報の損失のないデータ圧縮を可能にする JPEG 2000 は、セキュリティ、アーカイブ、医療などのアプリケーションには理想的な方式と言えます。

放送業界も JPEG 2000 に注目していました。放送事業者とビデオ サービス会社は、いわゆる「コントリビューション ネットワーク」内のポストプロダクション施設およびストリーミング施設に、大量のライブ ビデオを ( 遅延や画質の低下が起こらないように ) 伝送する必要があります ( 図 2)。したがって、プロ仕様のビデオ業界で特に重視されるのは、視覚情報の損失のないデータ圧縮の可能性であり、画質を維持しながら効率的な格納と伝送を可能にする圧縮方式が求められます。

また、JPEG 2000 の革新的な技術は、これ以外の面でも放送業界を大きく進化させました。MPEG フォーマットではビデオ フレームがグループにまとめて圧縮されるのに対して、JPEG 2000 ではビデオ ストリームの各フレームが個々に静止フレームとして圧縮されます。シングル フレーム圧縮方式により、レイテンシの低減だけでなく、フレームごとの後処理と編集も容易に行えます。また、JPEG 2000 ストリームは部分的に伸張および視聴が可能なため、同じストリームからの多様なアプリケーションと視聴体験が可能になります。

もう 1 つの大きな利点として、ストリームの送信エラーに対する耐性が高まります。JPEG 2000 では、フォワード エラー訂正 (FEC) を使用して送信エラーを訂正できない場合でも、デコーディング後のエラーの視覚的な影響が他のコーデックよりも小さくなります。最後に、JPEG 2000 は、エンコーディング / デコーディング プロセスを複数回実行しても画質を維持できます。さまざまな段階でビデオを管理するコントリビューション ネットワークでは、この特性は非常に重要です。

この利点を認識した機器メーカーは、まもなく JPEG 2000 エンコーダー / デコーダーを自社のビデオ機器に組み込み始めました。しかし、複数の施設間でビデオ ストリームを伝送する場合、独自規格に基づくプロトコルを含む実装方法の幅広い選択肢からどれを選ぶかは、メーカー間で統一されていませんでした。ビデオ サービス プロバイダーにとって困った点は、ごく少数のベンダーの製品にいつまでも縛られることになり、アプリケーションに最もよく適合する、費用対効果に優れたインフラストラクチャを構築できないことでした。

## ビデオ伝送の標準化のための勧告

したがって、サービス プロバイダー側では、既存の機器と将来の機器の良好な相互運用性を保証する、標準化された伝送方式を明確に要求していました。プロバイダー各社が必要としていたのは、一般的なネットワーク アーキテクチャになりつつある IP ネットワーク向けに最適化された伝送方式と、高スループットのデータ伝送に対応する標準化された機器です。そこで 2007 年から、米国映画テレビ技術者協会 (SMPTE) は Video Transport over IP の標準を公開し、それ以来拡張を繰り返してきました。SMPTE 2022 には、特に、MPEG-2 トランスポート ストリーム内の固定ビット レート (CBR) ビデオ信号用の IP プロトコルが含まれています ( 圧縮ビデオ用の SMPTE 2022 1 および 2 と非圧縮ビデオ用の SMPTE 2022 5 および 6)。

これらの仕様を基礎として、Video Services Forum (VSF) は、2013 年に VSF TR-01 勧告 (「MPEG-2 TS over IP 内の JPEG 2000 ブロードキャスト プロファイル ビデオの伝送」というタイトルの技術的勧告 ) を公開しました。VSF は、ビデオ ネットワーキング技術の相互運用性、品質測定基準、啓発活動を専門とする、サービス プロバイダー、ユーザー、メーカーで構成される国際団体です。

VSF TR-01 に準拠したすべてのデバイスは、放送業界における非圧縮ポイントツー ポイント ビデオ伝送の従来規格である Serial Digital Interface (SDI) 信号を入力として受け入れます。デバイスは、アクティブなビデオ データ、オーディオ データ、および補助データ ( キャプションなど ) を抽出し、

JPEG 2000 フォーマットでビデオを圧縮します。

そこから得られたビデオ ストリームは、オーディオ データおよび補助データと一緒に MPEG-2 トランスポート ストリームにマルチプレクスされます。このストリームは、SMPTE2022 に従って再び Real-time Transport Protocol (RTP) ストリームにカプセル化され、IP ネットワークを介して受信側デバイスに送信されます。受信側デバイスは、RTP/IP ストリームのカプセル化解除、MPEG-2 トランスポート ストリームのデマルチプレクス、JPEG 2000 のデコードを実行し、ビデオ データ、オーディオ データ、および補助データを SDI 信号として出力します。

## FPGA ベースのリファレンス ソリューションのインプリメント

2012 年 9 月、VSF 勧告の公開に先立って、ザイリンクスと Barco Silex 社は Video-over-IP ソリューションの開発に向けた提携を発表しました。その目標は、ハードウェア検証済みの IP ( 知的設計資産 ) コア、リファレンス デザイン、システム インテグレーション サービスの包括的プラットフォームを提供することです。この取り組みでは、Barco Silex 社は、ザイリンクスのコア (SMPTE 2022、SMPTE SDI、イーサネット MAC) と Barco Silex 社の高性能 JPEG 2000 コアおよび DDR3 メモリ コントローラー コアをマッチングさせるシステム インテグレーターとしての役割を担いました。その目標は、放送機器メーカーが製品開発を加速し、既存の製品と現在開発中の製品に最新の Video-over-IP 機能を追加できるようにすることです。

この枠組みの中で、ザイリンクスと Barco Silex 社は、4 チャネルのトランスミッター / レシーバー プラットフォームで構成されるリファレンス デザインを完成しました ( 図 3)。トランスミッターは、VSF TR-01 標準に従って最大 4 本の SDI 高精細 (HD) ストリーム (1080p30) を受け入れ、オプションにより JPEG 2000 で圧縮し、1Gbps ( 圧縮時 ) または 10Gbps ( 非圧縮 ) イーサネット上に送信します。逆に、レシーバー プラットフォームは、IP ストリームを受信し、カプセル化を解除して伸張し、最大 4 本の SDI HD リンクに載せます。

トランスミッター プラットフォーム内では、ザイリンクスの SMPTE SDI コアが、入力される SDI ビデオ ストリームを受信します。非圧縮パス上では、これらの SDI ストリームはザイリンクスの SMPTE 2022-5/6 Video-over-IP トランスミッター コアによってマルチプレクスされて固定サイズのデータグラムにカプセル化され、ザイリンクスの 10 ギガビット イーサネット MAC (10GEMAC) コアおよび 10G PCS/PMA コアを介して送信されます。

圧縮パス上では、SDI ストリームは、まず圧縮のために JPEG 2000 エンコーダーに送られます。次に、Barco Silex 社がインプリメントした専用 TS エンジン コアによって、VSF TR-01 に従って MPEG-2 トランスポート ストリーム パケットにカプセル化されます。最後に、SMPTE 2022-1/2 Video-over-IP トランスミッター コアが、これらのストリームを固定サイズのデータグラムにパックし、1G TEMAC を介して送信します。あるいは、10GEMAC コアおよび 10G PCS/PMA コアを使用して、これらのストリームを非圧縮ビデオ チャネルと合わせて 10 ギガビット リンク上でマルチプレクスすることもできます。

レシーバー プラットフォーム上では、非圧縮ストリームのイーサネット データグラムは 10GEMAC 上で収集されます。SMPTE 2022-5/6 Video-over-IP レシーバー コアは、これらのデータグラムをフィルタリングし、カプセル化解除して個々のストリームにデマルチプレクスし、SMPTE SDI コアを介して SDI ビデオを出力します。圧縮ストリームのイーサネット データグラムは、10GEMAC 上で収集され、SMPTE 2022-1/2 Video-over-IP レシーバー コアおよび TS エンジンによってカプセル化解除され、JPEG 2000 デコーダーに供給されます。デコーダーの出力ビデオは SDI に変換され、SMPTE SDI コアに送られます。

4 チャネルのそれぞれについて、他のチャネル上の処理とは無関係に、圧縮パスまたは非圧縮パスを選択できます。

## 相互運用性の高いソリューションの基盤を構築

ザイリンクスと Barco Silex 社は、Zynq®-7000 All Programmable SoC と Kintex®-7 FPGA の 2 種類のプラットフォームにリファレンス デザインをインプリメントしました。しかし、リファレンス デザインに使用されているブロックは、低コストの量産型アプリケーションから非常に要求の厳しい高性能アプリケーションに至るまで、機器メーカーの広範囲にわたるシステム要件に対応するソリューションに組み込むことができます。ザイリンクスの SMPTE 2022 ブロックおよびイーサネット MAC LogiCORE ™ ブロックなど、リファレンス デザインに使用した知的設計資産 IP コアは、UltraScale ™ レベルに至るまで、広範囲にわたるザイリンクス FPGA システムで利用可能です。

このリファレンス デザインは、エンコーディングおよびデコーディング用に、Barco Silex 社の JPEG 2000 エンコーダーおよびデコーダー IP コアを組み込んでいます。

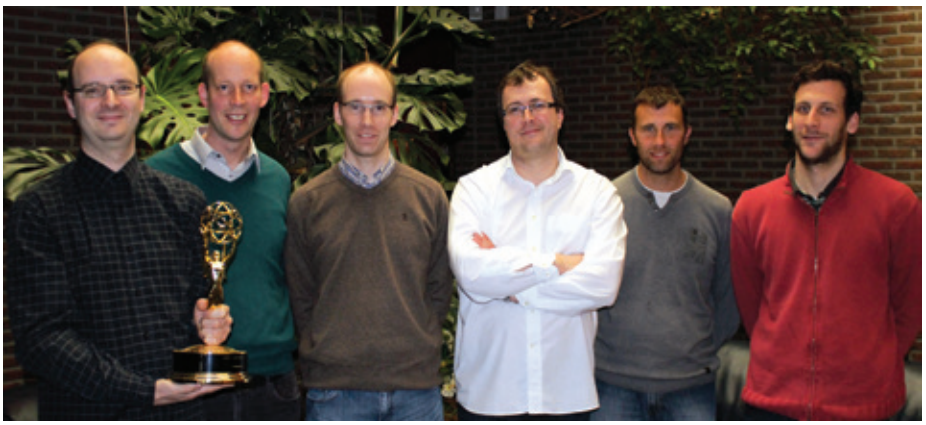


図 1 – Barco Silex 社のビデオ チームが担当したリファレンス デザインは、JPEG 2000 の相互運用性の標準化および製品化に関する 2014 年度テクノロジ & エンジニアリング エミー賞を受賞しました (左から、Luc Ploumhans、Sake Buwalda、François Marsin、Jean-François Marbehant、Jean-Marie Cloquet、Vincent Cousin)

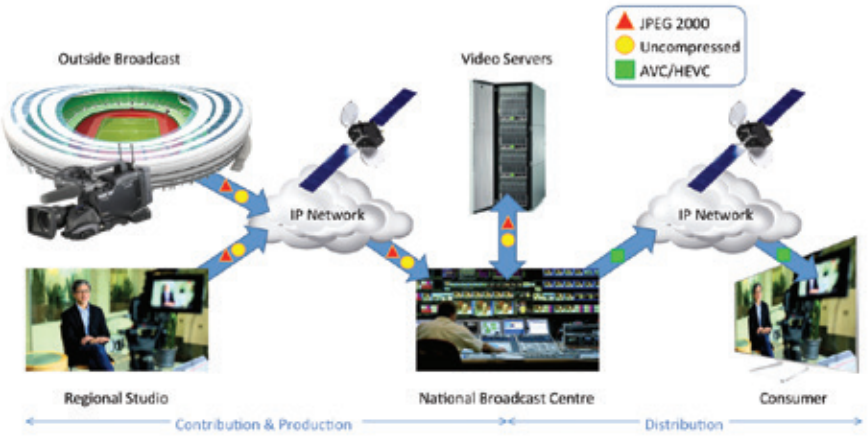



図 2 – 放送事業者のコントリビューション ネットワーク

これらの IP コアは、720p30/60、1080i、1080p30/60、および 2K/4K/8K ビデオの高性能同時マルチチャネル JPEG 2000 エンコーディング / デコーディング用に広く採用されている、シリコン検証済みのシングル FPGA ソリューションです。さらにこれらのコアは、市販されている JPEG 2000 オプションを非常に広範囲にわたってサポートします。複数のビデオ ストリームをスムーズな高データレート システムに集約するために必要不可欠なのが、Barco Silex 社の DDR3 メモリ コントローラーです。この高度なカスタマイズが可能なこのコントローラーは、アクセスをリオーダーして SDRAM の複数バンクに振り分けることによって高帯域幅を実現できるように最適化されています。

ザイリンクスと Barco Silex 社は、バージニア州アーリントンで 2014 年 2 月に開催された年 1 回の VidTrans カンファレンスにおける相互運用性の公開デモンストレーションで、このリファレンス デザインの第 1 世代を展示しました。VSF が企画したこのテストでは、参加 10 社 (Artel、Barco Silex、Ericsson、Evertz、Imagine Communications、IntoPIX、Media Links、Macnica、Nevion、およびザイリンクス ) が提供するテクノロジと機器を相互接続して、JPEG 2000 エンコーダーおよびデコーダーを使用してリアルタイムで圧縮された 720p30 および 1080i60 HD コンテンツのライブ送信を実演しました。

その数か月後、Barco Silex 社は、このリファレンス デザインが 4K および超高精細 (UHD) 信号の処理に対応できることを実証しました。次世代ビデオ配信用に提唱されている主な規格の 1 つである 4K ビデオは、1080p ビデオの 4 倍のピクセルを伝搬し、より鮮明な映像とより大型のビデオ ディスプレイを実現します。リファレンス デザインの 4 つの入力チャネルをクワッド SDI モード (4 本の SDI ケーブルで 4K を伝搬 ) で使用することで、4K 信号を入力として受け入れ、IP ネットワーク上に送信することも可能になりました。この手法により、このリファレンス デザインは最大 4K までのビデオ解像度にいつでも対応できます。

## FPGA によるビデオ業界の進化

ザイリンクスとビデオ専門企業である Barco Silex 社の協力関係の目標は、FPGA ベースのプラットフォームの高性能と柔軟性を、プロ仕様のビデオ市場に活かすことでした。機器メーカーは、Barco 社の JPEG 2000 コアとザイリンクスのトランスポート コアを組み合わせることで、標準化された放送機器を迅速に開発、更新し、製品への投資を将来にわたって保護できます。

このリファレンス デザインは、ビデオ業界における IP ネットワークの利用が一般化してきた時期に登場しました。機器メーカーがこの新しい市場でシェアを獲得できるかどうかは、どれだけ早く製品を市場に供給できるかにかかっています。ザイリンクスの FPGA をベースとする再プログラム可能なソリューションを使えば、標準がまだ進化の途上にある間に、製品を市場に投入できるのです。

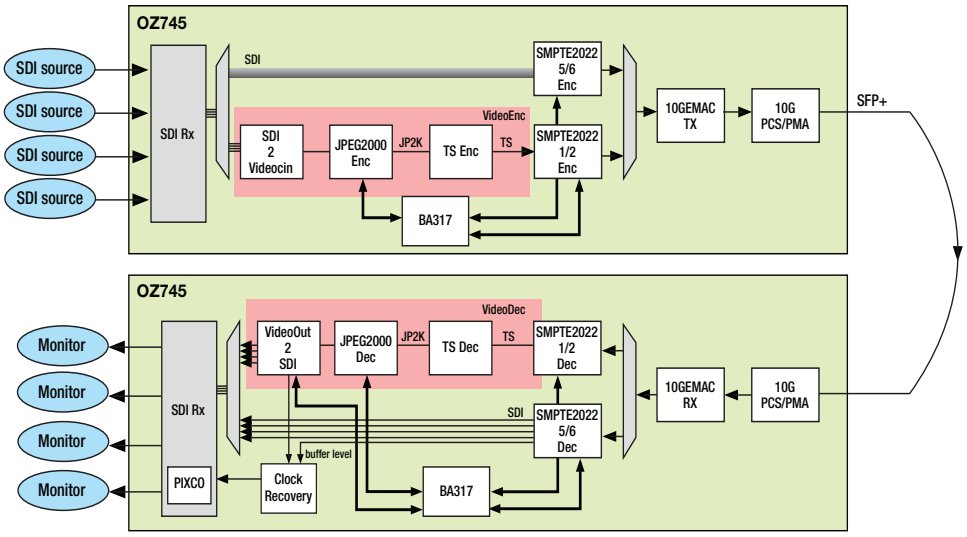



図 3 – トランスミッター/レシーバー プラットフォームで構成されるリファレンス デザインの概要

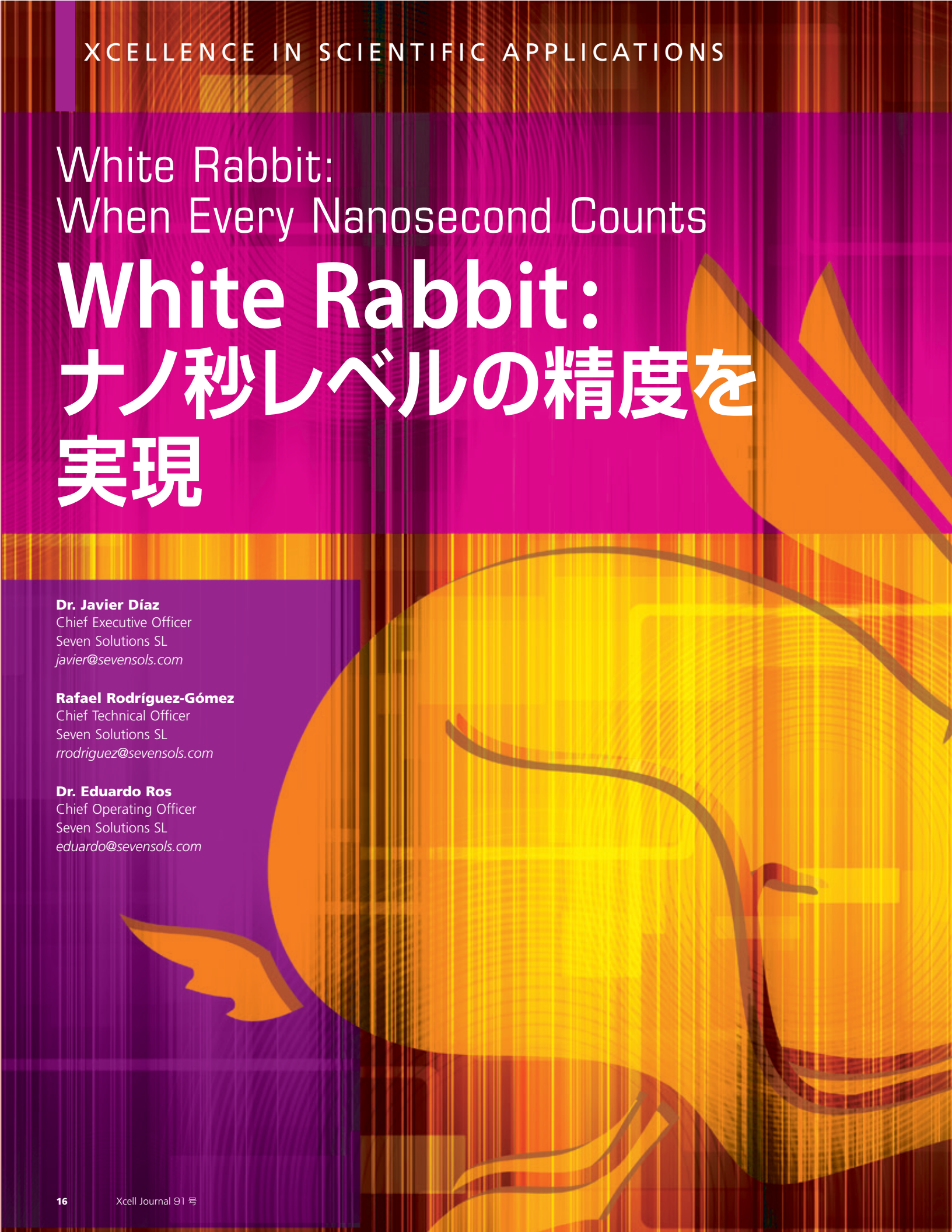
# White Rabbit: When Every Nanosecond Counts

# White Rabbit: ナノ秒レベルの精度を実現

**Dr. Javier Díaz**
Chief Executive Officer
Seven Solutions SL
*javier@sevensols.com*

**Rafael Rodríguez-Gómez**
Chief Technical Officer
Seven Solutions SL
*rrodriguez@sevensols.com*

**Dr. Eduardo Ros**
Chief Operating Officer
Seven Solutions SL
*eduardo@sevensols.com*



# イーサネット ベースの高精度タイミング ソリューションが、ザイリンクスの FPGA および SoC デバイス ベースの製品として市場に提供されます。

最近の通信および情報科学の進歩とともに、産業用の時刻転送の要件は、科学技術アプリケーションの要件に近づいています。たとえば、今後展開される 100G イーサネット ネットワークおよび 5G モバイル通信には数ナノ秒の範囲内のタイミング精度が必要とされ、電力分配用のスマート グリッドにはサブマイクロ秒の精度が要求されます。高頻度トレーディング (あるいは株式取引全般) のタイムスタンプ処理には、時刻認証局から事務センターに時刻情報を配信する信頼性の高いメカニズムが必要です。さらに、GPS や Galileo など、GNSS テクノロジに基づく測位サービスは、高精度の同期化メカニズムを活用しています。

CERN (欧州原子核研究機構) で生まれた White Rabbit と呼ばれるイーサネット ベースの技術では、これらのアプリケーションの高精度なタイミング要件を満たすことが約束されています。『不思議の国のアリス』の時間にとりつかれた野ウサギにちなんで命名された White Rabbit は、PTPv2 (IEEE-1588v2) およびシンクロナス イーサネットなどの標準メカニズムをベースとして、これらの標準との互換性を維持しながら、サブナノ秒の精度を実現するために改良を加えられた技術です。White Rabbit は、基本的に、長距離リンク上で自己較正を実行し、精度の低下をごくわずかに抑えて、非常に多数のデバイスに時刻を分配できます。

Seven Solutions SL 社は、White Rabbit (WR) の創生期である 2009 年から WR テクノロジの開発に取り組み、ザイリンクス All Programmable ソリューションを使用した WR 製品を市場に提供している新しい企業です。私たちの最新の製品は ZEN (Zynq® Embedded Node) ボードです。このタイミング ボードは、White Rabbit ネットワークの枠組みの中でクロックを同期化し、他のノードに正確なタイミング情報を提供する、高精度のリファレンス クロックを搭載しています。

## 時間の精度についての簡単な歴史

物理学者はこれまで常に正確な時間の重要性について理解し、長年にわたってさまざまな時間測定の手法を工夫してきました。天空走査に基づく簡単な手法（日時計、恒星走査）から、素粒子の世界の特性を利用した複雑なメカニズム（原子時計）まで、科学者は高精度なクロックの開発に懸命に取り組んできました。既存のクロックには、約 3 億年間の誤差を ±1 秒以内に抑えたものもあります。こうした高精度は、たとえば国立計量研究所のタイム スケールの維持など、多くのアプリケーションで鍵となります。

しかし、これらの極めて高精度なクロックは、高価格な上に非常に故障しやすく、大きな物理的空間を占有します。したがって、これらのクロックは、多くの場合、現実的な使用条件には適していません。実際、ほとんどのアプリケーションでは、低価格のクロック（水晶発振器）を組み込んだ電子機器が利用されます。多少資金に余裕があれば、さまざまな仕様の発振器の中から最適なものを選択できます。

簡単なアプリケーションの場合は、発振器の精度で十分です。しかし、多くのアプリケーション分野では、複数のデバイスが同時に動作するために同期通信またはグローバルな時間の概念が必要とされるため（分散型インスツルメンテーション）、互いに接続されていないこれらの「自走式クロック」を使用することはできません。設計者が高品質の発振器を組み込めば、この問題の一部は解決されるのですが、常に技術的に実現可能であるとは限りません。個々のクロックは依然として非同期型であり、周波数が多少ずれただけで、この手法は無効になります。

一方、チップスケール原子時計 (CSAC) などの高精度なクロックに基づくソリューションは、大量に利用するには価格が高すぎます。このような場合、代替手法として、（安定性が高いが通常は高価格な）リファレンス クロックから、正確に同期させる必要があるネットワーク内の他のすべての要素にクロック情報を分配することができます。問題は、どのようにそれを実現できるかです。

## 時刻転送の技術

時刻を分配する場合、多くの手法が考えられます。周波数の分配（回線を介して発振器の信号を送信する）と位相の分配（ネットワークのすべての要素で全く同一の時刻にイベントをトリガーする）は同じではないことに注意してください。

第 1 の問題（周波数の分配）は、たとえば同軸ケーブルまたは光ファイバーを使用して、クロックの発振を送信することによって解決できます。第 2 のシナリオ（位相の分配）では、回線上に毎秒送信されるパルスをエンコードし、このパルスをリファレンスとして使用して、新しい秒がいつ始まったかを認識できます。この手法は通常は PPS (Pulse-Per-Second) 信号と呼ばれます。

さらに、第 3 の問題があります。時刻を供給する必要があるのは、単にすべてのデバイスを同時にチック（かちっと音を立て）させたり（周波数）、カウントをいつ開始するかについて同じリファレンスを確保させたりする（位相）ためだけでなく、すべてのデバイスが同じ時刻になるように保証するためでもあります。したがって、中央のタイム サーバーから時刻情報を伝搬して、このメッセージの伝搬時間を測定し、その伝搬時間を各ノード内でアノテートすることにより、時刻の値を分配できます。周波数、位相 (PPS)、時刻の 3 つの要素が揃ったとき、そのネットワークは同期化されたと言えます。

現在の産業用ソリューションは、これらの機能をさまざまな方法で提供します。たとえば、GPS デバイスは、リファレンス 周波数 (10 ～ 50MHz)、PPS 信号、シリアル コードを供給し、（通常は NMEA プロトコルに基づく）時刻情報を提供します。さまざまな計装機器をさまざまな GPS レシーバーに容易に接続できるので、この手法は正確な同期を必要とする多くのシステムに広く使用されています。しかし、この手法は大量の低レベル信号を使用します。電力グリッド アプリケーションでは、時刻情報と PPS 情報を提供する IRIG-B と呼ばれる簡単なプロトコルに基づいて、これらの値が提供されます。かつては IRIG-B 手法は電力グリッドの同期化に十分に有効でした。しかし現在、「スマート グリッド」はますます複雑化し、新しいエネルギー監視アプリケーションが高精度を要求しているため、IRIG-B プロトコルでは対応が難しくなっています。

パケット ネットワークがほぼ全域的に普及した現在、スイッチング ネットワーク上の従来のメカニズムは、パケット ネットワークに適応する形で進化しています。SDH/SONET テクノロジは、Precision Time Protocol (PTPv2、すなわち IEEE-1588v2) とシンクロナス イーサネット (SyncE) に基づくソリューションへと少しずつ形を変えています。PTPv2 は、インターネットがネットワーク全体のコンピューターを同期させるのに使用するプロトコルである Network Time Protocol (NTP) が産業用に進化したものです。PTPv2 は、時刻の同期化の精度を大幅に向上させる、ハードウェア タイムスタンプのメカニズムを利用します。

第 2 のメカニズムである SyncE は、データ搬送波内にクロック信号をエンコードすることを可能にします。この方法で、ユーザーに対して透過的に、すべてのデバイスに対して時刻情報と周波数を分配できます。PTPv2 と SyncE の組み合わせにより、パケット ネットワークをシームレスに通信に利用できるので、この組み合わせは、現在では通信、電力グリッド、オートメーションの各アプリケーションにおける産業用時刻分配の最も一般的なソリューションになっています。位相の伝搬とシステムの拡張性に関して、若干の重要な問題が未解決のままになっていることに注意してください。

## 科学技術アプリケーション およびさらにその先へ

多くのアプリケーションでは、リファレンス クロック ソース情報をさまざまなデスティネーション ポイントに伝搬する必要があります。科学技術研究施設のインフラストラクチャには、おそらく最も厳しい条件で高精度な時刻分配が要求されます。CERN の LHC 加速器から、CTA、SKA、KM3NeT など電波天文学の大規模な分散型施設までのすべての施設が、超高精度な時刻分配と周波数分配を必要としています。

しかし、次世代の IT および通信アプリケーションにも、現在の標準的な手法では達成不可能な、非常に高精度な時刻転送が必要とされます。一例として、GPS の分野では、衛星信号伝搬時間の測定によって距離が測定されるため、測位と高精度の時間測定は密接に関連しています。一般的に、GNSS には電波妨害やなりすましの問題に対する脆弱性があります。したがって、時刻分配に GNSS を使用する場合は、重要なインフラストラクチャには補完的な冗長メカニズムとして（光ファイバーに基づく）地上設置型の

代替手段の使用を推奨します。

## White Rabbit ソリューション

White Rabbit (http://www.whiterabbit-solution.com/) は、正確なタイミングが得られるようにイーサネットを拡張した技術です。2009 年に CERN がオープンでコラボレーティブなソフトウェア / ハードウェア イニシアチブとして White Rabbit を構想すると、テクノロジ業界はその開発に熱心に参加しました。White Rabbit のソースは、さまざまな企業および研究機関の開発を奨励している Open Hardware Repository (OHWR、http://www.ohwr.org) で入手可能です。

スペインのグラナダに本社を置く Seven Solutions 社 (www.sevensols.com) は、当初から電子機器だけでなくファームウェアやゲートウェアを含む White Rabbit (WR) 製品の設計に協力していました。Seven Solutions 社は、WR テクノロジに基づくカスタム ソリューションおよびターンキー ソリューションも提供しています。

イーサネットの拡張版である White Rabbit テクノロジは、次期 Precision Time Protocol 標準 (IEEE-1588v3) で高精度プロファイルの枠組みとして検討されています。標準化されれば、図 1 に示すように、将来の広範囲にわたるテクノロジへの WR の統合が容易になります。

## White Rabbit テクノロジの分析

White Rabbit には、イーサネットの拡張という枠組みの中で ( したがって、イーサネット通信の構造を維持しながら )、最適なタイミング精度が得られるように設計された多くのメカニズムが組み込まれています。WR は、PTP、シンクロナス イーサネット、DMTD (Digital Dual-Mixer Time Difference) 位相トラッキングを統合しています。

Seven Solutions 社の新しい ZEN ボードは、White Rabbit の主要な要素を 1 つにまとめた製品です ( 図 2)。ザイリンクスの Zynq-7000 All Programmable SoC をベースにした ZEN ボードは、White Rabbit コア (WRC) と、高精度クロックを提供できるギガビット イーサネット MAC を実装しています。WRC にインプリメントされた同期化メカニズムは、次の要素で構成されます。

- 周波数の同期化 ( シンソナイゼーション ): データ搬送波内にクロック信号をエンコー

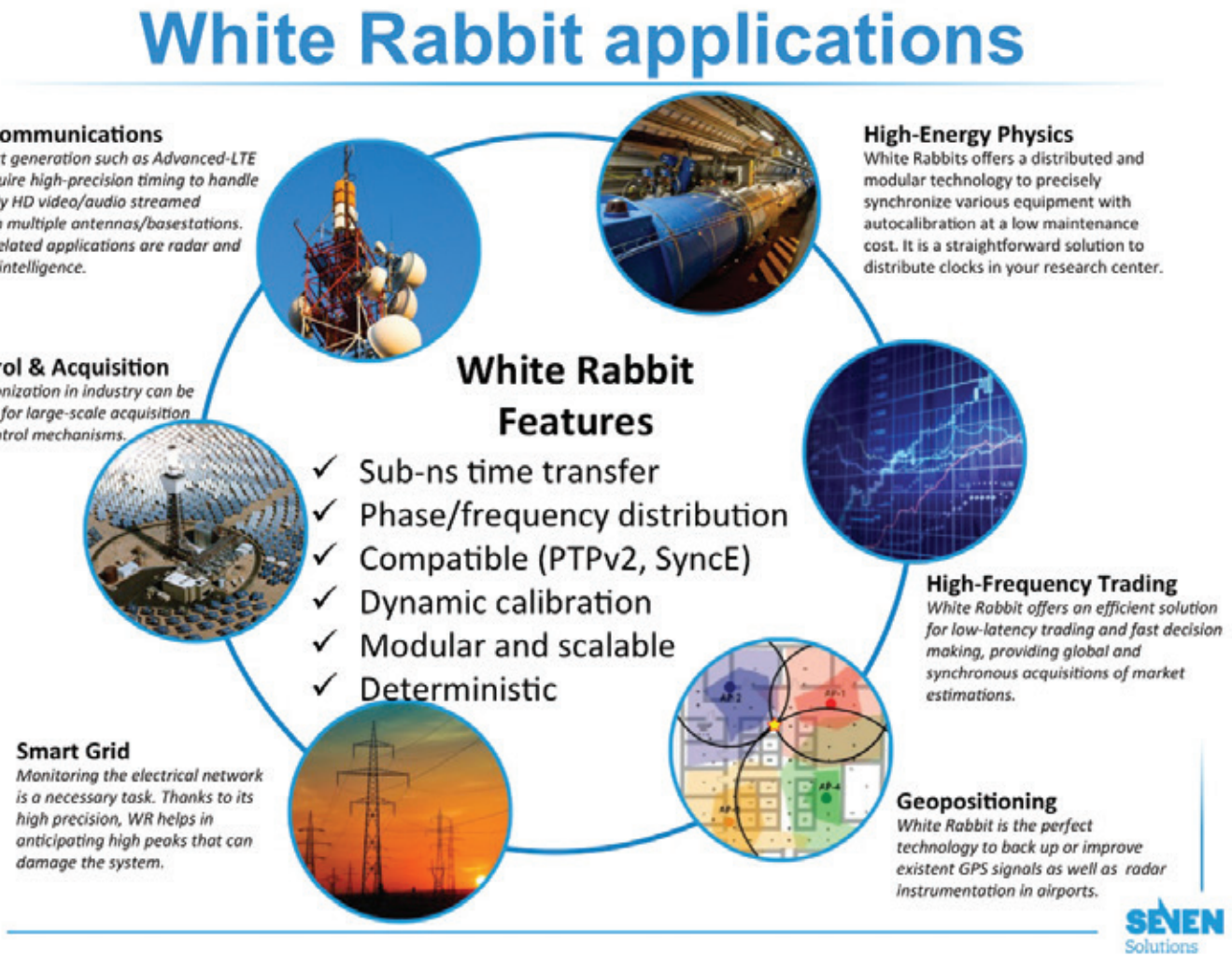



図 1 – White Rabbit アプリケーションの概要

ドする SyncE を使用して実現されます。すべてのノードが同じ周波数を使用することを保証するために、光リンクから復元される外部クロックに合わせて調整されたローカル発振器に基づくメカニズムを利用します。

- 位相の同期化 : 一つのノードの物理クロックはマスター エレメントに再送信され、同様にマスター エレメントの物理クロックは一つのノードに再送信されるため、マスターは ( スレーブから受信した ) クロック信号の位相とマスターそれ自体のクロック信号の位相を比較できます。位相のずれは、光ファイバーを介した信号の伝搬時間 (PTP を使用して正しく測定される ) に等しくなるはずです。この情報を得たマスターは、マスターのクロックとスレーブからのクロックの位相差を計算し、マスターと全く同じ値に位相をシフトすることをスレーブに要求します。このプロセスは、FPGA ゲートウェア内にデジタル DMTD をインプリメントすることによって、デジタル方式で実行されます。
- 時刻の同期化 : リンク伝搬時間を測定してグローバルな時間の概念を提供する、PTPv2 プロトコルを使用した結果として実現されます。White Rabbit は、各通信転送 ( ループの順方向と逆方向 ) に双方向光ファイバー内で異なる波長を使用したために生じる伝搬時間の非対称性も考慮に入れて、標準 PTP プロトコルの精度を向上させます。周波数と位相は既に同期化されているため、White Rabbit ネットワーク内のすべてのデバイスでグローバルな時間の概念を保証できます。

これらの処理はすべて、一部は適切な FPGA ゲートウェアを使用して、また一部はエンベデッド ソフト コアを使用して、WRC 内にインプリメントされます。White Rabbit 製品は、これらの各種クロック処理の実行に必要とされる、適切な発振器、PLL、タイミング エレクトロニクスを搭載しています。

1 つの事例として、ZEN ボードについて詳しく説明します。このボードはデュアル WRC (D-WRC) を搭載しています。これはオリジナルの WRC の改訂版であり、Seven Solutions 社がザイリンクス 7 シリーズ ベースの新しい製品ラインで開発したものです。D-WRC は、2 つの White Rabbit ノードを同期させることも、デイジーチェーン ネットワーク内の中間リンクとして機能することもできます。また、ZEN ボードは、D-WRC によって制御される、高精度、低ジッター、温度補償型のクロック リソースを搭載しています。

さらに、Zynq SoC は Linux OS 上で動作するデュアルコア ARM® Cortex™ -A9 プロセッサを搭載しているので、ユーザー アプリケーションの開発も容易に行えます。オンボードの Linux を使って、コンフィギュレーション用の Web サービス、ステータス監視用の SNMP のサポート、リモートからのファームウェアのロードと更新など、便利な新しい機能を利用できます。

ZEN ボードは、高精度な時刻供給機器として機能するように設計されています。このボードは、さまざまな接続や拡張に対応しており、非常に多様な用途に利用できます。

- IRIG-B I/O は、ZEN ボードで使用される

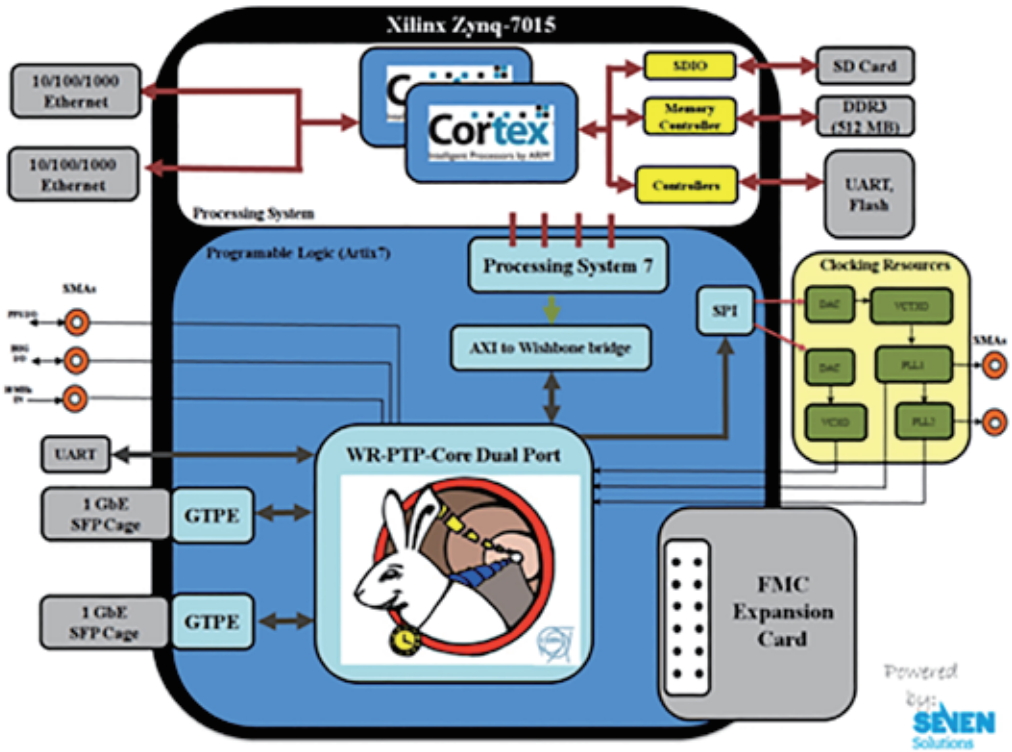



図 2 – ザイリンクス Zynq SoC デバイスをベースとする White Rabbit ゲートウェアの要素

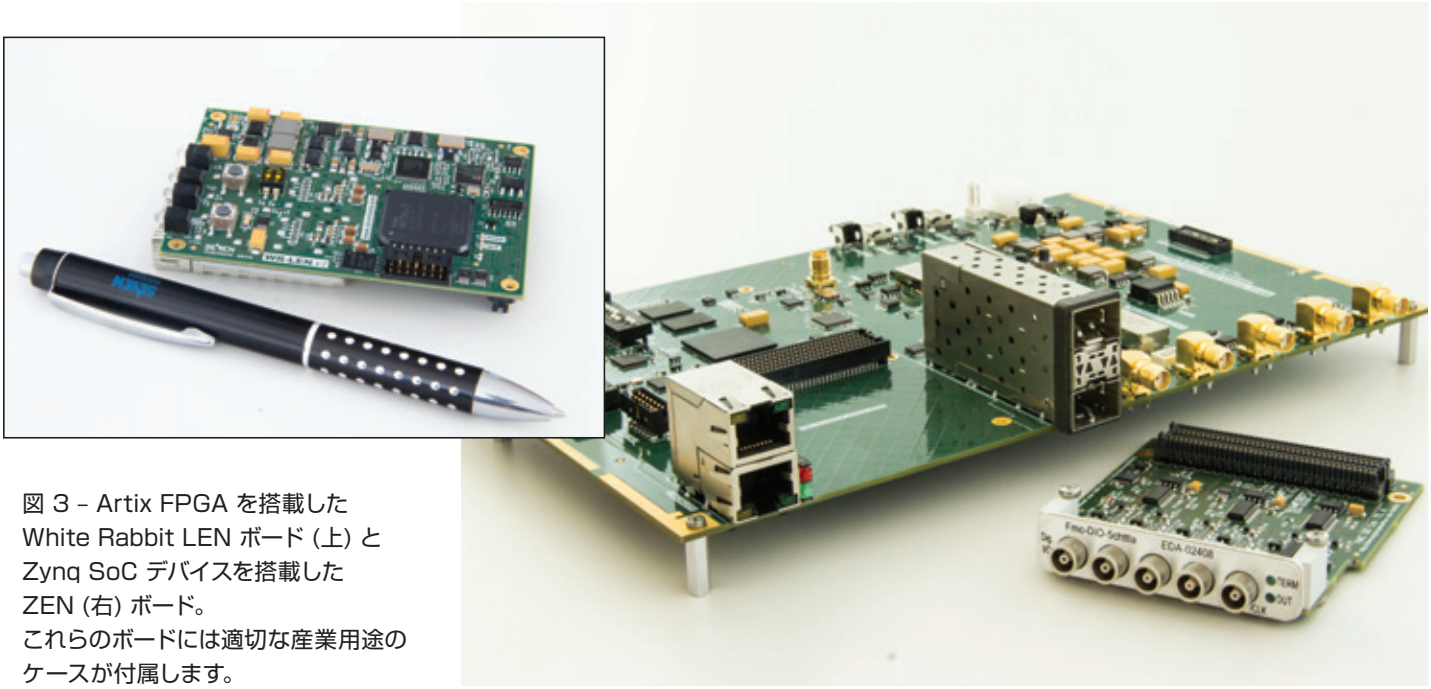


図 3 – Artix FPGA を搭載した White Rabbit LEN ボード (上) と Zynq SoC デバイスを搭載した ZEN (右) ボード。これらのボードには適切な産業用途のケースが付属します。

時刻です。マスターまたはスレーブのいずれでも機能します。

- ARM プロセッサに接続される 2 個の 10/100/1000 イーサネット ポートは、各種のネットワーク アプリケーション / プロトコルに使用できます (NTP、sNTP、PTPv2、管理機能など )。
- WR 準拠リンクの接続用に、2 個の SFP ケージが用意されています。
- SMA コネクタにより、ZEN ボードは、より高精度なクロック ( たとえば、GPS ソースや安定性の高い発振器など ) に同期し、WR によって同期化されるさまざまなクロック セットを提供できます。
- FMC コネクタに、WR プロジェクトの枠組みの中で開発されたメザニン ボードや、他の市販の産業用ボードを接続できます。これらの FMC カードは、ZEN ボードの可能性を広げ、多様な製品構成を可能にします。
- メモリ リソースには、SD、DDR3、フラッシュが含まれます。
- D-WRC および Cortex プロセッサの管理とデバッグ用に、2 個の UART-USB コネクタを搭載しています。

要約すると、ZEN ボードは、Zynq SoC の優秀性とそれがもたらす新しいレベルのシステム デザイン機能を提供する一方、サブナノ秒の精度で同期化を実現し、デイジーチェーン接続で動作するノードをエンド ユーザーに提供します。

## White Rabbit 対応機器

White Rabbit テクノロジは、CERN が推進する Open Hardware コミュニティ (Open Hardware Repository、OHWR) から生まれました。このテクノロジを素早く習得できるように、Seven Solutions 社は、SPEC と呼ばれる 2 個の Spartan®-6 搭載ボード (1 つはマスター、もう 1 つはスレーブとしてコンフィギュレーション可能 ) で構成される White Rabbit 入門キットを開発しました。その目的は、ユーザーが早期評価のためのさまざまな実験を実行することを奨励することです。

White Rabbit テクノロジの最も複雑な要素はスイッチです。Seven Solutions 社は、(CERN、GSI、および他のパートナーと協力して ) MicroTCA フォーム ファクターのメイン ボードを備えた 18 ポートの White Rabbit スイッチを開発しました。このスイッチの中核となる要素は Virtex®-6 (LX240T) FPGA です。このデバイスと、組込み Linux OS が動作する外部プロセッサ (ARM926E) の組み合わせにより、システム更新やファイル管理などの高度な処理を実行します。このスイッチは、18 個の GTX リンクを SFP に使用し、40 個の GPIO を汎用タスク (LED、SFP 検出など ) に使用します。このスイッチは非常に複雑な製品であり、標準的な通信ツールを使用しながら、タイミングの分配とデータ パケットの処理を実行できます。

先頃 Seven Solutions 社は、White Rabbit コアを LEN ボード上のザイリンクス Artix® FPGA ファミリに移植し ( 図 3)、既存の OHWR デバイスよりも費用対効果とエネルギー効率に優れたソリューションを実現しました。さらに Seven Solutions 社は、Zynq SoC デバイスをベースとする White Rabbit 製品の開発に成功しました。既に説明した WR-ZEN ノードは、ノードとコンピューターがすべて同じボードに統合される、汎用性の高い完全なシステム オン チップ手法を提示しています。このソリューションにより、コスト削減とシステムの柔軟性の向上が可能となり、メンテナンスも容易に行えます。

Seven Solutions 社が開発した最新の産業用製品は、管理、コンフィギュレーション、監視用の標準インターフェイスを搭載しています。これらの製品は、拡張された機能、サポート、ドキュメントとともに、White Rabbit テクノロジのすべての利点を活用できます。

## White Rabbit アプリケーション

White Rabbit テクノロジの最初の利用目的は、科学技術アプリケーションでした。最近では、高エネルギー物理学と電波天文学の分散型施設の枠組みの中で、このテクノロジは複数の施設と研究プロジェクトに統合し利用されています。White Rabbit は既に (CERN および GSI などの研究機関の) 複数の粒子加速器に採用され、KM3NeT、HISCORE など各種の国際的な科学研究イニシアチブにも採用が検討されています。このように、WR の手法は、高精度なタイミングと周波数の転送が重視される大規模施設の分散型インスツルメンテーションのように、非常に要求の厳しいアプリケーションで検証されています。

2014 年には、White Rabbit の長距離リンク試験が 2 箇所で実施されました。オランダでは VSL が 125km のテストを行い、フィンランドでは MIKES が 1,000km のテストを行っています。

高精度なタイミングは、科学技術分野を超えた幅広いアプリケーションで要求されています。スマート電力グリッドには信頼性の高い正確なタイミングが必須であり、高頻度トレーディングも認証された正確なタイミングに依存しています。

これらのアプリケーション分野の多くは、現在のところ GPS タイミング信号を利用していますが、この方式は基本的には ( 環境条件や、偶発的または悪意のある電波妨害となりすましの ) 脆弱性を抱えています。安全性が重視されるインフラストラクチャには、GPS を使用するべきではありません (「U.S. Air Force Chief Warns against Over-Reliance on GPS」、『Inside GNSS News』 2010 年 1 月 20 日 )。この意味で、White Rabbit は、地上設置型の光ファイバーを介して高精度なタイミングと周波数転送を可能にする代替手段となります。標準的な通信ネットワークを利用できるため、この手法は費用対効果に優れています。

Seven Solutions 社は、高可用性が要求される重要なアプリケーションをターゲットとする、White Rabbit の基本機能を超えた新しい産業用製品を開発しています。冗長電源、ホットスワップ対応ファン、ホールドオーバー オシレーターなどの技術により、システム障害や長時間の修理が許されない施設への White Rabbit 製品の導入が可能となります。

図 4 は、WR-LEN を分散型のメカニズムとして使用し、費用対効果の高い簡単な方法でタイミング情報を供給する方法を示しています。このシステムは GPS によく似た

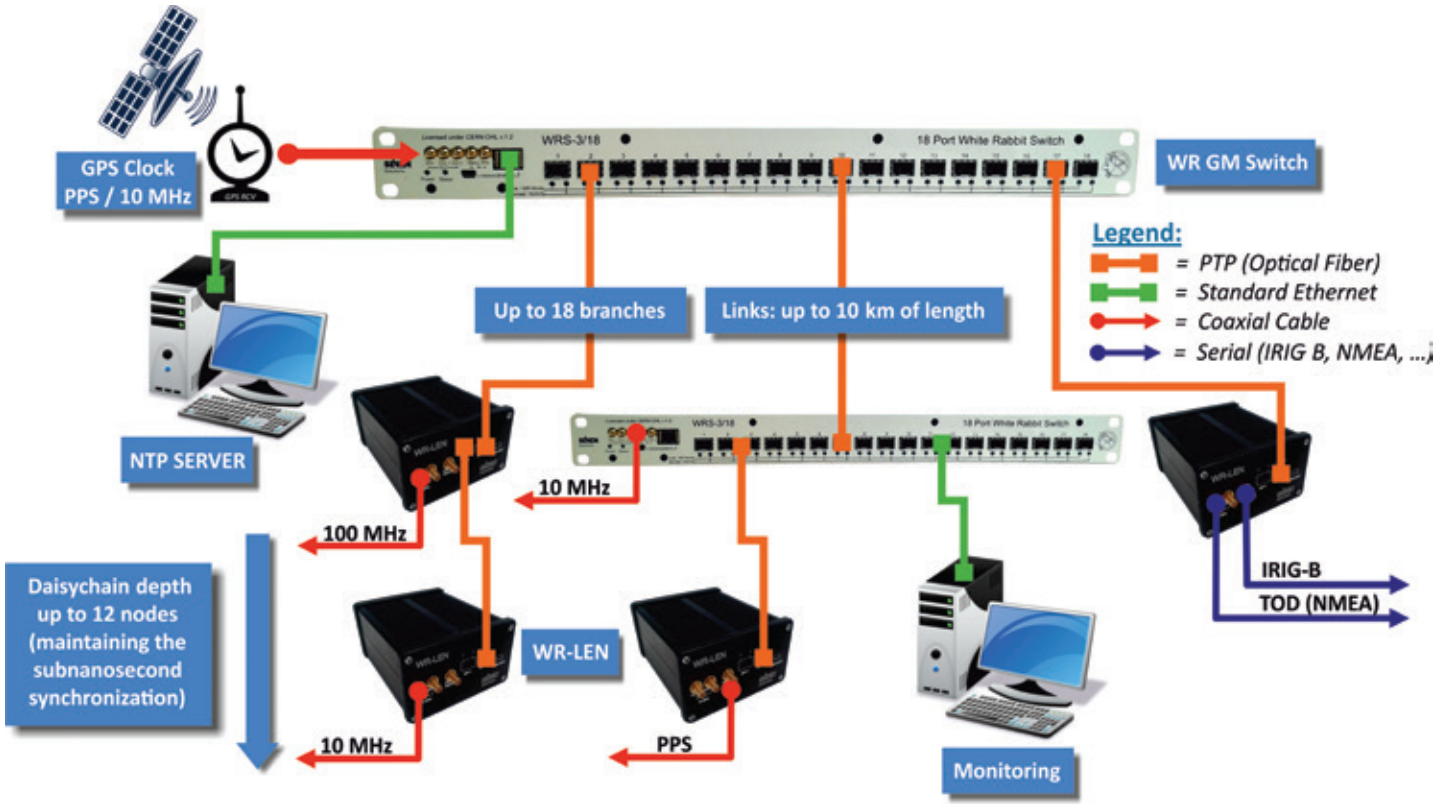



図 4 – 各種のノードに正確なタイミングを供給する White Rabbit ネットワーク。
WR-LEN ノードはデイジーチェーン構成で使用できます。

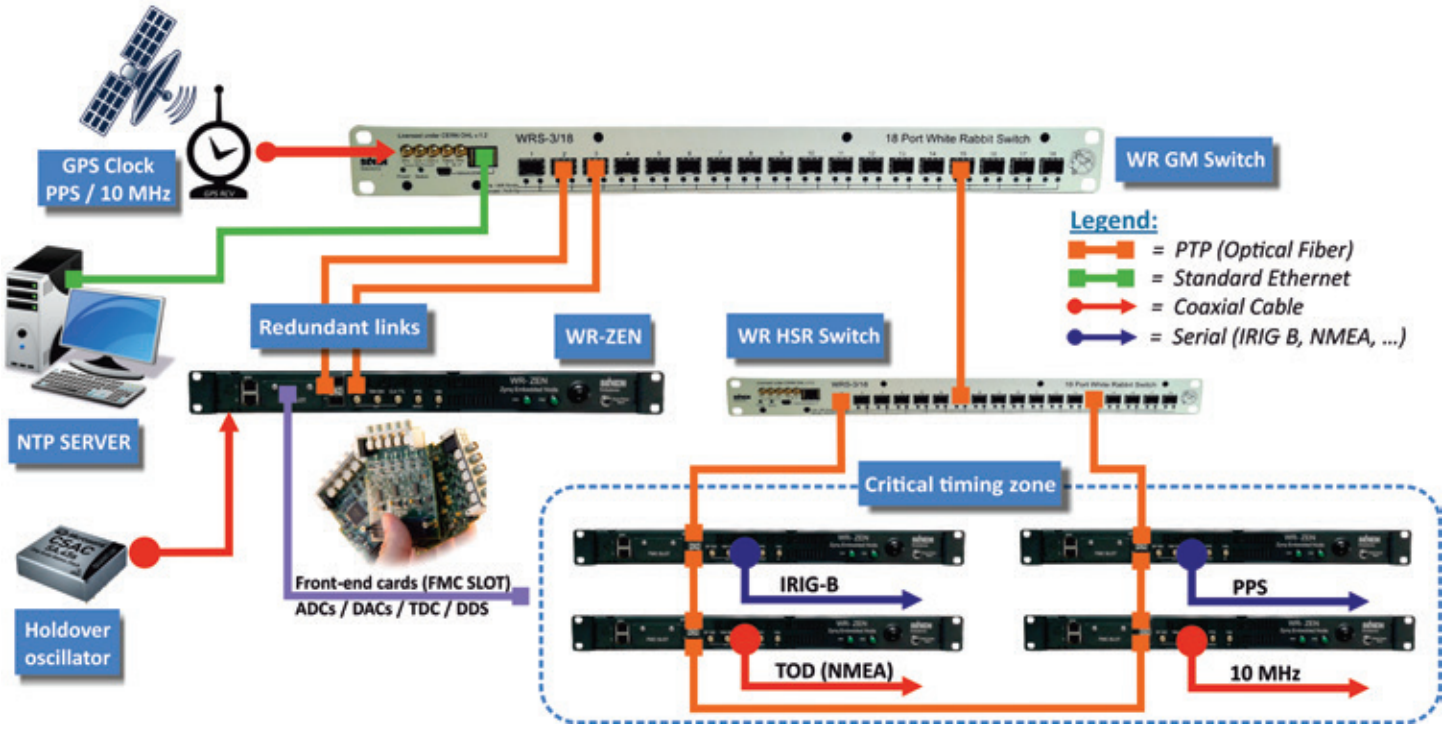



図 5 – ZEN 時刻供給機器をベースとする、安全性が重視されるシステムのネットワーク コンフィギュレーション

方法でタイミングを分配できます。このシステムは電力グリッド アプリケーションに適合する IRIG-B 出力フォーマットでタイミングを供給します。また、PTPv2 インターフェイスも利用可能です。PTPv2 インターフェイスは、今日の電力グリッド施設で使用されている PTP ネットワークと統合できるため、有効な選択肢になります。

図 5 に、ZEN ボードを利用した時刻供給機器をベースとするネットワーク コンフィギュレーションを示します。この場合は、LEN ボードよりもはるかに強力な機能を利用できます。冗長電源、冗長ネットワークトポロジ、ホールドオーバー CSAC オシレーターを重要な要素として採用することが可能です。さらに、FMC 拡張ポートにメザニン カードを装着して、追加の検知および制御アプリケーションを開発できます。

## 将来の展開

White Rabbit は、スマート グリッド、通信、高頻度トレーディングなど、さまざまな最終アプリケーション分野で入力情報の同期化の要件を解決できる有望なテクノロジです。WR は、位相の同期化のような問題を解決すると同時に、長距離 ( 数百 km) に分散配置された数万台のデバイスのクロックをサブナノ秒の精度で同期させることができます。その結果、White Rabbit は、超高精度の時刻転送と、ペナルティのない完全なデータ転送を同時に実現します。

これらの機能と、White Rabbit の優れた拡張性は、世界規模の地上設置型同期化メカニズムの開発を可能にします。このシステムは ( 基地局アンテナに基づく ) GPS ソリューションのバックエンド テクノロジとして使用でき、自動運転車や室内ナビゲーションなどの新しいアプリケーションの可能性を開きます。今後展開される 100G 通信ネットワークには、正確なサービス品質評価のためのメカニズムが役に立ちそうです。また 5G ワイヤレス技術は、よく知られた位相同期の問題の解決に WR を利用できます。

これらは、White Rabbit が将来大きな影響を与えそうなアプリケーションのほんの一部にすぎません。さらに、White Rabbit が IEEE-1588v3 プロファイル内で標準化されると ( 現在は検討中の段階 )、多くのベンダーが WR を容易に利用できるようになります。筆者らは、最も挑戦的なアプリケーションはこれから現れると考えています。

Seven Solutions 社は、White Rabbit テクノロジをベースとする産業用製品を初めて提供した企業です。Seven Solutions 社のターンキー ソリューションは、標準的な通信インターフェイスに基づいてユーザー アプリケーションに容易に統合でき、Web サービスや SNMP などの標準ソフトウェアだけを使ってコンフィギュレーション / 読み出しが可能です。今後 Seven Solutions 社は、RF 生成を分配する機能 ( リファレンス周波数の送信が不要 ) や、多くの通信アプリケーションに必要とされる高い精度でイベント収集をトリガーする機能を、製品に組み込んでいく予定です。

# Evaluating the Linearity of RF-DAC Multiband Transmitters

# RF-DAC マルチバンドトランスミッターの線形性の評価

ベル研究所の研究者が、ザイリンクスの FPGA、コア、MATLAB を使って RF DAC を迅速に評価できる柔軟なプラットフォームの作成方法を説明します。

**Lei Guan**
Member of Technical Staff
Bell Laboratories, Alcatel Lucent Ireland
*lei.guan@alcatel-lucent.com*

現在、無線通信業界は新しいオールインワン時代に突入し、すべてのネットワーク事業者がさらにコンパクトなマルチバンド インフラストラクチャ ソリューションを追求しています。新しい RF クラス データ コンバーター ( すなわち、RF DAC および RF ADC) のアーキテクチャは、コンパクトなマルチバンド トランシーバーの作成を可能にします。しかし、これらの新しいデバイス本来の非線形性がその障害になる可能性があります。

たとえば、RF デバイスの非線形性には、周波数ドメインでインバンドとアウトオブバンドの 2 つの面があります。インバンドの非線形性は TX バンド内の不要な周波数項であり、アウトオブバンドの非線形性は TX バンド外の望ましくない周波数項です。

システム技術者が RF DAC を使用してマルチバンド トランスミッターを試作する場合、RF DAC コンポーネントが各標準の線形性の要件を満たすようにすることが重要です。したがって、早期プロトタイプの段階から、マルチバンド アプリケーションにおける RF DAC の非線形性能を適切に評価できる柔軟なテスト プラットフォームが基本的に必要になります。

筆者らは、アイルランドのベル研究所において、次世代ワイヤレス システムの候補である RF DAC を迅速に評価できる柔軟なソフトウェア / ハードウェア プラットフォームを開発しました。この研究開発プロジェクトの 3 つの主要な要素は、高性能なザイリンクス FPGA、ザイリンクスの IP ( 知的設計資産 ) コア、MATLAB® です。

このエンジニアリング プロセスを詳しく説明する前に、簡単に概要を述べておきます。このデザインで筆者らは、システムの柔軟性をできるだけ維持しながら FPGA リソースの使用率を最小限に抑えるよう試み、必要な機能のインプリメントのみに関心を絞りました。テスト システム全体を準備するにあたって、筆者らは最新の Analog Devices 社製 RF-DAC 評価ボード (AD9129 および AD9739a) とザイリンクス ML605 評価ボードを選びました。ML605 ボードには Virtex®-6 XC6VLX240T-1FFG1156 FPGA デバイスが付属しており、この FPGA は RF DAC とのインターフェイス用の高速スイッチング I/O ( 最大 710MHz) と SERDES ユニット ( 最大 5Gbps) を搭載しています。

それでは、ザイリンクスの FPGA、IP コア、MATLAB を使ってこのシンプルで高性能なテスト プラットフォームの作成方法について詳しく説明しましょう。

## システム レベルの要件とデザイン

この評価プラットフォームの主な目的は、ユーザーがカスタマイズしたさまざまなテスト データ シーケンスを使って RF DAC をスティミュレートすることです。この目的のために、筆者らは、連続波 (CW) 信号テスト (xDDS) とワイドバンド信号テスト (xRAM) の 2 つのテスト手法を設計しました。

RF コンポーネントの非線形性の特性評価には、従来はマルチトーン CW テストが一般的に使用されてきました。筆者らは、同じテスト方針を維持して、ダイレクト デジタル シンセサイザ (DDS) をベースとするチューニング可能な 4 トーン ロジック コアを作成しました。このコアは、2 トーン信号の 1 つのペアを使用して、2 つの別々の周波数帯で RF DAC をスティミュレートします。互いに独立した 4 つのトーンをチューニングすることで、RF DAC の線形性性能、すなわち周波数ドメイン内の相互変調スパーズ (spurs) の位置とパワーを評価できます。

CW 信号テストは、基本的にはナローバンドのテストです。RF DAC のワイドバンド性能をさらに評価するには、デュアルモード UMTS (2.1GHz)/LTE (2.6GHz) 信号のような同時マルチバンド / マルチモード信号で RF DAC を駆動する必要があります。この目的のために、筆者らは、繰り返しテストのためにデュアルバンド ユーザー データが格納される 2 つの サブグループを持つ、オンチップ BRAM アレイ ベースのデータ ストレージ コアを作成しました。

図 1 に、システム レベルのデザインの簡略図を示します。おわかりのように、ここではわかりやすいデザイン手法を採用し、できるだけシンプルにプラットフォームを構築して、アップグレード機能を備えたモジュールにしています。

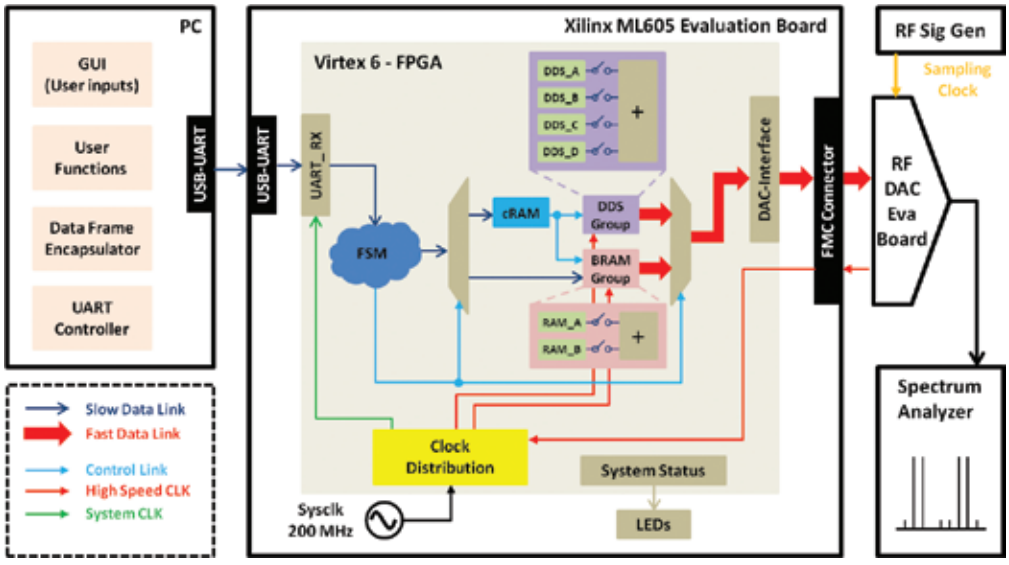



図 1 – システム レベルのプラットフォームの簡略ブロック図

## ハードウェア デザイン : ザイリンクスの FPGA コア

図 1 の FPGA 部分は、システムに根本的に必要とされる、インプリメントされたロジック ユニットの概要を示しています。これらのユニットは、クロック分配ユニット、ステート マシン ベースのシステム制御ユニット、DDS コア ベースのマルチトーン生成ユニットと、ブロック RAM 上に構築される 2 つのユニット ( 小型 BRAM ベースの制御メッセージ格納ユニットである cRAM コアと、BRAM アレイ ベースのユーザー データ格納ユニットである dRAM コア ) で構成されます。また、PC に対する UART シリアル インターフェイスと、RF DAC に対する高速データ インターフェイスも実装されます。

クロックは FPGA の脈拍です。筆者らは、複数のクロックが FPGA バンクに適切に分配されるように、対話形式で容易にクロックの定義と指定を実行できる、ザイリンクスのクロック管理コアを選びました。

ステート マシン上に構築されるコンパクトな命令コアは、システム制御ユニットとして機能します。図 2 に示すように、初期ステート (S0) ではヘッダー検出ユニットがアクティブになり、UART レシーバーから入力されるデータ バイトの監視とフィルタリングを実行します。データ バイトは、図 3 に示すように MATLAB 内で生成され、フレームにカプセル化されました。

基本的に、このシステム内には 2 種類のデータ フレームがあります。ヘッダーが “FF01” のフレーム (cRAM フレーム ) は、DDS への位相増分値とシステム制御メッセージの送信に使用されます。他のフレーム、ヘッダーが “FF10” または “FF11” のフレーム (dRAM フレーム ) は、ユーザーがカスタマイズしたデータの転送に使用されます。ステート “S1x” は、位相増分値の更新と制御命令の実行のために、ヘッダーが “FF01” のデータのみを処理します。ステート “S2x” および “S3x” は、それぞれユーザーがカスタマイズした 2 つのバンドのデータの受信と格納に使用されます。ビジー信号を使用して、データ シーケンスの最後に最終ストップ ビットが検出されるまで、データを継続的にラッチします。たとえば、単一 / 複数の DDS またはユーザー データ シーケンスを起動する制御メッセージは、cRAM フレームの最後の 2 バイトに格納されます。これらの命令は、cRAM_rd_done 信号の立ち上がりエッジで実行されます。

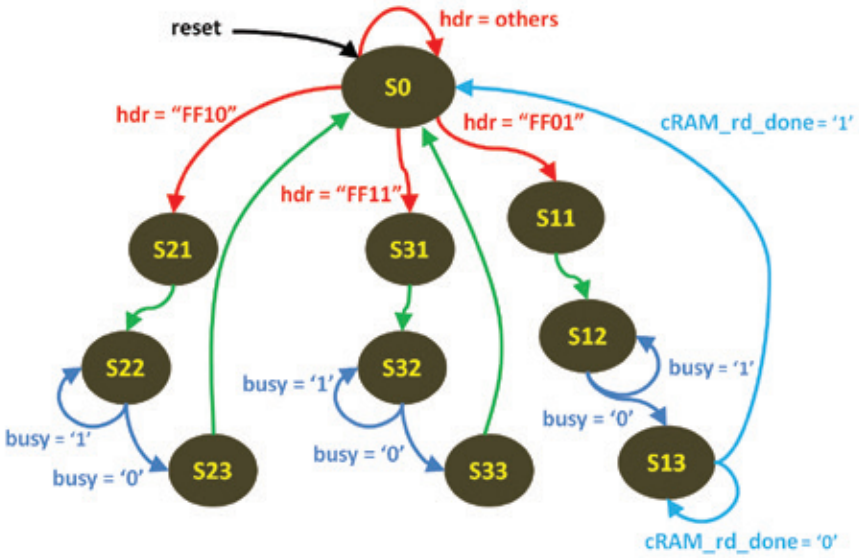



図 2 – 主要なステート マシンの詳細なデザイン チャート

次に筆者らは、位相増分モードにコンフィギュレーションしたザイリンクス DDS コアを使用して、互いに独立した 4 つのトーン生成ユニットをインスタンシエートしました。特定の周波数の位相増分値は MATLAB 内で生成され、cRAM フレームを介して FPGA にダウンロードされました。

筆者らは加算器によって複数のトーンを結合し、これらのトーンを次の段にパイプライン化しました。DDS コアの出力は 2 の補数バイナリ フォーマットであるため、RF DAC がオフセット バイナリなどの他のデータ フォーマットを要求している場合は、フォーマット変換ユニットが必要になります。

一般的に、小規模から中規模のユーザー ストレージを作成する場合、高性能なオンチップ BRAM が常に第 1 の選択肢となります。たとえば、このプラットフォームでは、ザイリンクスのブロック メモリ ジェネレーター コアを使用して、2 つの周波数帯のために互いに独立した 2 つのデータ RAM を構築しました。各データ RAM は幅 16 ビット、深さ 192k です。

PC と FPGA 間の通信用には、UART シリアル インターフェイス ユニットを作成し、比較的低速の 921.6kbps (115.2 キロバイト / 秒に相当 ) にコンフィギュレーションしました。このインターフェイスは、cRAM フレーム (18 バイト ) の転送に約 0.16 ミリ秒、dRAM フレーム ( 約 384 キロバイト ) の転送に約 3.33 秒かかります。

デバイス メーカーは通常、自社製チップへの高速データ インターフェイスのサンプル デザインを VHDL または Verilog フォーマットで提供しています。経験豊富な FPGA 技術者であれば、このリファレンス デザインを再利用またはカスタマイズすることはそれほど難しくありません。たとえば、このシステムの AD9739a および AD9129 RF DAC の場合、Analog Devices 社からパラレル LVDS インターフェイスのリファレンス デザインが提供されています。また、チップ ベンダーのサンプル デザインが利用できない場合のために、ザイリンクスは非常に便利な高速インターフェイス用 IP コアをいくつか用意しています (CPRI、JESD204B など )。

## ソフトウェア デザイン : MATLAB の DSP 機能と GUI

MATLAB はデジタル信号処理 (DSP) 機能の面で多くの利点を備えているため、筆者らはソフトウェア ホストとして MATLAB を選びました。しかも MATLAB は、グラフィカル ユーザー インターフェイス (GUI) をレイアウトできる、GUIDE と呼ばれる便利なツールを備えています。それでは、このプロジェクトでは MATLAB のどのような機能が必要なのでしょうか。

実際に必要なのは、低レベルの DSP 機能およびデータ フロー制御機能に関連する

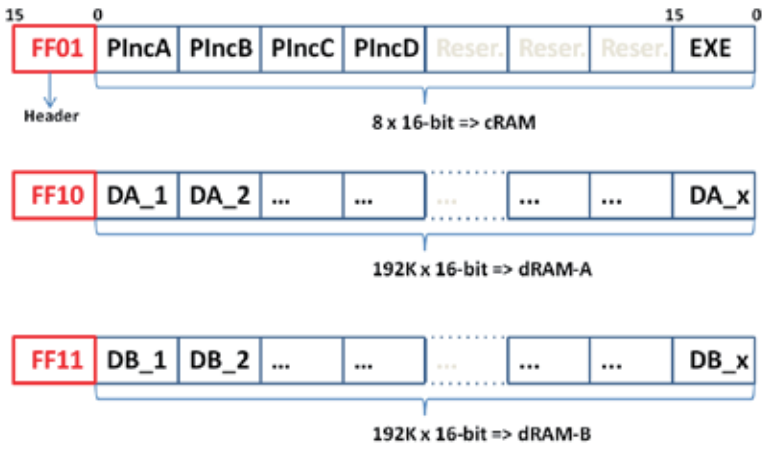



図 3 – データ フレームのカプセル化の図解

ユーザー インターフェイスです。必要な DSP 機能は、位相増分値カリキュレーター、ベースバンド データ シーケンス ジェネレーター、デジタル アップコンバーターです。制御機能は、データ フレーム エンキャプシュレーター、UART インターフェイス コントローラー、システム ステータス インジケーターです。

図 4 に、筆者らがこのプラットフォーム用に作成した GUI を示します。まず RF DAC の主要なパラメーター ( すなわち、サンプリング レート ) を定義し、次に xDDS モードまたは xRAM モードのどちらでデバイスをスティミュレートするかを選択します。次に、各サブパネルで、それに対応する MATLAB 信号処理機能を起動するためのパラメーターをカスタマイズできます。xDDS モードでは、サンプリング レートが fs の周波数トーン fc の位相増分値は、phase_incr = fc*2$^{nbits}$/fs という簡単な式で計算できます ( ここで、nbits は DDS が周波数の合成に使用するバイナリ ビット数です )。“Start” ボタンをクリックすると、生成された位相増分値が固定小数点フォーマットに変換され、図 3 に示すようにさまざまなヘッダーおよび制御メッセージを含む 2 バイト データ フレームにカプセル化され、次に UART を介して cRAM ユニットに送信され、FPGA 内で実行されます。

xRAM モードでは、MATLAB 内でベースバンド データ シーケンスを生成し、フルスケール ( 符号付き 16 ビット ) に正規化し、必要な周波数にアップコンバートしました。処理済みのデータを UART を介して dRAM にダウンロードした後、Start ボタンをクリックすると、ワイドバンド信号テストを起動できます。FPGA 側と同じプロトコル パラメーターを使用して、MATLAB 内で UART シリアル インターフェイスをコンフィギュレーションすることを忘れないでください。

最後に筆者らは、信号ジェネレーター R&S SMU200A を使用して、RF DAC を論理的に「オンにする」ためのサンプリング クロックを提供しました。そして、周波数ドメインでの RF DAC の線形性性能を評価するために、RF DAC の出力をスペクトラム アナライザーに接続しました。

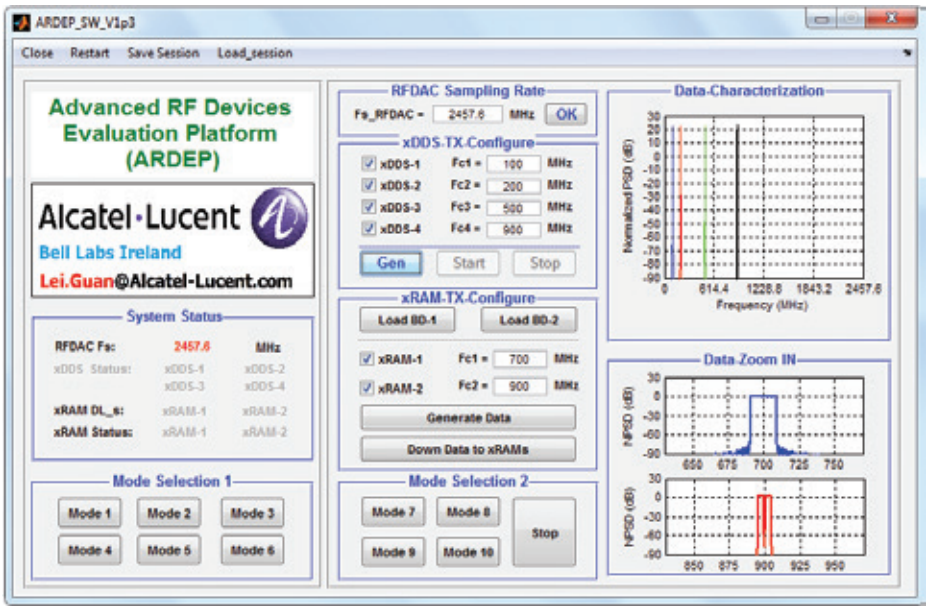



図 4 – グラフィカル ユーザー インターフェイスのスナップショット

## 迅速な評価

早期プロトタイピングの段階で、主要な RF コンポーネントの線形性性能の評価が重要な問題になることがあります。筆者らのハードウェア / ソフトウェア プラットフォームを使えば、性能面で妥協することなく、この評価を迅速に管理できます。ここで RF パワー アンプを追加し、ここで提案したプラットフォームを使って、カスケード接続されたシステムの線形性を評価できます。非線形性が特定できたら、何らかのデジタル プリディストーション アルゴリズムをインプリメントし、カスケード接続されたシステムから不要な非線形性を除去できます。

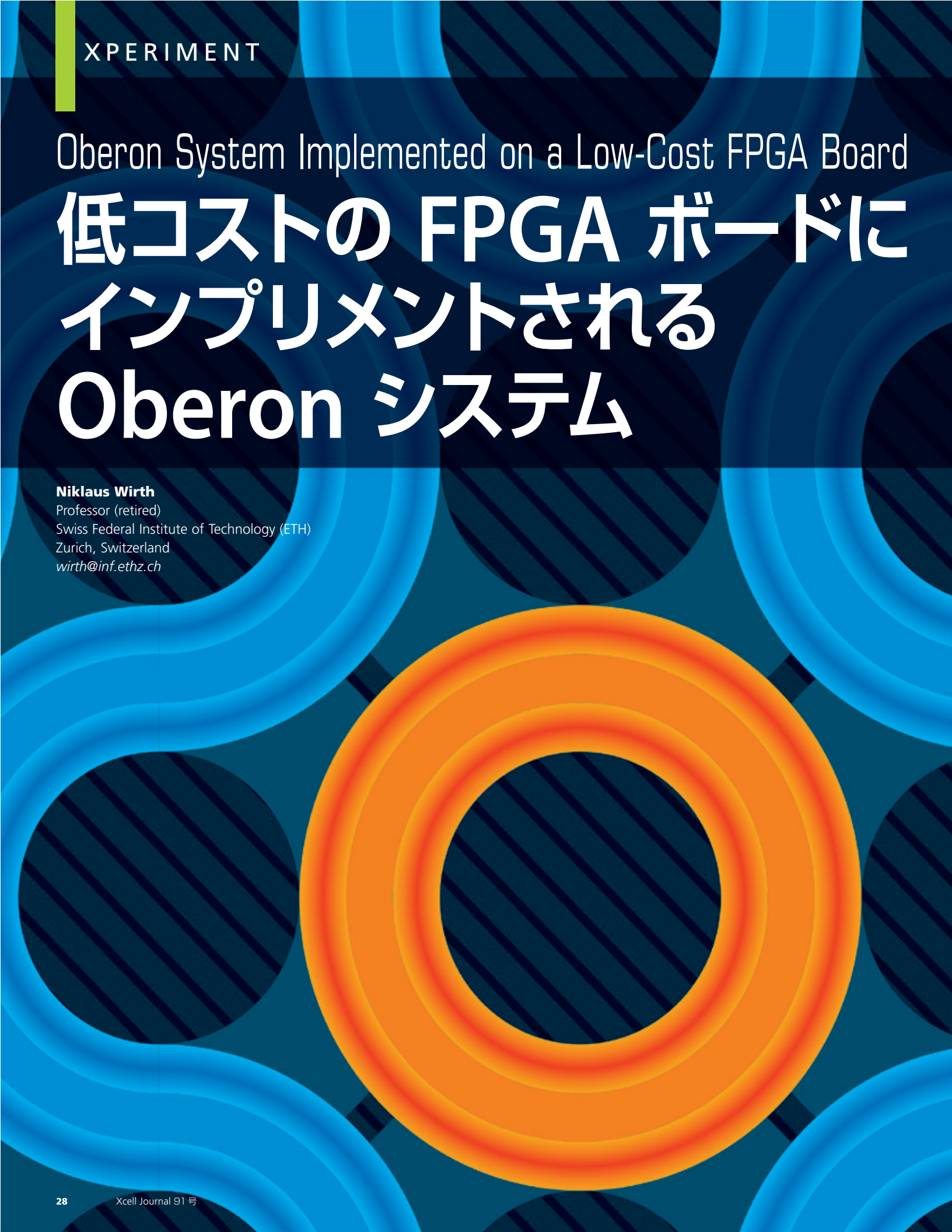
FPGA デザインにザイリンクスの IP コアを適切に使用すれば、開発サイクルが大幅に短縮され、デジタル システムの堅牢性が向上します。今後、筆者らはこのプラットフォームのデータ インターフェイス モジュールを JESD204B 規格にアップグレードし、同期化された複数の同期 RF DAC に必要な高速データ レートをサポートする予定です。それと同時に、筆者らは FPGA ホストをザイリンクスの ML605 から Zynq®-7000 All Programmable SoC ZC706 評価キットへと移行しています。外部の DSP や個別 PC 上の制御機能を必要としないスタンドアロン ソリューションの作成には、Zynq SoC デザインが良い選択肢になるでしょう。

このプラットフォームとデジタル プリディストーションの詳細は、筆者 (lei.guan@alcatel-lucent.com) までお問い合わせください。

# Oberon System Implemented on a Low-Cost FPGA Board

# 低コストの FPGA ボードにインプリメントされる Oberon システム

**Niklaus Wirth**
Professor (retired)
Swiss Federal Institute of Technology (ETH)
Zurich, Switzerland
*wirth@inf.ethz.ch*



**ザイリンクスの Spartan-3 ボードは、ソフトウェア教育用の筆者の Oberon プログラミング言語およびコンパイラを改良するための基盤となります。**

1988 年、ユルグ グートクネヒト (Jürg Gutknecht) 氏と筆者は、筆者が以前に開発した Pascal と Modula-2 という 2 つの言語の後継として、Oberon プログラミング言語 [1,2] を完成して公開しました。筆者らは当初、コンピュータ サイエンス専攻の学生のシステム プログラミング教育を支援する目的で、Modula-2 よりも合理化された効率的な言語として Oberon 言語を設計しました。1990 年には、筆者らはこの試みをさらに先に進めて、ウィンドウとワード プロセッシング機能を備えたワークステーション用の現代的な OS として、Oberon オペレーティング システムを開発しました。次に筆者らは、Oberon コンパイラと Oberon オペレーティング システムに関する詳しい解説書を発表しました。この解説書 (『Project Oberon』) には、詳しいインストラクション (命令) の説明とソース コードが記載されています。

数年前、筆者の友人であるポール リード (Paul Reed) 氏は、この解説書はシステム デザインの教育上高い価値があり、将来のイノベーターたちが信頼性の高いシステムをゼロから設計する際の良い出発点になるという理由で、この解説書の改訂と再版を筆者に勧めました。

しかし、大きな障害が 1 つありました。筆者が最初に開発したコンパイラは、現在では入手困難なプロセッサをターゲットとするものでした。したがって、筆者の解決策は、現在のプロセッサ向けにコンパイラを書き換えることでした。しかし、かなり手間をかけて探した結果、筆者の明解性、規則性、簡潔性の基準を満たすプロセッサは見つかりませんでした。このため、筆者は自分でプロセッサを設計することにしました。現在の FPGA はシステム ソフトウェアとハードウェアを同じように設計できるので、筆者はこのアイデアを実現できました。さらに、ザイリンクスの FPGA を選んだおかげで、1990 年のオリジナル バージョンのデザインをできるだけ維持しながら、システムを更新できました。

Xcell Journal は、伝説的な研究者であるニクラウス ヴィルト (Niklaus Wirth) 教授が執筆した記事を本号に掲載できたことを光栄に思います。ヴィルト教授は、Pascal 言語およびいくつかの派生プログラミング言語を発明し、コンピュータおよびソフトウェア エンジニアリングのいくつかの古典的な手法を開拓した方です。ACM チューリング賞と IEEE コンピュータ パイオニア賞を受賞したヴィルト教授は、既に教壇を退かれていますが、明日のイノベーターの育成に取り組む教育者への支援を続けています。

新しいプロセッサは RISC と名付けられ、1 メガバイトのスタティック RAM (SRAM) メモリを搭載した低コストの Digilent Spartan®-3 開発ボードに実装されました。筆者が追加したシステム ハードウェアは、旧システムのハードディスク ドライブを置き換える SD カードおよびマウス用インターフェイスのみです。

この解説書とシステム全体のソース コードは、projectoberon.com [3,4,5] から入手できます。このサイトから、S3RISCinstall.zip という名前の一つのファイルも入手できます。このファイルには、インストラクション ( 命令 ) の説明、SD カードのファイルシステム イメージ、(Spartan-3 ボードのプラットフォーム フラッシュ用 PROM ファイル形式の ) FPGA コンフィギュレーション ビット ファイルと、SD カード / マウス用インターフェイス ハードウェアの構築方法の詳細が含まれています。

## RISC プロセッサ

このプロセッサは、論理演算ユニット (ALU)、16 個の 32 ビット レジスタの配列、( 命令レジスタ (IR) とプログラム カウンタ (PC) を備えた ) 制御ユニットで構成され、Verilog モジュール RISC5 によって表現されます。

このプロセッサは 20 個の命令を搭載しています。その内訳は、4 つの移動 / シフト / ローテート命令、4 つの論理演算命令、4 つの整数演算命令、4 つの浮動小数点演算命令、2 つのメモリ アクセス命令、2 つの分岐命令です。

RISC5 は RISC5Top によってインポートされます。RISC5Top は、各種の ( メモリ マップド ) デバイスおよび SRAM (256M × 32 ビット ) へのインターフェイスを含む環境です。システム全体 ( 図 1) は、次の Verilog モジュールで構成されます ( 次に行数を示す )。

筆者は、1024 × 768 × 1 ビット パーピクセル = 98,304 バイト ( 基本的には利用可能な 1 メガバイトのメイン メモリの 10%) を占めるように、白黒 VGA ディスプレイをメモリ マップしました。SD カードは元のシステムの 80 メガバイトを置き換えます。SD カードには、バイト単位または 32 ビット ワード単位でデータを受け入れてシリアル化する、標準 SPI インターフェイスを介してアクセスします。キーボードとマウスは、標準のシリアル PS-2 インターフェイスで接続されます。さらに、非同期シリアル RS-232 回線および汎用 8 ビット パラレル I/O

| | | |
|---|---|---|
| RISC5Top | environment | 194 |
| RISC5 | processor | 201 |
| Multiplier | integer arithmetic | 47 |
| Divider | | 24 |
| FPAdder | floating-point arithmetic | 98 |
| FPMultiplier | | 33 |
| FPDivider | | 35 |
| SPI | SD card and transmitter/receiver | 25 |
| VID | 1024 x 768 video controller | 73 |
| PS2 | keyboard | 25 |
| Mouse | mouse | 95 |
| RS232T | RS232 transmitter | 23 |
| RS232R | RS232 receiver | 25 |

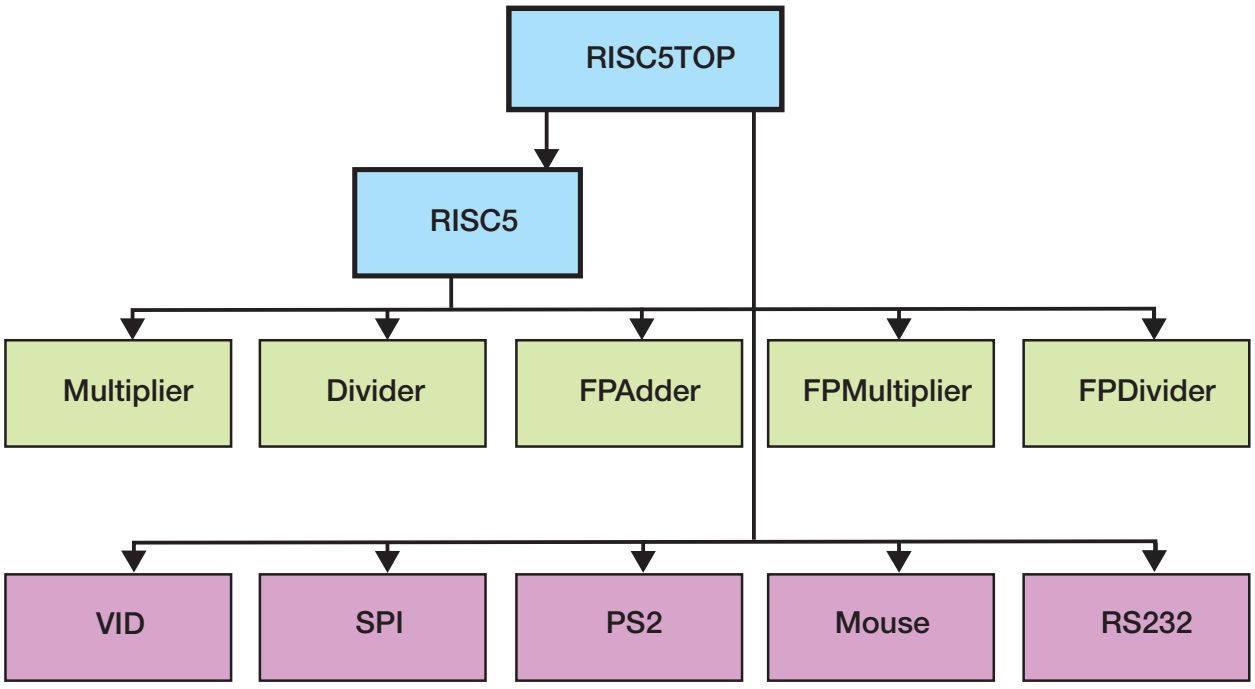



図 1 – システムとシステムに含まれる Verilog モジュールの図

インターフェイスが存在します。モジュール RISC5Top は、1 ミリ秒ごとにインクリメントするカウンタも備えています。

## Oberon オペレーティング システム

このオペレーティング システム ソフトウェアは、ガベージ コレクター付きメモリ アロケータおよびファイル システムを含むコアと、ローダー、テキスト システム、ビューア システム、テキスト エディターで構成されます。

「Oberon」と呼ばれるモジュールは中央のタスク ディスパッチャーであり、「System」は基本コマンド モジュールです。動作を起動するには、ディスプレイ上の任意のビューア内でテキスト「M.P」の真ん中のボタンをクリックします (P はモジュール M 内で宣言されるプロシージャの名前 )。M が存在しない場合は、そのモジュールが自動的にロードされます。ただし、ほとんどのテキスト編集コマンドは、単にマウス クリックで起動されます。左ボタンをクリックすると、テキストの位置をマークするカレットが設定されます。右ボタンをクリックすると、テキスト ストレッチを選択できます。

モジュール「Kernel」には、ディスク格納管理とガベージ コレクターが含まれます。ビューアはタイル表示されて重ならないようにしました。標準レイアウトでは、2 つの垂直方向のトラックと任意の数のビューアが表示されます。単にビューアのタイトル バーをドラッグするだけで、ビューアを拡大、縮小、移動できます。図 2 に、モニター上で動作するユーザー インターフェイスと、Spartan-3 ボード、キーボード、マウスを示します。

このシステムは、ロードされると、モジュール空間内の 112,640 バイト (21%) およびヒープの 16,128 バイト (3%) を占有します。図 3 に示すように、このシステムは次のモジュールで構成されます ( 次に行数を示す )。

| | | |
|---|---|---|
| Kernel | 271 | (inner core) |
| FileDir | 352 | |
| Files | 505 | |
| Modules (loader) | 226 | |
| Viewers | 216 | (outer core) |
| Texts | 532 | |
| Oberon | 411 | |
| MenuViewers | 208 | |
| TextFrames | 874 | |
| System | 420 | |
| Edit | 233 | |

電源投入時またはリセット時のシステムの初期化が約 2 秒で完了することは注目に値します。この時間にはファイル ディレクトリのガベージ コレクション スキャンが含まれます。

## Oberon コンパイラ

Oberon オペレーティング システム上でホスティングされる Oberon コンパイラは、シンプルなトップダウンの再帰下降構文解析手法を使用します。このコンパイラは、モジュールの選択したソース テキスト上でコマンド ORP.Compile @ を使って起動できます。パーサーは、識別子、数字、特殊記号 (BEGIN、END、+ など ) を提供するスキャナーから記号を入力します。この方法が便利で洗練されていることは、多くのアプリケーションで実証されています。この手法の詳細は、筆者の著書『Compiler Construction』[6,7] で説明しています。

パーサーは、コード ジェネレーター モジュール内でプロシージャを呼び出します。プロシージャはコード配列内に命令を直接付加します。すべての分岐宛先がわかっている場合、前方分岐命令には、モジュールのコンパイルの最後 ( フィックスアップ ) にジャンプ アドレスが与えられます。

すべての可変アドレスは、ベース レジスタを基準とします。( 実行時にプロシージャの入口で設定される ) ローカル変数のベース レジスタは R14 ( スタック ポインター ) であり、グローバル変数およびインポートされた変数のベース レジスタは R13 です。ベース アドレスはシステム グローバル モジュール テーブルからオンデマンドでロードされます。このテーブルのアドレスはレジスタ R12 に保持されます。R15 は、RISC アーキテクチャによって決定されるリターン アドレス ( リンク ) に使用されます。したがって、R0 ～ R11 は式の評価とプロシージャ パラメーターの受け渡しに利用可能です。

コンパイラ全体は、比較的小さい効率的な 4 つのモジュールで構成されます ( 次に行数を示す )。

| | | |
|---|---|---|
| ORP | parser | 968 |
| ORG | code generator | 1120 |
| ORB | base def | 435 |
| ORS | scanner | 311 |

コンパイラは、( コンパイルの実行前に ) モジュール空間の 115,912 バイト (22%) およびヒープの 17,508 バイト (4%) を占有します。コンパイラのソース コードは約 65 キロバイト長です。コンパイラそれ自体のコンパイルは、25MHz RISC プロセッサ上ではわずか数秒で完了します [8]。

コンパイラは、値が NIL のポインターによって、配列インデックスのチェックと参照のチェックを常に生成します。違反があった

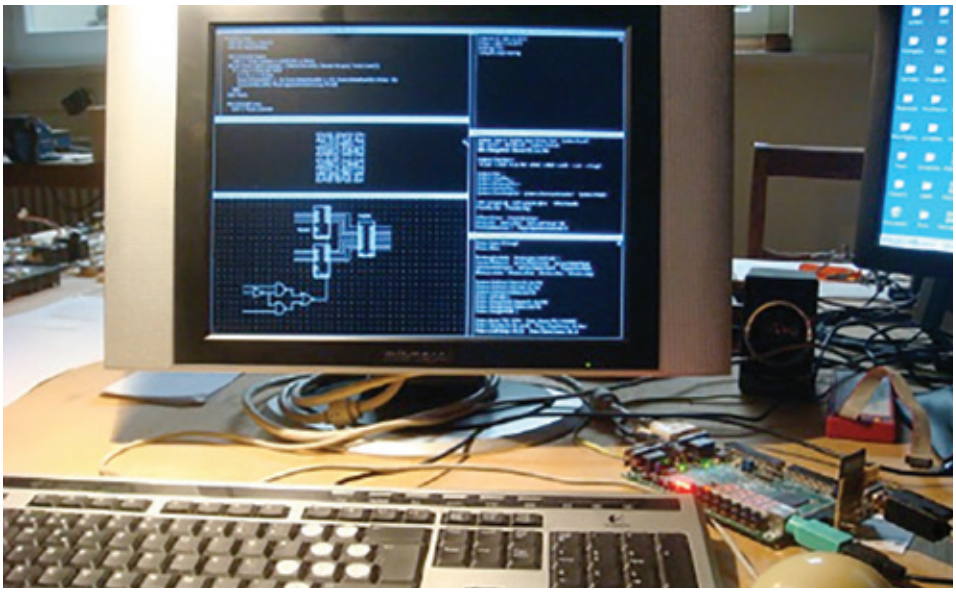


図 2 – ユーザー インターフェイスを表示するモニター、右側に Spartan-3 ボード

場合、コンパイラはトラップを発生させます。この方法で、エラーと破壊に対する高度なセキュリティを確保します。実際に、システム インテグリティの違反は、疑似モジュール SYSTEM 内の操作 ( すなわち、PUT および COPY) の使用によってのみ発生します。これらの操作は、デバイス インターフェイスにアクセスするドライバー モジュールに限定する必要があります。SYSTEM はこれらをインポート リスト内で容易に認識します。オペレーティング システム全体が Oberon 言語でプログラムされているので、アセンブラコードを使用する必要はありません。

筆者が Digilent Spartan-3 開発ボードを選択したのは、このボードが低コストかつシンプルであり、学校のクラスルームに教育用キットとして利用しやすいからです。重要な利点は、このボードがスタティック RAM を搭載しており、( バイト単位のデータ転送を選択した場合でも ) 簡単にインターフェイスがとれることです。残念なことに、より新しいボードはすべてダイナミック RAM を搭載しています。ダイナミック RAM は、より大容量ではありますが、リフレッシュおよび初期化 ( キャリブレーション ) 用の回路が必要になるため、インターフェイスがはるかに複雑になります。この回路だけで、スタティック RAM を搭載したプロセッサ全体と同じくらい複雑になることもあります。コントローラーがオンチップで提供されている場合でも、このような複雑さは、検査のためにすべての要素をオープンにするという筆者らの方針を損ないます。

# Lola HDL と、Lola から Verilog への変換

Lola と呼ばれるハードウェア記述言語 (HDL) は、ハードウェア デザインの基礎教育の手段として 1990 年に定義されました。当時は回路図に代わってテキスト形式の定義が普及し始めた時期であり、最初の FPGA が入手可能になっていましたが、まだ一般的なデザインには採用されていませんでした。Lola は、(FPGA にロードされるビット ファイルを生成する ) コンパイラによってインプリメントされました。ビット ファイルの形式は、Algotronix Inc. 社および Concurrent Logic Inc. 社によって公開されました。いずれの形式も、自動配置配線に最適と思われる、非常にシンプルな構造のセルを特徴としていました。

Oberon を FPGA 上に再インプリメントするプロジェクトに続いて、Lola を復活させる構想も浮上しました。ザイリンクスが独自規格であるという理由でビット ファイル形式の公開を拒んでいるという事実に加えて、ザイリンクスの FPGA のセルは非常に複雑な構造を採用しているため、筆者らは配置配線機能のインプリメントには踏み込まずにいました。

明らかな対策は、独自規格に基づくビット ファイルを生成するのではなく、ザイリンクスが合成ツールを提供している言語への変換を実行する、Lola コンパイラを作成することでした。筆者らは Verilog を選びました。この解決策は、まず Lola モジュールの解析、次にモジュールの変換、次に再解析という非常に大きな迂回を意味します。これらのすべての手順を経て、筆者らは、Lola コンパイラが適切なエラー レポート機能とタイプ整合性チェック機能を備えていることを確認しました。

Lola-2 の開発を促進するため、筆者らは RISC5 プロセッサのすべてのモジュールを Lola で再定義することを目指しました。現在ではこの目標は達成されました。

## Lola 言語

Lola は Oberon スタイルの小規模で簡潔な言語です(http://www.inf.ethz.ch/personal/wirth/Lola/Lola2.pdf を参照 )。

簡単に説明するために、ここでは Lola テキストの例を 1 つだけ示します ( 図 1)。ソース テキストの単位をモジュールと呼びます。

```
MODULE Counter0 (IN CLK50M, rstIn: BIT;
  IN swi: BYTE; OUT leds: BYTE);

TYPE IBUFG := MODULE (IN I: BIT; OUT O: BIT) ^;
VAR clk, tick0, tick1: BIT;
  clkInBuf: IBUFG;
REG (clk) rst: BIT;
  cnt0: [16] BIT;  (*half milliseconds*)
  cnt1: [10] BIT;  (*half seconds*)
  cnt2: BYTE;

BEGIN leds := swi.7 -> swi : swi.0 -> cnt1[9:2] : cnt2;
  tick0 := (cnt0 = 49999);
  tick1 := tick0 & (cnt1 = 499);
  rst := ~rstIn;
  cnt0 := ~rst -> 0 : tick0 -> 0 : cnt0 + 1;
  cnt1 := ~rst -> 0 : tick1 -> 0 : cnt1 + tick0;
  cnt2 := ~rst -> 0 : cnt2 + tick1;
  clkInBuf (CLK50M, clk)
END Counter0.
```

図 1 – ミリ秒および秒をカウントし、秒数をボードの LED 上に表示するカウンタを示す Lola テキスト

## まとめ

40 年以上前、C.A.R. ホーア (C.A.R. Hoare) 氏は、あらゆる科学技術課程の学生たちが、各自の試みを実験するように求められる前に、多くの模範的な設計例に触れるべきであるという意見を述べました。しかし、プログラミングとソフトウェア デザインは、このような賢明な方針とは全く対照的な状況にありました。これらの課程の学生たちは、プログラムの例を読む前に、最初からプログラムを書くように要求されていました。

このような悲惨な状況は、多くの模範的な例を記載した資料がほとんどないことが原因でした。そこで筆者は、この状況を多少なりとも改善しようと決意し、1975 年にアルゴリズムとデータ構造の解説書を執筆しました。それ以降、筆者は (J. グートクネヒトとともに )、オペレーティング システム課程の教育者としての職務の傍ら、システム Oberon を設計しました (1986 年 ～ 1988 年 )。

それ以来、プログラミング教育に明らかな改善が見られないまま、システムの規模と複雑性は急激に増大しました。オープン ソースの取り組みは歓迎すべきことですが、状況は基本的には変わっていません。なぜなら、ほとんどのプログラムはコンピュータ上で「走らせる」ために作られており、人が咀嚼して理解するようには作られていないからです。

筆者は、すべてのプログラムはコンピュータのためだけでなく、人が読むために設計する必要があると提唱し続けています。プログラムは公開可能でなければなりません。

モジュールの見出しは、モジュールの名前と、モジュールの入力パラメーターおよび出力パラメーターの名前およびタイプを指定します。見出しに続いて、変数やレジスタなどのローカル オブジェクトの宣言を含むセクションがあります。それに続いて、変数とレジスタの値を代入によって定義するセクションがあります。BYTE は 8 ビットの配列を示します。

## Lola コンパイラ

Lola コンパイラは、シンプルなトップダウンの再帰下降構文解析手法を使用します。このコンパイラは、選択した Lola ソース テキスト上でコマンド LSC.Compile @ によって起動されます。パーサーは、識別子、数字、特殊記号 (BEGIN、END、+ など ) を提供するスキャナーから記号を入力します。この方法が便利で洗練されていることは、多くのアプリケーションで実証されています。この手法については、『Compiler Construction』(Part 1 および Part 2) で説明しています。

Verilog テキストをオンザフライで直接生成する代わりに、パーサー内に文が存在し、解析の副次的作用として、さらなる処理に適した形で入力テキストを表現するツリーが生成されます。この構造には、さまざまなトランスレーターを呼び出すことによってさまざまな異なる出力を容易に生成できるという利点があります。そのうちの 1 つが Verilog へのトランスレーターです。コマンドは LSV. List outputfile.v です。VHDL に変換する他のコマンドや、ツリーをそのままリストアップする他のコマンドもあります。また、配置配線ツールによる処理のためにネットリストを生成するコマンドもあります。

このように、コンパイラ全体は、少なくとも 4 つの比較的小さい効率的なモジュールで構成されます ( 次に行数を示す )。

| | | |
|---|---|---|
| LSS | scanner | 159 |
| LSB | base | 52 |
| LSC | compiler/parser | 503 |
| LSV | Verilog generator | 215 |

Lola から Verilog への変換のインストラクション ( 命令 ) は、http://www.inf.ethz.ch/personal/wirth/Lola/LolaCompiler.pdf に記載されています。

## ソフトウェア プログラムとハードウェア プログラムの違い

これまでの多くの取り組みの結果、HDL は一見、通常のプログラミング言語 (PL) と同じように見えます。また、HDL は通常、プログラミング言語の集合の中にカウンターパートとなる言語があり、その言語のスタイルを採用していることにも注意してください。たとえば、Verilog の原型は C であり、VHDL の原型は Ada、Lola の原型は Oberon です。筆者らは、特に構文がよく似ているかまたは同一である場合、2 つのクラスの根本的な違いを認識することが重要であると考えています。両者の根本的な違いは何でしょうか。

簡単に説明するために、分析の対象を同期回路 ( すなわち、すべてのレジスタが同じクロックで動作する回路 ) に絞ります。一般的に、できる限り同期回路を使用することは、確かに健全な設計方針です。これにより、明らかに、回路のすべての要素は同時に ( 文字どおり同じ時刻に ) 動作します。各変数およびレジスタはただ 1 つの式によって定義され ( 組み合わせ回路 )、複数の代入は無効とされます。レジスタおよび変数への代入は 1 クロック サイクルごとに繰り返されるので、各 HDL プログラムが 1 つの永久に繰り返される大きな節に囲まれている状態を想像することができます。

ジョン フォン ノイマン (John von Neumann) は、独創的な発想により、シーケンサを備えたプロセッサ アーキテクチャを導入しました。シーケンサには命令レジスタが含まれており、このレジスタに従って 1 サイクルごとに特定の回路が選択され、他の回路は無視されるため、ALU のさまざまな部分が上手に再利用されます。これで、サイクルまたはステップは基本的にシーケンシャルになり、プログラム カウンタは変数をプログラム内および命令シーケンス内の特定の位置に関連付けるため、同じ変数への値の再代入が可能となります。このようなシーケンサのアイデアは、巨大なプログラムを比較的簡単な回路で実行することを可能にします。

これまでのことを要約すると、Lola-2 は、Oberon プログラミング言語のスタイルの HDL です。ここで取り上げたコンパイラは、Lola モジュールを Verilog モジュールに変換します。Lola の利点は、言語の構造がシンプルで規則的であることと、コンパイラがタイプ チェックを重視し、エラー診断機能が向上していることです。RISC プロセッサのモジュール セット全体が Lola で表現されています (http://www.inf.ethz.ch/personal/wirth/Lola/index.html を参照 )。

— ニクラウス ヴィルト (Niklaus Wirth)

これは実行可能プログラムを（しかも適切で効率的なプログラムを）作成することよりもはるかに大変な課題であり、この課題のいかなる部分もアセンブラ コードで記述してはならないことを意味します。

このような「人的要因」を無視した結果、多くの分野で新しいアプリケーションが注意深く設計されることなく、完成までデバッグが繰り返され、しばしば悲惨な結果をもたらしています。理解性の達成に重要な条件は、簡潔性と規則性を守ること、不要な飾りを捨てること、余計な機能の追加を避けること、単なる慣習と真の便利さを区別することです。

このシステムの小さなサイズは、どれほど少ないリソースでどれほど多くのことを達成できるかの証明です。Oberon システムは、ファイル システム、テキスト エディター、ビューア（ウィンドウ）管理機能を備えているにもかかわらず、現在のほとんどのオペレーティング システムと比べて途方もなく小さなサイズに抑えられています。その副次的効果は、システムが少数のシンプルな規則に基づいて動作するため、使用方法を容易に学習できることです。

最後に、このような簡潔性のもう 1 つの利点は、バックドアのような未知の機能がないかと心配することなく、この基本システムを安全に構築できることです。システム インテグリティに対する攻撃の危険が高まっていることを考慮すると、これは非常に重要な特性であり、安全性が重視されるアプリケーションには必要不可欠です。特に、筆者らのシステムのハードウェアにもこのような隠れた部分はありません。結局のところ、規模が大きすぎて誰もその全体を理解できないような基盤の上に構築されたシステムについては、何の保証もできないのです。

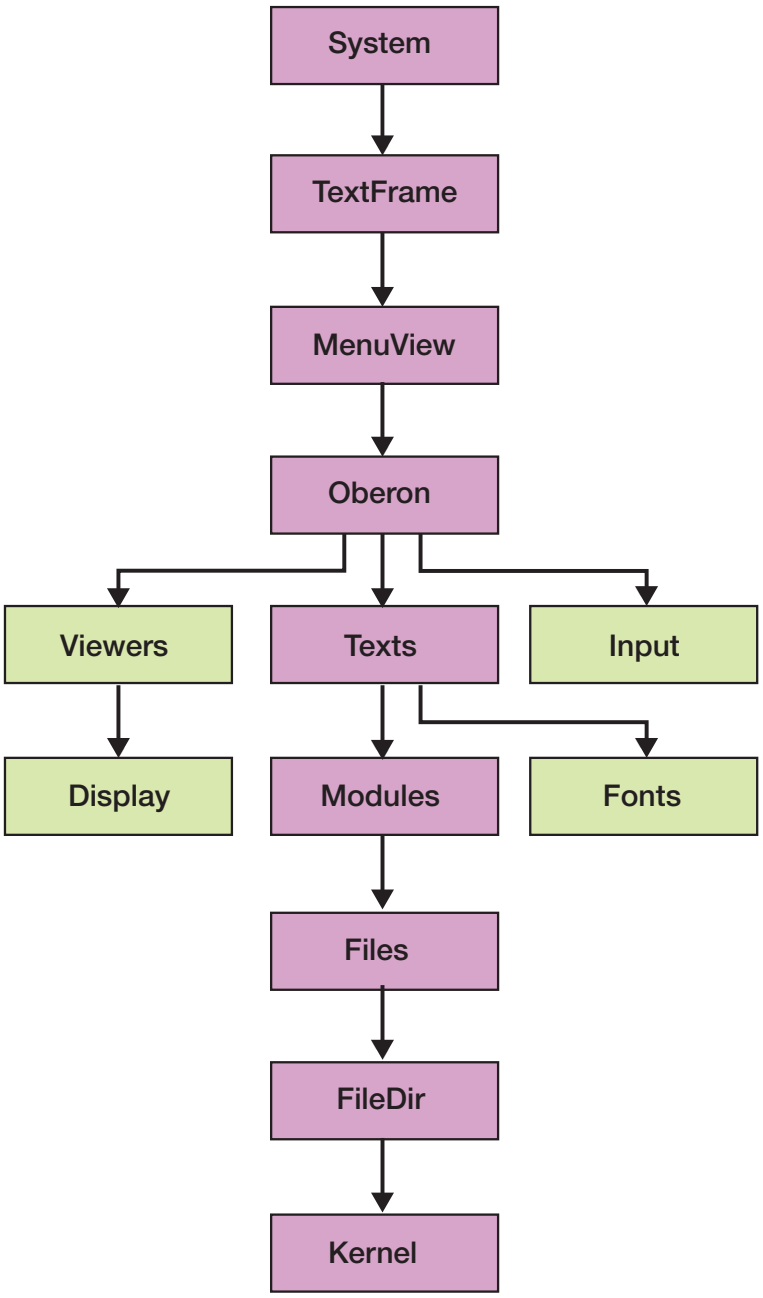



図 3 – システムとシステム モジュール

## 謝辞

ポール リード (Paul Reed) 氏の貴重な貢献に深く感謝します。リード氏は、筆者が『Project Oberon』を再編集し、システム全体を FPGA 上にインプリメントし直すように勧めてくれました。このプロジェクトにはリード氏の貢献が必要不可欠でした。ディスクを SD カードで置き換えたのはリード氏のアイデアであり、SPI、PS-2、および VID インターフェイスの Verilog インプリメンテーションにも貢献しました。

## 参考資料

1. http://www.inf.ethz.ch/personal/wirth/Oberon/Oberon07.Report.pdf
2. http://www.inf.ethz.ch/personal/wirth/Oberon/PIO.pdf
3. www.inf.ethz.ch/personal/wirth/ProjectOberon/index.html
4. www.inf.ethz.ch/personal/wirth/Oberon/PIO.pdf
5. www.inf.ethz.ch/personal/wirth/Oberon/Oberon07.Report.pdf
6. http://www.inf.ethz.ch/personal/wirth/CompilerConstruction/CompilerConstruction1.pdf
7. http://www.inf.ethz.ch/personal/wirth/CompilerConstruction/CompilerConstruction2.pdf
8. http://www.inf.ethz.ch/personal/wirth/ProjectOberon/PO.Applications.pdf (Ch. 12)

# XILINX® ウェブセミナー

All Programmable FPGA、SoC、3D IC の世界的なリーディング プロバイダーの
ザイリンクスが提供するプログラマブル ロジックからプログラマブル システム
インテグレーションのさまざまな機能と活用方法をご紹介します。
コストを抑え、最大のパフォーマンスを実現するための最新情報を手に入れてください。

**ニーズに合わせたプログラムを各種取り揃えて好評配信中!!**

## 注目のセミナー
## UltraScale アーキテクチャ概要

### FPGA入門編

FPGA の基本を理解したい方へ FPGA の全体概要を解説した入門編と、ものづくりにチャレンジする経営者、技術管理者の方へ FPGA を採用する利点をご説明します。

- ▶ 30分で判る! FPGA入門
- ▶ 15分で判る! FPGA採用理由

### FPGA/SoC 活用編

ザイリンクス FPGA/SoC を使った最先端デザインの設計手法や、さまざまなアプリケーション設計に求められるデザイン チャレンジに対するソリューションをご紹介・解説します。

- ▶ All Programmableで実現するハイエンド組込みヒューマン マシン インターフェイス
- ▶ ザイリンクス All Programmable ソリューションで実現する機能安全
- ▶ Zynq SoC を使用したマルチチャンネル リアルタイム ビデオ プロセッサの設計
- ▶ Zynq SoC を使用した最先端 エンベデッド システムの設計 ～アクセレータでのソフトウェア ボトルネックの解消方法～
- ▶ 7 シリーズ ターゲット デザイン プラットフォーム

### 開発ツール編

プログラマブルデバイスである FPGA の設計には開発ツールがキーになります。ザイリンクスが提供するユーザー フレンドリーな開発ツールの特徴や使い方、先端設計メソドロジについて解説します。

- ▶ 次世代FPGA設計手法セミナー PlanAhead デザイン解析ツール ～ 第1部、第2部、第3部、デモ ～
- ▶ AMBA AXI4 テクニカルセミナー

### FPGA/SoC 概要編

FPGA の世界トップシェアを誇るザイリンクスが提案するソリューションや、ザイリンクスの最先端 FPGA の詳細を解説します。

- ▶ UltraScale アーキテクチャ概要
- ▶ Zynq-7000 SoC アーキテクチャとエコシステム
- ▶ 28nm ザイリンクス 7 シリーズ FPGA のアジャイル ミックスド シグナル テクノロジ

セミナー内容の詳細／ご視聴は今すぐこちらから ▶▶▶ http://japan.xilinx.com/webseminar/

Removing the Barrier for FPGA-Based OpenCL Data Center Servers

# FPGA ベースの OpenCL データ センター サーバーの制約を解消

ザイリンクスの SDAccel 環境は、CPU と同様の開発環境とランタイム体験を FPGA にも提供し、データ センター設計の負担を軽減します。

**Devadas Varma**
Senior Engineering Director
SDAccel and Vivado High-Level Synthesis
Xilinx, Inc.
*dvarma@xilinx.com*

**Tom Feist**
Senior Director
Design Methodology Marketing
Xilinx, Inc.
*tfeist@xilinx.com*

中小企業のサーバー ルームから、米国の主要企業をサポートし、クラウド コンピューティング サービスへのアクセスを提供するエンタープライズ データ センターに至るまで、今日のデータ センターは、まさに現代経済の屋台骨と言うことができます。NRDC (Natural Resources Defense Council) によると、データ センターは、米国内で電力消費量が最も大きく、最も急速に伸びている分野の 1 つです。2013 年の米国内のデータ センターの消費電力は、ニューヨーク市全世帯の消費電力の 2 倍を上回る 910 億 kWh と推定され、2020 年までに 1,400 億 kWh に達する見通しです [1]。明らかに、データ センターを拡張して信頼性を高め、運用コストを削減するには、消費電力の削減が必要不可欠です。

データ センター サーバーのタイプは、サーバー アプリケーションに応じて異なります。多くのサーバーは中断なく長時間運用されるため、ハードウェアの信頼性と耐久性が非常に重視されます。サーバーは一般的なコンピューター部品で製造できますが、ミッション クリティカルなエンタープライズ サーバーには、しばしばグラフィックス プロセッシング ユニット (GPU) やデジタル シグナル プロセッサ (DSP) など、アプリケーションを高速化するための専用ハードウェアが使用されています。現在、多くの企業が、フィールド プログラマブル ゲート アレイ (FPGA) の追加によって比較的低消費電力で高度並列処理アーキテクチャを実現するソリューションに注目しています。ザイリンクスの新しい SDAccel™ 開発環境は、使い慣れた CPU/GPU 向け環境と同様の環境を開発者に提供することで、この分野への FPGA の利用を制約しているプログラミングの問題を解決します。

## 単位ワット当たり性能の向上

Amazon Web Services、Google Compute、Microsoft Azure、Facebook、中国の Baidu など、巨大な画像貯蔵庫を擁するパブリック クラウドは、非常に高速な画像認識機能を必要としています。あるインプリメンテーションで Google の科学者は、16,000 個のコンピューター プロセッサを接続して 1 つのエンティティを作成し、インターネット上で自由に動作させて自律的に学習させることで、最大級の機械学習ニューラル ネットワークを構築しました。この研究は、巨大データ センター内で利用可能な大量のコンピューター クラスターを活用した、新しい世代のコンピューター サイエンスを代表するものです。考えられるアプリケーションには、画像検索、音声認識、機械翻訳の向上などがあります。しかし、CPU のみを使用してデータ センターを設計するのは、電力効率の高い手法とは言えません。高速化と省電力化を両立させるには、代替的なソリューションが必要です。

中国最大の検索エンジン専門企業である Baidu (百度) 社は、ディープ ニューラル ネットワーク処理を利用して、音声認識、画像検索、自然言語処理の問題に取り組んでいます。同社は、オンライン予測にニューラル バック プロパゲーション アルゴリズムを使用する場合、FPGA ソリューションは CPU/GPU ソリューションよりもはるかに容易に拡張でき、消費電力も少ないことにすぐに気づきました [2]。

ザイリンクスの 7 シリーズ デバイスおよび Ultra- Scale™ デバイスなど、高度に集積された新世代の 28nm および 20nm FPGA ファミリは、データ センター サーバーのホスト カードおよびライン カードへの FPGA の統合の動きを加速させています。これらの FPGA は、単位ワット当たり性能が同等の CPU や GPU の 20 倍を超え、一部のアプリケーションでは従来の CPU に比べてレイテンシが最大 50 ～ 75 倍も向上します。

しかし、設計チームに FPGA ハードウェアの開発経験者が少ない ( あるいは全くいない ) 場合は、これらのデバイスを十分に活用するのに必要な RTL (VHDL または Verilog) の経験に乏しいため、FPGA への移行は難しいと考えられていました。この問題を解決するため、ザイリンクスは、プログラミングの負担を軽減する方法として、Open Computing Language (OpenCL™ ) に注目しました。

## OpenCL コードの移植性

OpenCL は、ヘテロジニアス デザインで CPU、GPU、FPGA、DSP ブロックの統合を支援する目的で、Apple Inc. が開発し、Khronos Group [3] が推進してきた技術です。ヘテロジニアス プラットフォーム上で実行されるプログラム作成用の OpenCL の枠組みを強化するために、CPU、GPU、FPGA の大手ベンダー ( ザイリンクスを含む ) は、OpenCL の言語とその API の開発に取り組んでいます。

CPU/GPU/FPGA ベンダー、サーバー メーカー、データ センター管理者の間で OpenCL がしだいに広く受け入れられていることは、すべての関係者が 1 つの包括的な事実を認識していることを示しています。その事実とは、プロセッサが 20nm 以下のプロセス技術に移行し、特別な省電力ステートが追加されても、シングル プロセッサ アーキテクチャ向けの C ベースのコンパイラでは、サーバー ラック内の全体的な消費電力をわずかしか削減できないということです。

OpenCL は、CPU、GPU、DSP、FPGA などのプロセッサで構成されるヘテロジニアスなプラットフォーム上で実行されるプログラムを作成するための枠組みです。OpenCL には、(C99 をベースとする ) プログラミング言語と、プラットフォームの制御とターゲット デバイス上でのプログラム実行用のアプリケーション プログラミング インターフェイス (API) が含まれています。OpenCL は、タスクベースおよびデータベースの並列性を利用したパラレル コンピューティング機能を提供します。

## ザイリンクスの OpenCL 用 SDAccel 開発環境

ザイリンクスは、約 10 年にわたってドメイン固有仕様環境の開発に取り組んできました。データ センター管理者とサーバー / スイッチ メーカー双方のデータ センターのパフォーマンスへの関心の高まりが、データ センター アプリケーションのデザイン最適化のための統一的な環境に向けた垂直的発展を促しました。その結果、アプリケーションの高速化に対応した OpenCL 開発環境である SDAccel ™ が完成しました。

新しいザイリンクス SDAccel 環境 ( 図 1) は、完全な FPGA ベースのハードウェアと OpenCL ソフトウェアをデータ センター アプリケーション開発者に提供します。SDAccel には、オンチップ FPGA リソースを効率的に利用できる高速なアーキテクチャ最適化コンパイラと、コードの開発、プロファイリング、デバッグ用の Eclipse 統合設計環境 (IDE) ベースの使い慣れたソフトウェア開発フローが含まれています。この IDE は、CPU/GPU 向けと同様の作業環境を提供します。

さらに、SDAccel は、ザイリンクスの動的にリコンフィギャラブルなテクノロジを利用して、各種のアプリケーション向けに最適化されたアクセラレータ カーネルをオンザフライで切り替えることができます。アプリケーションの実行時に、サーバーの CPU と

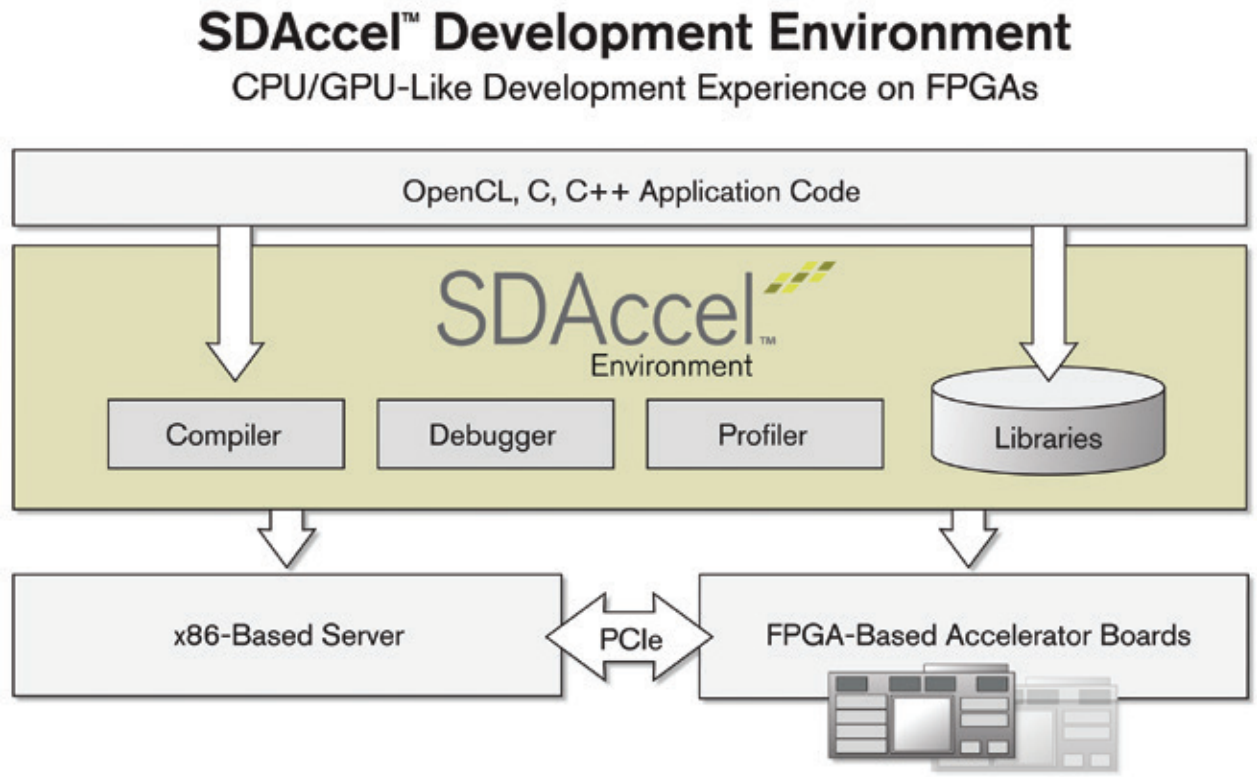



図 1 – SDAccel 環境には、アーキテクチャ最適化コンパイラ、ライブラリ、デバッガー、プロファイラが含まれ、CPU/GPU 向け環境と同様のプログラミング体験を提供します。

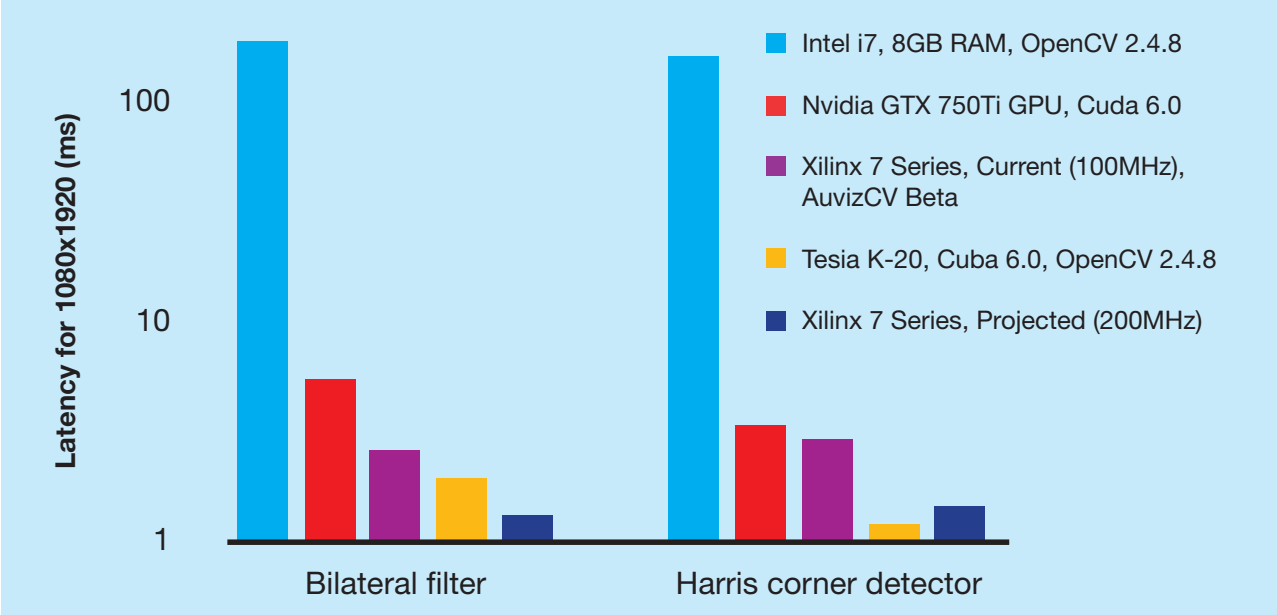



図 2 – CPU、GPU、FPGA の各アーキテクチャ向けに OpenCL で記述された
ビデオ処理アルゴリズムは、FPGA 上で高速で実行されます
(ベンチマークは Auviz が AuvizCV ライブラリを使用して実行)

FPGA 間のインターフェイスを中断することなく、FPGA の内外で複数のカーネルを切り替えることができるため、アプリケーションをノンストップで高速化できます。

SDAccel のアーキテクチャ最適化コンパイラにより、ソフトウェア開発者は、ストリーミング、低レイテンシ、およびカスタム データパスの各アプリケーションを最適化してコンパイルできます。SDAccel コンパイラは、高性能なザイリンクス FPGA をターゲットとして、C、C++、OpenCL を自由に組み合わせたソース コードをサポートします。SDAccel コンパイラは、容易にアプリケーションを移行してコストを削減できるように、コードの互換性と従来のソフトウェア プログラミング モデルを維持しながら、ハイエンド CPU の 10 倍のパフォーマンスと GPU の 10 分の 1 の消費電力を提供します。

FPGA に最適化された多彩なアプリケーション高速化ライブラリを備えた FPGA ベースの開発環境は、SDAccel 以外にありません。このライブラリには、OpenCL ビルトイン、任意精度データ型 ( 固定小数点 )、浮動小数点関数、math.h 関数、ビデオ関数、信号処理関数、線形代数関数が含まれます。

ネストされた複雑なデータパスを含むビデオ処理などの実際の演算ワークロードでは、FPGA ファブリック本来の柔軟性が、CPU および GPU の固定されたアーキテクチャよりもパフォーマンスと消費電力の面で有利なことは明らかです。図 2 のベンチマークに示すように、SDAccel でコンパイルされた FPGA ソリューションは、同じコードを CPU にインプリメントした場合の性能を上回り、GPU にインプリメントしたソリューションに匹敵するパフォーマンスを提供します。

バイラテラル フィルターと Harris コーナー検出器は、いずれもデバイス グローバル メモリを使ってカーネル間でデータを転送する標準的な OpenCL 設計手法を使用してコーディングされています。SDAccel で生成された FPGA インプリメンテーションは、高帯域幅のメモリ転送と低レイテンシの演算に使用されるオンチップ メモリ バンクを作成することによって、メモリ アクセスを最適化します。これらのアプリケーション固有のメモリ バンクの作成と使用は、SDAccel コンパイラのアーキテクチャ対応型機能を示しています。

## ソフトウェア ワークフロー

FPGA は、長きにわたり、CPU/GPU インプリメンテーションと比べたアルゴリズムのパフォーマンス向上と消費電力削減への期待を担ってきました。しかし現在まで、FPGA を効率的に利用するために必要な独自のプログラミング方式が、その実現を制約してきました。SDAccel は、オンザフライでのインシステム リコンフィギュレーションを可能にするソフトウェア ワークフローをサポートすることで、このような制約を解消し、データ センターのハードウェア アクセラレーションの投資収益を最大限に引き上げます。SDAccel は、機能と使いやすさの面で競合ツールをはるかに超える、独自の完全な FPGA ベースのソリューションです。詳細は、http://japan.xilinx.com/products/design-tools/sdx/sdaccel.html. を参照してください。

## 参考資料

1. http://www.nrdc.org/energy/data-center-efficiency-assessment.asp
2. http://www.pcworld.com/article/2464260/microsoft-baidu-find-speedier-search-results-through-specialized-chips.html
3. https://www.khronos.org/opencl

# Optimizing an OpenCL Application for Video Watermarking in FPGAs

# FPGA のビデオウォーターマーキング用 OpenCL アプリケーションを最適化

## ザイリンクスの SDAccel 開発環境は、メモリバウンドの問題を解決する最適化手法を提供します。

**Jasmina Vasiljevic**
Researcher
University of Toronto
*vasilijev@eecg.toronto.edu*

**Fernando Martinez Vallina, PhD**
Software Development Manager
Xilinx, Inc.
*vallina@xilinx.com*

現在、ビデオのストリーミングとダウンロードは消費者のインターネット トラフィックの大半を占め、クラウド コンピューティングの拡大の推進力になっています。このタイプのコンテンツ需要の急速な増大とともに、ビデオ処理アプリケーションは、専用のシステムからデータ センターへと移行しています。このような運用方式の転換は、トランスコーディングやウォーターマーキングなど、ビデオ コンテンツの準備と配信のさまざまな段階で、大量のデータ処理のニーズに対応した演算ノードの迅速な拡張を可能にします。

先頃筆者らは、ザイリンクスの SDAccel™ 開発環境を使用して、OpenCL™ で記述されたビデオ ウォーターマーキング アプリケーションを FPGA アクセラレータ カード用にコンパイルして最適化しました。ビデオ コンテンツ プロバイダーは、コンテンツの商標表示と保護にウォーターマーキングを使用しています。筆者らの目標は、Alpha Data 社の ADM-PCIE-7V3 カード上で動作するウォーターマーキング アプリケーションを設計し、解像度 1080p の高精細度 (HD) ビデオを 30fps ( フレーム / 秒 ) の目標スループットで処理することです。

SDAccel 開発環境では、基礎となる FPGA のインプリメンテーション ツールの知識を持たない設計者でも、OpenCL で記述されたアプリケーションを FPGA 用にコンパイルできます。ビデオ ウォーターマーキング アプリケーションは、SDAccel で利用できる主な最適化手法を紹介するのに理想的な例です。

## ビデオ ウォーターマーキングとロゴの挿入

ビデオ ウォーターマーキング アルゴリズムの主な機能は、ビデオ ストリームの特定の位置にロゴを挿入することです。ウォーターマークに使用されるロゴは、アクティブ ロゴまたはパッシブ ロゴのいずれかです。アクティブ ロゴは通常、短いビデオ クリップの繰り返しで表現され、パッシブ ロゴは静止画像です。

放送会社が自社のビデオ ストリームに商標を表示する場合、最も一般的な手法は、パッシブ ウォーターマークとして企業ロゴを使用することです。したがって、これが筆者らのサンプル デザインの目的となります。このアプリケーションは、次の式の演算に基づいて、パッシブ ロゴをピクセル単位の粒度で挿入します。

$$out\_y[x][y] = (255-mask[x][y]) * in\_y[x][y] + mask[x][y] * logo\_y[x][y]$$

$$out\_cr[x][y] = (255-mask[x][y]) * in\_cr[x][y] + mask[x][y] * logo\_cr[x][y]$$

$$out\_cb[x][y] = (255-mask[x][y]) * in\_cb[x][y] + mask[x][y] * logo\_cb[x][y]$$

入力フレームと出力フレームは、YCbCr カラー スペースを使用してピクセルを表現した 2 次元配列です。このカラー スペースは、Y、Cb、Cr の 3 つの成分で各ピクセルを表現します。Y は輝度成分、Cb は青の色差成分、Cr は赤の色差成分です。各成分は 8 ビット値で表現され、1 ピクセル当たり合計 24 ビットになります。

ロゴは、挿入されるコンテンツを含む 2 次元イメージです。マスクもイメージですが、マスクにはロゴの輪郭のみが含まれます。マスク内のピクセルは白または黒のいずれかです。マスクの白のピクセルの位置にロゴが挿入され、黒のピクセルの位置には元のピクセルがそのまま残ります。図 1 に、ビデオ ウォーターマーク アルゴリズムの処理の例を示します。

## ターゲット システムと初期インプリメンテーション

筆者らがアプリケーションを実行したシステムを図 2 に示します。このシステムは、PCIe® リンクを介して x86 プロセッサと通信する Alpha Data 社の ADMPCIE- 7V3 カードで構成されます。このシステム内で、ホスト プロセッサは、ディスクから入力ビデオ ストリームを取得し、デバイス グローバル メモリに転送します。デバイス グローバル メモリとは、FPGA から直接アクセスできる FPGA カード上のメモリです。ビデオ フレームをデバイス グローバル メモリに配置するほか、ホストからロゴとマスクをアクセラレータ カードに転送し、オンチップ メモリに配置して低レイテンシの BRAM メモリを利用できるようにします。このアプリケーションではパッシブ ロゴを使用するため、オンチップ

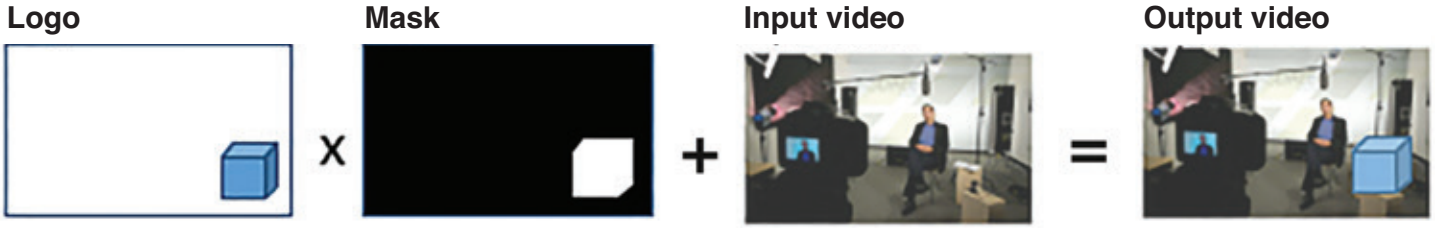



図 1 – ビデオ ウォーターマーキング アルゴリズムの動作

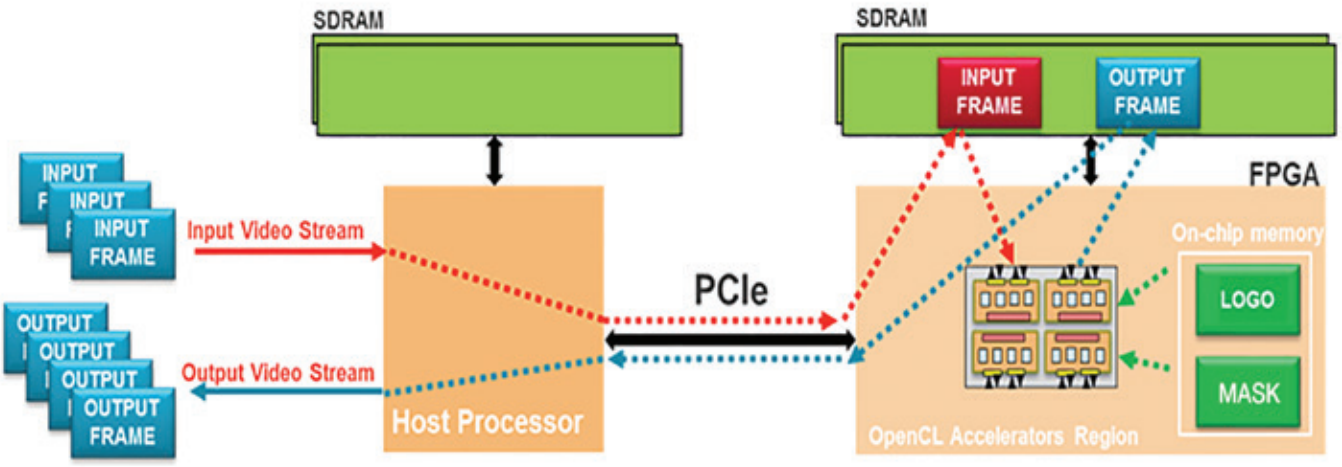



図 2 – ビデオ ウォーターマーキング アプリケーションのシステム概要

メモリに格納されるのは、静止画像と配置位置のデータのみです。

データの準備ができたら、ホスト プロセッサは FPGA ファブリックのウォーターマーキング カーネルにスタート信号を送信します。この信号は、このカーネルの 3 つの動作をトリガーします。すなわち、カーネルは、デバイス グローバル メモリから入力ビデオ フレームのフェッチを開始し、マスクによって定義された位置にロゴを挿入し、処理済みのフレームをデバイス グローバル メモリに戻してプロセッサがフェッチできるようにします。

ビデオ ストリームの各フレームのデータ転送と演算の協調は、図 3 のコードによって実行されます。

このコードはホスト プロセッサ上で動作し、FPGA アクセラレータ カードへのビデオ フレームの送信、アクセラレータの起動、FPGA アクセラレータ カードからの処理済みフレームの取得を実行します。

FPGA 用ウォーターマーキング アルゴリズムの最初のインプリメンテーションを図 4 に示します。このコードは機能的には正しくこのアプリケーションをインプリメントしていますが、パフォーマンスの最適化を行わず、FPGA ファブリックの機能も考慮に入れていません。このコードは現状のまま SDAccel でコンパイルして Alpha Data カード上で実行することができ、最大スループットは 0.5 fps になります。

図 4 のコードからわかるように、このウォーターマーキング アルゴリズムは、演算量の多いデザインではありません。処理時間の大半は、ビデオ フレームの読み出し / 書き込みのメモリ アクセスに消費されます。したがって、筆者らはサンプル デザインを最適化する際にメモリ帯域幅に焦点を絞りました。

## ベクトル化を使用したメモリ アクセスの最適化

ソフトウェアでプログラム可能な他のファブリックに対する FPGA ファブリックの利点の 1 つは、メモリへのインターコネクト バスを柔軟にコンフィギュレーションできることです。SDAccel は、アプリケーション カーネルに基づいて、メモリへのカスタムサイズのデータパスおよびアーキテクチャを作成します。一度に複数のピクセルを処理するようにコードを変更することで、カーネルからのメモリ帯域幅の拡大が推論可能になります。この手法はベクトル化と呼ばれます。

適切なベクトル化のレベルは、アプリケーションと使用している FPGA アクセラレータ カードによって異なります。この Alpha Data カードの場合、デバイス グローバル メモリへのインターフェイスの幅は 512 ビットで、SDAccel 内のカーネルが利用できる AXI インターコネクトの最大幅と一致します。この 512 ビットの境界に基づいて、一度に 20 ピクセルを処理するようにアプリケーションを変更します (24 ビット / ピクセル × 20 ピクセル = 504 ビット)。SDAccel はベクター データ型を完全にサポートしています。したがって、このアプリケーションでは、図 5 に示すように、すべての配列のデータ型を char20 に変更することによって簡単にコードをベクトル化でき、12 fps のスループットが得られます。

## バーストを使用したメモリ アクセスの最適化

ベクトル化によってアプリケーションのパフォーマンスは大幅に向上しますが、それだけでは 30fps の目標に達するには不十分です。カーネルが発行するメモリ転送は一度に 20 ピクセルに限られるため、アプリケーションのメモリバウンドは解消されていません。アプリケーションに対するメモリバウンドの影響を軽減するには、20 ピクセルを超えるデータ セットに対してメモリのバースト読み出し / 書き込み操作を生成するように、カーネル コードを変更する必要がありました。変更したカーネル コードを図 6 に示します。

カーネル コードの最初の変更では、ピクセルのブロックを一度に格納するカーネル内のオンチップ ストレージを定義しました。このオンチップ メモリは、カーネル コード内で宣言される配列によって定義されます。このコードは、メモリに対するバースト トランザクションを記述するために、DDR からカーネル内部の BRAM ストレージへデータ

のブロックを移動する memcpy コマンドをインスタンシエートします。オンチップ メモリ リソースのサイズと処理されるデータの量に基づいて、1 つのビデオ フレームは 1920 × 54 ピクセルの 20 ブロックに分割されます ( 図 7 を参照 )。

memcpy 操作によってデータ ブロックがカーネル配列内に配置されると、アルゴリズムはそのデータ ブロックに対してウォーターマーキング アルゴリズムを実行し、結果をカーネル配列に戻します。ブロック処理の結果は、memcpy 操作を使って DDR メモリに転送し戻されます。与えられたフレーム内のすべてのブロックの処理が完了するまで、この操作シーケンスが 20 回繰り返されます。このようにカーネル コードを変更した結果、システム パフォーマンスは 38 fps となり、元の目標の 30 fps を超えました。

Host Code

SEND FRAME
COMPUTE FRAME
RECEIVE FRAME

```
for (i=0; i<FRAMES; i++) {
  // Send a video frame to the FPGA device
  err = clEnqueueWriteBuffer(commands, d_frin, CL_TRUE, 0,
        sizeof(int) * LENGTH_FRAN, h_frin, 0, NULL, &writeEvent0;
  clWaitForEvents(1, &writeEvent);

  // Run logo insertion on the input frame
  err = clEnqueueTask(commands, kernel_load_block, 0, NULL,
        &kernelEvent);
  clWaitForEvents(1, &kernelEVent);

  // Read the output frame back to CPU
  err = clEnqueueReadBuffer( commands, d_frout, CL_TRUE, 0,
        sizeof(int) * LENGTH_FROUT, h_frout, 0, NULL, &readEvent);
  clWritForEvents(1, &readEvent);
}
```

図 3 – 各フレームのデータ転送と演算を協調させるコード

```
for (y=0; y<FRAMES_HEIGHT; y++) {
  for (x=0; x<FRAMES_WIDTH/20; x++) {

        i = y*FRAME_WIDTH/20 + x;
        out_y[i]   = (255-mask[i]) * in_y[i]   + mask[i] * logo_y[i];
        out_cr[i]  = (255-mask[i]) * in_cr[i]  + mask[i] * logo_cr[i];
        out_cb[i]  = (255-mask[i]) * in_cb[i]  + mask[i] * logo_cb[i];
  }
}
```

図 4 – ウォーターマーキング カーネルの初期インプリメンテーション

```
for (y=0; y<FRAMES_HEIGHT; y++) {
  for (x=0; x<FRAMES_WIDTH/20; x++) {

        i = y*FRAME_WIDTH/20 + x;

        out_y[i]   = (255-mask[i]) * in_y[i]   + mask[i] * logo_y[i];
        out_cr[i]  = (255-mask[i]) * in_cr[i]  + mask[i] * logo_cr[i];
        out_cb[i]  = (255-mask[i]) * in_cb[i]  + mask[i] * logo_cb[i];
  }
}
```

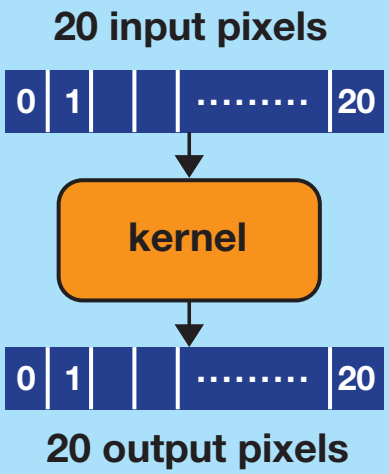



図 5 – ベクトル化されたカーネル コード

```
char20 l_in_y[BLOCK_WIDHT*BLOCK_HEIGHT/20];
char20 l_in_cr[BLOCK_WIDHT*BLOCK_HEIGHT/20];
char20 l_in_cb[BLOCK_WIDHT*BLOCK_HEIGHT/20];

char20 l_out_y[BLOCK_WIDHT*BLOCK_HEIGHT/20];
char20 l_out_cr[BLOCK_WIDHT*BLOCK_HEIGHT/20];
char20 l_out_cb[BLOCK_WIDHT*BLOCK_HEIGHT/20];

for (block_id=0; block_id<FRAME_HEIGHT/BLOCK_HEIGHT; block_id++){

  memcpy(l_in_y, in_y + block_id*BLOCK_WIDHT*BLOCK_HEIGHT/20, BLOCK_WIDHT*BLOCK_HEIGHT*3);
  memcpy(l_in_cr, in_cr + block_id*BLOCK_WIDHT*BLOCK_HEIGHT/20, BLOCK_WIDHT*BLOCK_HEIGHT*3);
  memcpy(l_in_cb, in_cb + block_id*BLOCK_WIDHT*BLOCK_HEIGHT/20, BLOCK_WIDHT*BLOCK_HEIGHT*3);

  for (y=0; y<BLOCK_HEIGHT; y++) {
    for (x=0; x<BLOCK_WIDTH/20; x++) {

      i = y*BLOCK_WIDTH/20 + x;

      l_out_y[i]  = (255-mask[i]) * l_in_y[i]  + mask[i] * logo_y[i];
      l_out_cr[i] = (255-mask[i]) * l_in_cr[i] + mask[i] * logo_cr[i];
      l_out_cb[i] = (255-mask[i]) * l_in_cb[i] + mask[i] * logo_cb[i];
  }

  memcpy(out_y, l_out_y + block_id*BLOCK_WIDHT*BLOCK_HEIGHT/20, BLOCK_WIDHT*BLOCK_HEIGHT*3);
  memcpy(out_cr, l_out_cr + block_id*BLOCK_WIDHT*BLOCK_HEIGHT/20, BLOCK_WIDHT*BLOCK_HEIGHT*3);
  memcpy(out_cb, l_out_cb + block_id*BLOCK_WIDHT*BLOCK_HEIGHT/20, BLOCK_WIDHT*BLOCK_HEIGHT*3);
}
```



図 6 – バースト データ転送に最適化されたカーネル コード

## 幅広い適用可能性

SDAccel を使用したこのようなアプリケーションの作成に必要な最適化手法は、ソフトウェア最適化です。これらの最適化手法は、GPU などの FPGA 以外のプロセッシング ファブリックからパフォーマンスを引き出すために必要となる手法と同様のものです。SDAccel を使用した結果、PCIe リンクの動作の設定、ドライバー、IP コアの配置およびインターコネクトの詳細に配慮する必要がなくなり、設計者はターゲット アプリケーションのみに集中できます。

筆者らのウォーターマーキング アプリケーションの最適化手法は、SDAccel を使ってコンパイルされるすべてのデザインに応用できます。実際、ビデオ ウォーターマーキングは、ザイリンクスの SDAccel で利用可能な最適化手法を紹介する優れた「ハウツー」教材となります。


図 7 – ブロックに分割されたビデオ フレーム


# GET PUBLISHED


## 記事投稿のお願い

**みなさんも Xcell Publications の記事を書いてみませんか？　執筆は思ったより簡単です。**

Xcell 編集チームは、プランニング、コピー編集、グラフィックス開発、ページ レイアウトなどの編集プロセスを通じて、アイデアの展開から記事の出版まで、新しい執筆者の方や経験豊富な方々を日頃からお手伝いしています。このエキサイティングで実りの多いチャンスの詳細は、下記までお問い合わせください。

Xcell Publication 発行人　Mike Santarini (xcell@xilinx.com)

Xcell PUBLICATIONS

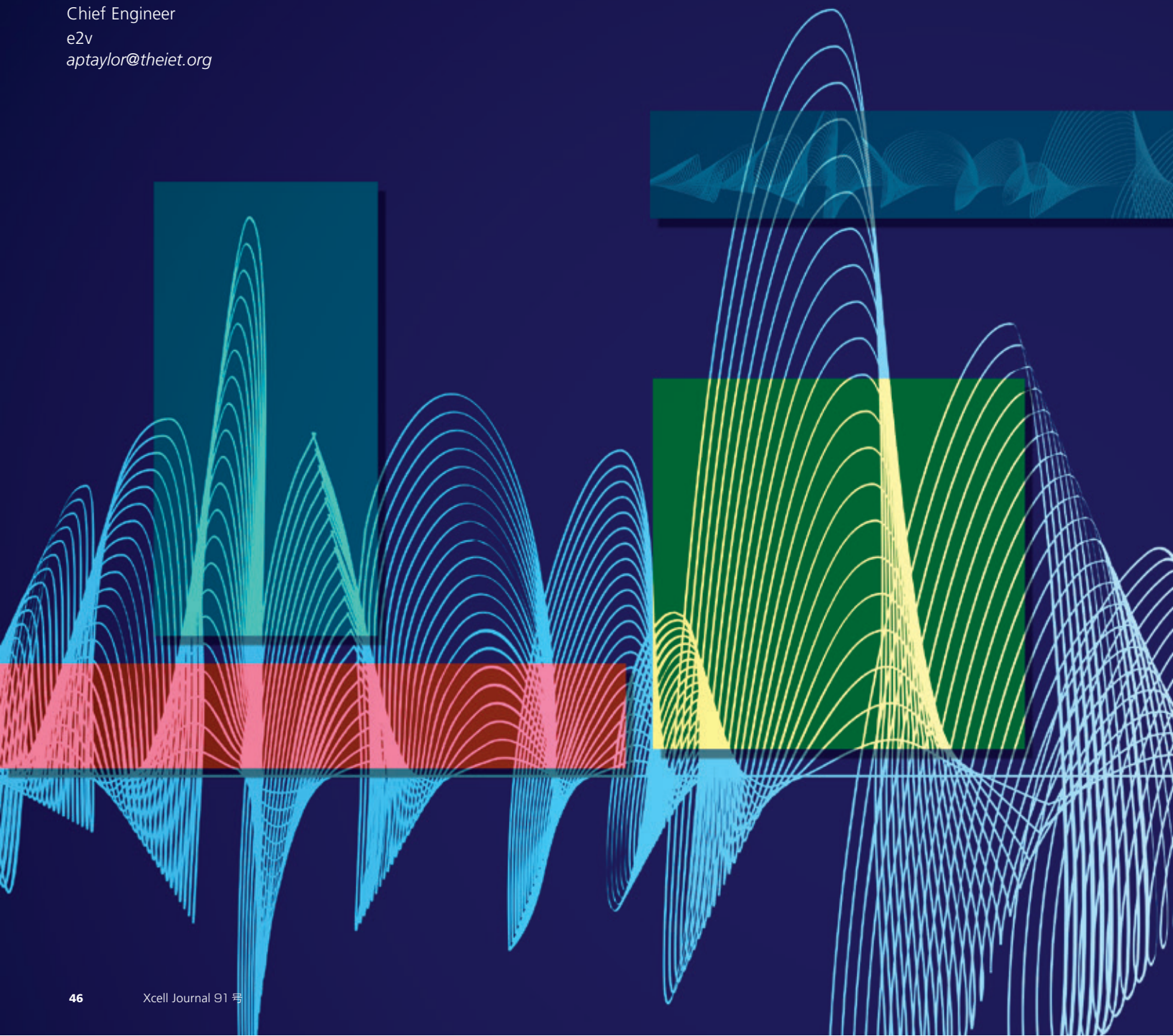
japan.xilinx.com/xcell/

XILINX

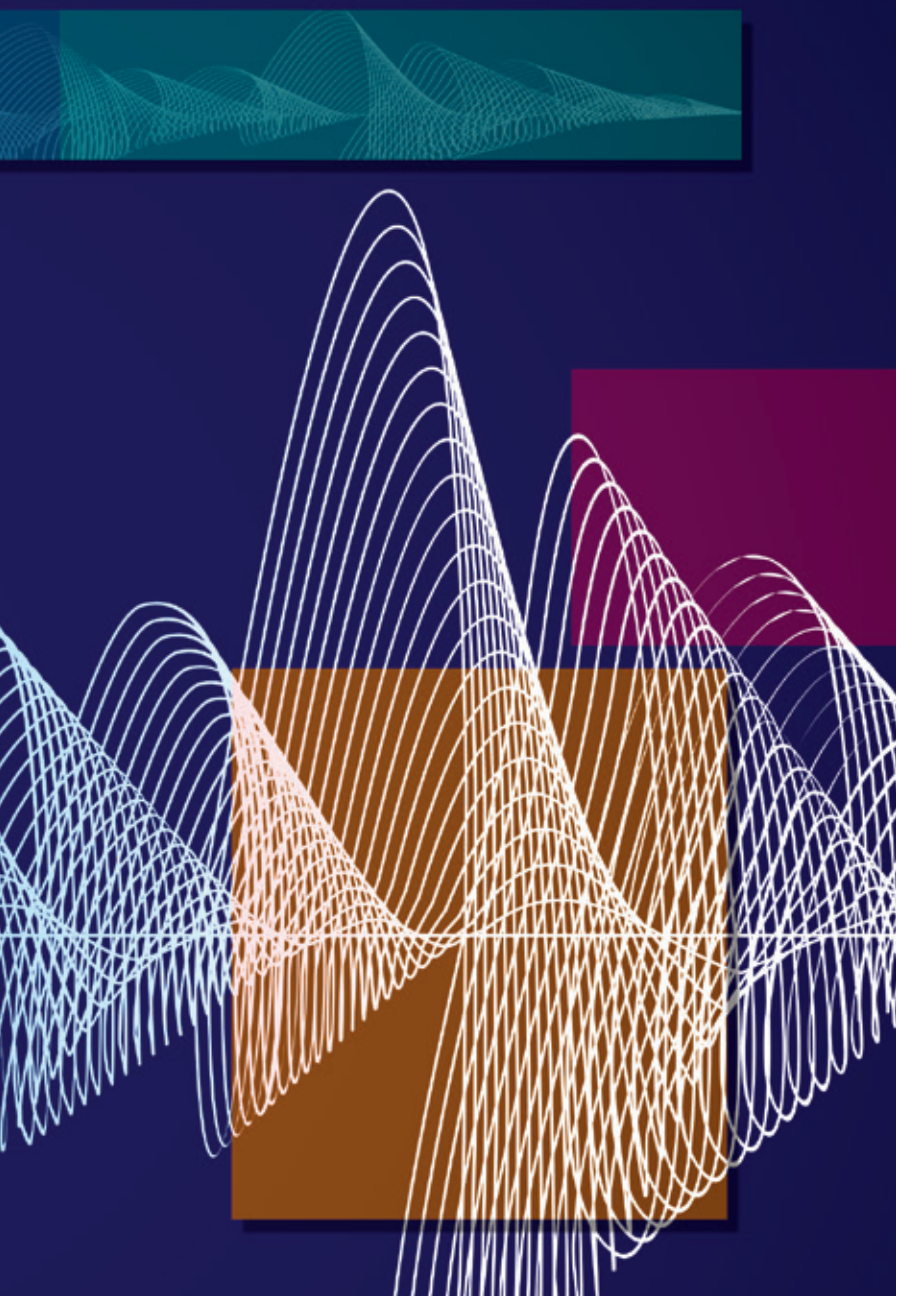
# Coming to Grips with the Frequency Domain

# 周波数ドメインの基本を把握

**Adam P. Taylor**
Chief Engineer
e2v
*aptaylor@theiet.org*



**多くのアプリケーションでは、周波数ドメインでの作業能力が必要とされます。この記事では、FPGA デザインにどのように周波数ドメインを組み込むかについて説明します。**



多くの技術者は、周波数ドメインの作業は時間ドメインの作業よりも難しいと感じています。おそらくこれは、周波数ドメインには複雑な数学が関わってくるためです。しかし、ザイリンクスの FPGA ベースのソリューションの真の潜在能力を活用するには、両方のドメインで円滑に作業できる能力が必要です。

幸いなことに、周波数ドメインの使い方を習得するのは、思ったほど難しくありません。ユーザーが設計したカスタム モジュールまたは市販の IP モジュールを使用して、時間ドメインと周波数ドメイン間で信号を容易に変換するための支援が得られます。また、周波数ドメインで高速処理を実現するための手法も用意されています。

## 時間ドメインと周波数ドメイン

信号の解析と操作は、時間ドメインと周波数ドメインのどちらでも実行できます。時間ドメインでは時間の経過とともに変化する信号を解析し、周波数ドメインではその信号の周波数の解析を行います。いつ、どちらの解析を実行するかは、設計プロジェクトの技術者に要求される基本的な判断力の 1 つです。

電子システム内の被測定信号は、通常はセンサーから出力されるか、またはシステムの他の部分から生成される、変化する電圧、電流、または周波数です。時間ドメインでは、信号の振幅、周波数、および周期とともに、信号の立ち上がり時間と立ち下がり時間などのより興味深いパラメーターを測定できます。ラボ環境で時間ドメインの信号を観察する場合、オシロスコープまたはロジック アナライザーを使用するのが一般的です。

しかし、周波数ドメインにも信号パラメーターは存在します。周波数ドメインに含まれる情報にアクセスするには、周波数ドメインで信号を解析する必要があります。周波数ドメインでは、信号の周波数成分、それらの振幅、各周波数の位相を特定できます。周波数ドメインの作業では、たたみ込みを容易に実行できるため、信号の操作がはるかに簡単に行えます。たたみ込みとは、2 つの信号

を結合して第 3 の信号を形成する数学的手法です。時間ドメインの解析と同じように、ラボ環境で周波数ドメインの信号を観察する場合は、スペクトラム アナライザーを使用できます。

たとえば、より大規模なシステムの電圧または温度を監視するシステムなどのアプリケーションでは、時間ドメインで作業することをお勧めします。ノイズが問題になる可能性はありますが、多くの場合は多数のサンプルの平均をとれば十分です。他方では、周波数ドメインで作業することが望ましいアプリケーションもあります。たとえば、信号処理アプリケーションで特定の信号を他の信号からフィルタリングする必要がある場合や、信号をノイズ源から切り離す必要がある場合は、周波数ドメイン内で解析するのが最も良い方法です。

時間ドメインで作業する場合、サンプリングは時間ドメイン内で行われるため、量子化されたデジタル信号の後処理はほとんど必要ありません。対照的に、周波数ドメインで作業するには、まず量子化されたデータを時間ドメインから周波数ドメインに変換する変換アプリケーションが必要になります。同様に、後処理されたデータを周波数ドメインから出力するには、時間ドメインへの逆方向の変換を実行する必要があります。

## 両ドメイン間の変換方法

信号のタイプによって ( 繰り返し信号か非繰り返し信号か、ディスクリート信号か非ディスクリート信号か )、時間ドメインと周波数ドメイン間で信号を変換するには、フーリエ級数、フーリエ変換、Z 変換などの多くの手法があります。特に電子信号処理アプリケーションおよび FPGA アプリケーションでは、通常はフーリエ変換のサブセットである離散フーリエ変換 (DFT) が使用されます。DFT を使用して、周期的なディスクリート信号を解析します。すなわち、これらの信号は、多くのアプリケーションでシステム内の A/D コンバーターによって供給されるサンプリング周波数で、等間隔に配列された多数の n ビット サンプルで構成されます。

簡単に言うと、DFT が行う操作は、入力信号を、その信号の正弦波成分と余弦波成分を表す 2 つの出力信号に分解することです。N 個のサンプルの時間ドメイン シーケンスに対して、DFT は、N/2+1 個の余弦波サンプルと N/2+1 個の正弦波サンプルの 2 つのグループを返します。余弦波サンプルは実数成分、正弦波サンプルは虚数成分と呼ばれます ( 図 1)。入力信号の幅が n ビットの場合、実数サンプルと虚数サンプルの幅は n/2 になります。

次の式からわかるように、DFT を計算するアルゴリズムは非常に簡単です。

$$ReX[k] = \sum_{i=0}^{N-1} x[i] \cos\left(\frac{2\pi ki}{N}\right)$$

$$ImX[k] = -\sum_{i=0}^{N-1} x[i] \sin\left(\frac{2\pi ki}{N}\right)$$

ここで、x[i] は時間ドメイン信号であり、i は 0 から N-1 までの範囲、k は 0 から N/2 までの範囲です。このアルゴリズムは相関法と呼ばれ、入力信号をそのイテレーションの正弦波または余弦波で乗算し、信号の振幅を求めるものです。

もちろん、アプリケーション内のどこかのポイントで、周波数ドメインから時間ドメインに戻る必要があります。この目的のために、実数波形と虚数波形を結合する合成式を使用して、時間ドメイン信号を次のように再作成できます。

$$x[i] = \sum_{k=0}^{N/2} Re\bar{X}[k] \cos\left(\frac{2\pi ki}{N}\right) + \sum_{k=0}^{N/2} Im\bar{X}[k] \sin\left(\frac{2\pi ki}{N}\right)$$

ただし、ReX と ImX は余弦波と正弦波のスケーリングされたバージョンです。したがって、余弦波と正弦波をスケーリングする必要があります。そこで、ImX[k] または ReX[k] は N/2 で除算され、ReX と ImX の値が求められます。ただし、ReX[0] と ReX[N/2] の場合のみ、ImX[k] と ReX[k] は N で除算されます。この手法は、逆離散フーリエ変換 (IDFT) と呼ばれます ( この名称の理由は明らかです )。

DFT と IDFT の計算に使用されるアルゴリズムについてよく理解したら、これらのアルゴリズムの用途を知っていると便利です。Octave、MATLAB®、さらには Excel などのツールを使用して、キャプチャしたデータの DFT 計算を実行できます。また、オシロスコープなどの多くのラボ ツールは、要求に

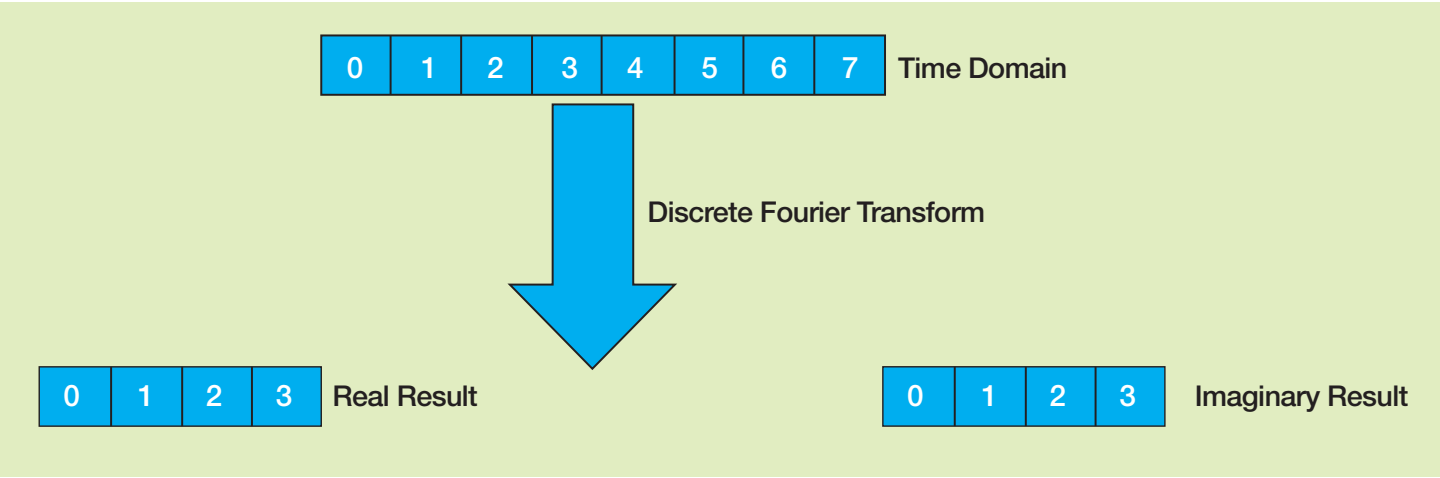



図 1 – 時間ドメインの N 個のビットから、周波数ドメインの n/2 個の実数ビットおよび n/2 個の虚数ビットへの変換

応じて DFT を実行できます。

ただし、上記の DFT および IDFT は、入力が複素数ではなく実数であるという理由で、実数 DFT および実数 IDFT と呼ばれていることに注意してください。このことがなぜ重要かは、すぐに明らかになります。

## 離散フーリエ変換の用途

通信から、画像処理、レーダー、およびソナーに至るまで、FPGA にインプリメントされる解析手法の中で、離散フーリエ変換ほど強力で適応性に富んだ手法はまずありません。確かに DFT は、非常に広く使用されている FPGA ベースのアプリケーションの基盤を形成します。すなわち DFT は、有限インパルス応答 (FIR) フィルターの係数の生成の基礎となります (Xcell Journal 英語版 78 号の記事「Ins and Outs of Digital Filter Design and Implementation」を参照 )。

しかし、DFT の用途はフィルタリングに限られません。通信処理の分野では、DFT と IDFT を使用して、通信チャネルのチャネライゼーションとリコンビネーションを実行しています。スペクトル監視アプリケーションでは、DFT と IDFT を使用して、監視対象の帯域内にどのような周波数が存在するかを判定しています。また画像処理の分野では、たとえば画像パターン認識を実行するために、DFT と IDFT を使用してフィルター カーネルによる画像のたたみ込みを処理しています。これらのアプリケーションはすべて、通常は上記のアルゴリズムよりも効率的な DFT 計算アルゴリズムを使用してインプリメントされます。

結局のところ、FPGA 内の DFT の機能を理解し、インプリメントする能力は、すべての FPGA 開発者が身につけておくべきスキルと言えます。

## FPGA ベースのインプリメンテーション

上記の DFT と IDFT は、通常、各ループで N 回の計算を実行するネストされたループとしてインプリメントされます。したがって、DFT 計算の実行時間は、次の式で表されます。

$$DFTtime = N * N * Kdft$$

ここで、Kdft は、実行される各イテレーションの処理時間です。明らかに、この処理の実行には非常に長い時間がかかることがあります。この理由で、FPGA 内の DFT は、通常は高速フーリエ変換 (FFT) と呼ばれるアルゴリズムを使用してインプリメントされます。FFT は、多くの産業に非常に大きく貢献してきたため、人生で最も重要なアルゴリズムと呼ばれることもあります。

FFT は DFT アルゴリズムとは多少異なり、複素 DFT を計算します。すなわち、FFT は実数および虚数の時間ドメイン信号を受け入れ、幅 n/2 ビットではなく幅 n ビットの周波数ドメインに結果を生成します。つまり、実数 DFT の計算では、まず虚部を 0 に設定してから、時間ドメイン信号を実部に移動しなければなりません。ザイリンクス FPGA 内に FFT をインプリメントする場合、2 つの選択肢があります。設計者は、ご希望の HDL を使ってゼロから FFT を作成するか、あるいは Vivado® Design Suite IP カタログまたは他のソース内に用意された FFT IP コアを使用できます。どうしても IP コアを使いたくない理由がある場合を除いて、ザイリンクスの IP コアを使用して開発期間を短縮することを推奨します。

FFT の基本的な手法は、時間ドメイン信号を複数のシングル ポイント時間ドメイン信号に分解することです。サンプルの順序が変更されるため、このプロセスはしばしばビット反転と呼ばれます。ビット反転アルゴリズムをショートカットとして使用しない場合、これらのシングル ポイント時間ドメイン信号の生成に必要な段数は、Log2 N によって計算されます (N はビット数 )。

次に、これらのシングル ポイント時間ドメイン信号を使用して、各ポイントの周波数スペクトラムを計算します。周波数スペクトラムはシングル ポイント時間ドメインに等しいため、この計算は非常に簡単です。

FFT アルゴリズムが複雑になるのは、これらのシングル周波数ポイントを再結合するときです。時間ドメインを分解したときとは逆に、これらのスペクトラム ポイントを一度に 1 段ずつ再結合する必要があります。したがって、スペクトラムを再作成するには再度 Log2 N 段が必要であり、ここで有名な FFT のバタフライ演算が実行されます。
DFT の実行時間と比べると、FFT の実行時間は次の式で表されます。

$$FFTtime = Kfft * N Log2 N$$

その結果、DFT の計算に必要な実行時間が大幅に短縮されます。

また、FPGA 内に FFT をインプリメントする場合は、FFT のサイズも考慮に入れる必要があります。このサイズによってノイズ フロアが決まります。ノイズ フロアを下回る信号は、被測定信号として検出できません。また、FFT のサイズによって周波数ビンの間隔も決まります。FFT のサイズは次の式で計算できます。

$$FFTNoise\ Floor\ (dB) = 6.02n + 1.77 + 10log10(\frac{FFTsize}{2})$$

ここで、n は時間ドメイン内の量子化ビット数、FFTSize は FFT のサイズです。FPGA ベースのインプリメンテーションでは、この値は通常は 2 のべき乗 ( たとえば、256、512、1,024 など ) になります。周波数ビンは次の式に従って等間隔になります。

$$Bin\ Width = \frac{\frac{Fs}{2}}{FFTsize}$$

非常に簡単な例として、サンプリング周波数 (FS) が 100MHz、FFT のサイズが 128 の場合、周波数の分解能は 0.39Hz になります。これはもちろん、互いの間隔が 0.39Hz 以内の周波数は区別できないという意味です。

## サンプリングの高速化

FPGA および高性能システム内の FFT では、多くのアプリケーションが非常に高い周波数で動作します。高周波数動作のインプリメンテーションには、特有の問題が生じることがあります。

高周波数では、(1 サイクル当たり 2 つ以上のサンプルをサンプリングする ) ナイキスト サンプル レートは単に維持できません。したがって、異なる手法が必要になります。1 つの例として、ADC を使用して 3GHz のフルパワー帯域幅アナログ入力を 2.5GHz のサンプル レートでサンプリングする場合を考えます。ナイキスト レートの基準に従って、1.25GHz を超える信号は、使用する第 1 ナイキスト ゾーンにエイリアシングされます。

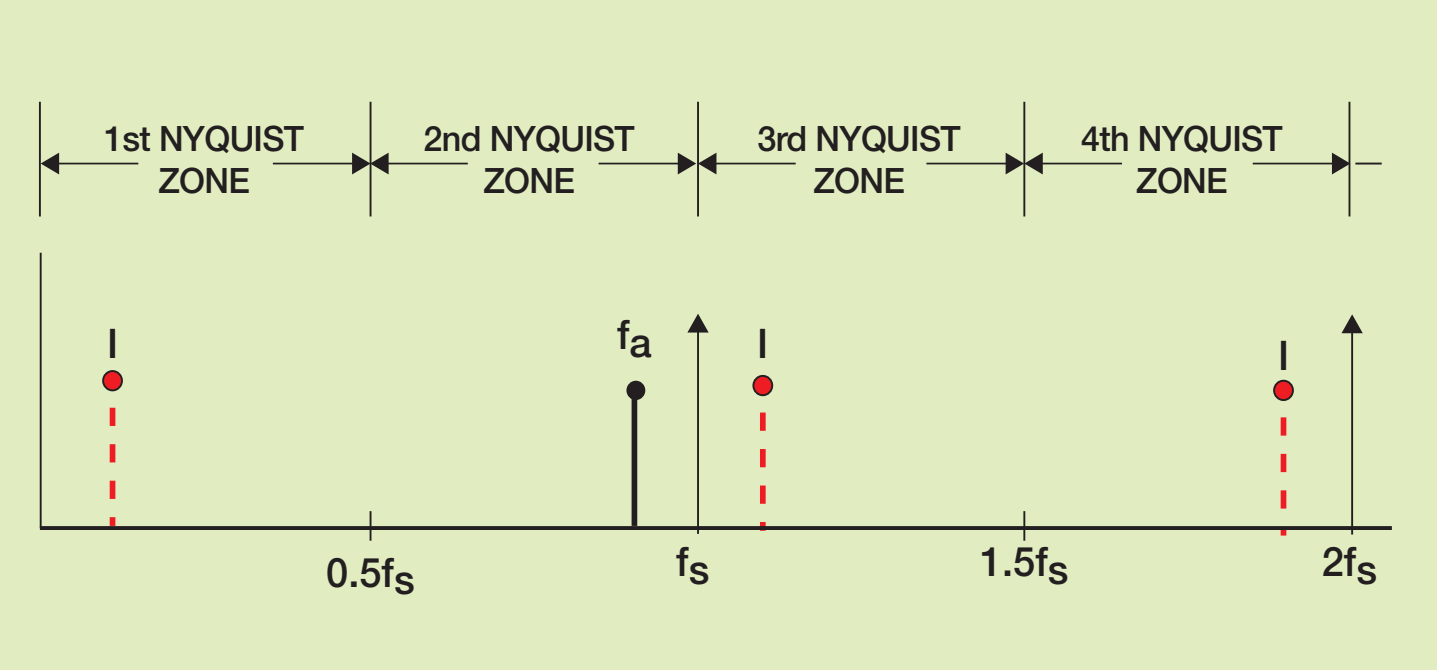



図 2 – ナイキスト ゾーンとエイリアシング

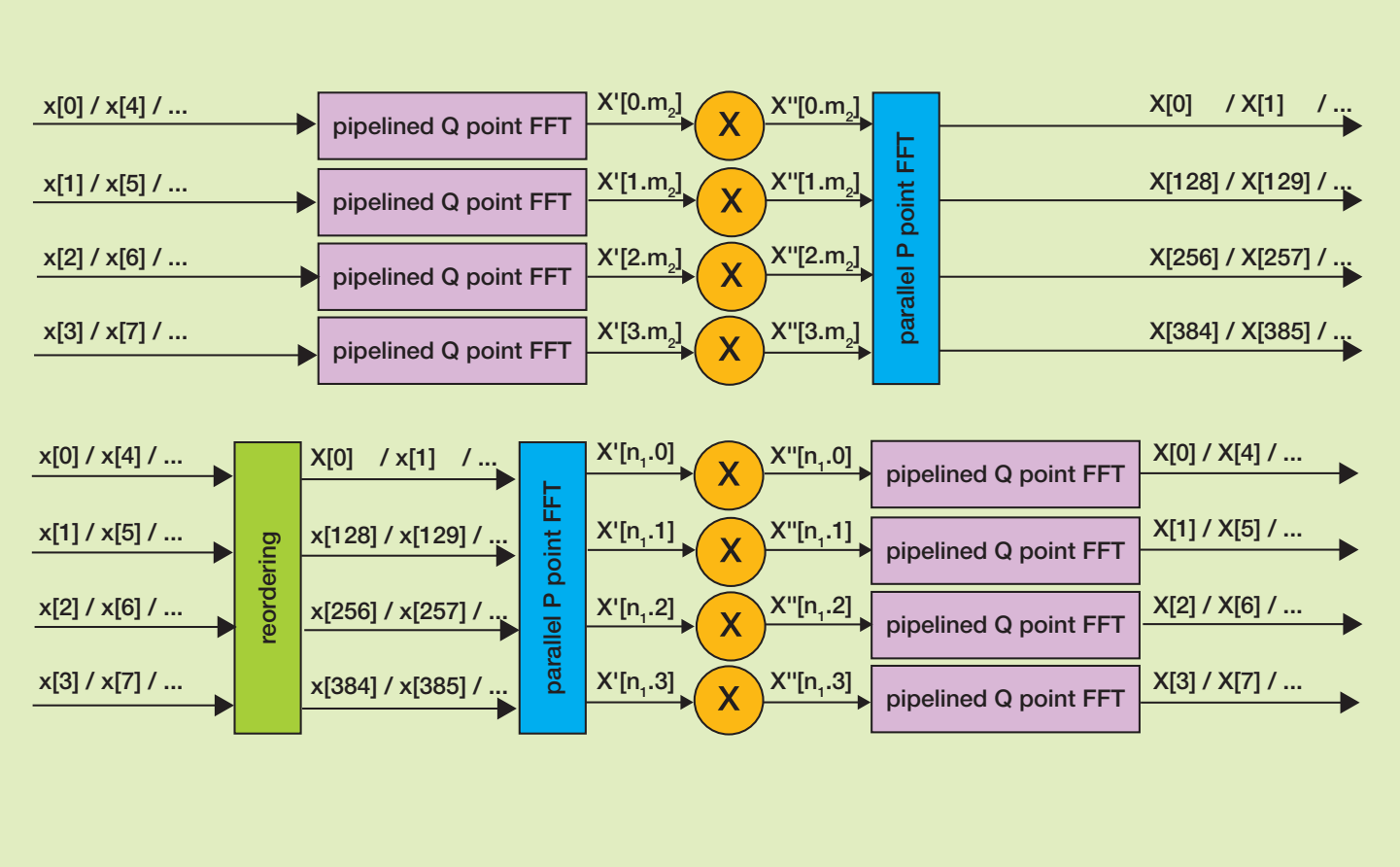



図 3 – 分割型の FFT 構造と結合型の FFT 構造

これらのエイリアシングされたイメージは、基本信号の高調波成分であり、図 2 に示すように非エイリアス信号と同じ情報を含んでいます。

高調波または高調波コンテンツの周波数位置を計算するには、次のアルゴリズムを使用できます。

$$Fharm = N \times Ffund$$
$$IF\ (Fharm = Odd\ Nyquist\ Zone)$$
$$Floc = Fharm\ Mod\ Ffund$$
$$Else$$
$$Floc = Ffund\text{-}(Fharm\ Mod\ Ffund)$$
$$End$$

ここで、N は問題の高調波を示す整数です。

この例をさらに続けて、サンプル レートが 2500MHz、基本信号が 1807MHz の場合を考えると、第 1 ナイキスト ゾーン内の 693MHz に高調波成分が存在し、これを FFT 内でさらに処理できます。

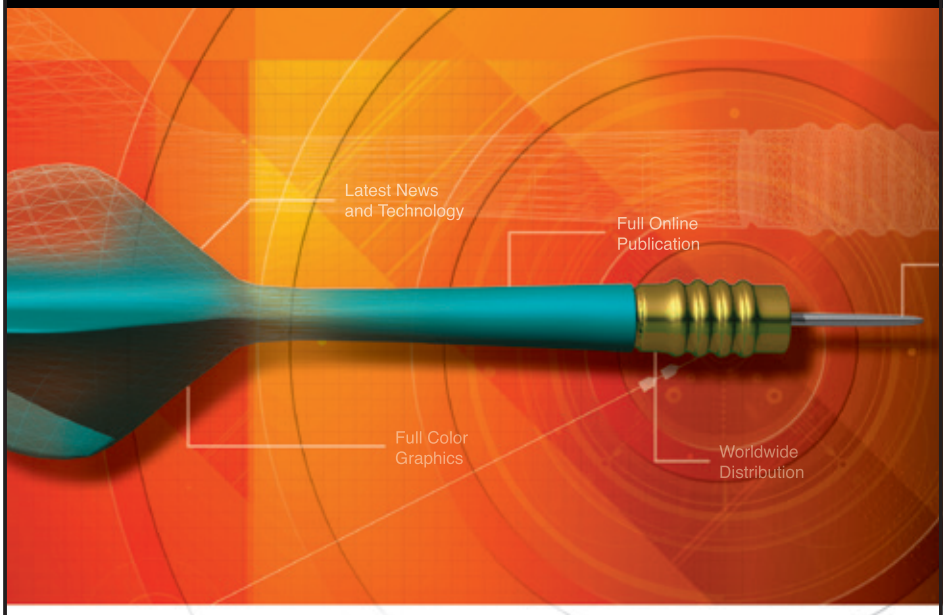
周波数スペクトラムの基本が理解できたら、次に検討するべき重要な要因は、どのようにこれらの ADC および DAC デバイスと FPGA のインターフェイスをとるかということです。ADC からのデータを FS/2 で受信することは不可能です ( 上の例ではサンプリング周波数は 2.5Gbps)。この理由で、高性能なデータ コンバーターは、コンバーターのサンプル レートよりも低いデータ レート ( 通常は FS/4 または FS/2) で動作する、多重化 ( マルチプレクス ) されたデジタル入力および出力を使用します。

FPGA から複数のデータ ストリームでデータを受信したら、次の問題は、DFT を実行する場合に FPGA の内部でどのようにデータを処理するかということです。通信用プロセッサや電波天文学など、多くのアプリケーションで利用されている一般的な手法の 1 つは、図 3 に示すように、結合型または分割型の FFT 構造を使用することです。

このアプリケーションは簡単な FFT よりも複雑になりますが、このような手法によって、処理の高速化が可能となります。

おわかりのように、周波数ドメイン内での作業は、特に IP モジュールを使って時間ドメインと周波数ドメイン間で信号を変換できる場合、思ったほど難しくありません。さらに、高速処理のインプリメンテーションを可能にする多くの手法が利用できます。


GET ON TARGET




**パートナーの皆様、貴社の製品・サービスを Xcell journal 誌上で PR してみませんか？**

Xcell Journal は プログラマブル デジタル システム開発者へ
ザイリンクスおよびエコシステム製品の最新情報をはじめ、
システム／アプリケーションの解説、サービス／サポート情報、
サードパーティー各社の製品情報などをお届けしています。

現在では日本各地の 9,000 名を超える幅広い分野の
エンジニアの皆様にご愛読いただいており、
ザイリンクスの Web サイトから、無償でダウンロード
または iPad 対応デジタル版が購読できます。

貴社製品／ソリューションのプロモーションに非常に効果的なメディアです。

**広告掲載に関するお問い合わせ先**
Xcell Journal 日本語版への広告出向に関するお問い合せは E-mail にてご連絡下さい。
**有限会社 エイ・シー・シー**
**sohyama@acc-j.com**

Xcell PUBLICATIONS

# Marrying SoC Platform Designs with System Generator for DSP

# SoC プラットフォーム デザインと System Generator for DSP を統合

**Daniel E. Michek**
Senior Manager, System-Level Product Marketing
Xilinx, Inc.
*daniel.michek@xilinx.com*

Vivado System Generator ツールは、SoC プラットフォームのデザインに容易に接続でき、ボード インターフェイスとプロセッシング システムを利用できます。

FPGA アプリケーションは急速に進化しており、それに伴って FPGA デザイン フローも進化しています。FPGA はもはや単なるグルー ロジックではなく、独自規格に基づくバックエンド インターフェイスに IP（知的設計資産）コアを接続する信号処理チェーンの中心として使用されるだけでもありません。現在の FPGA は、プロセッサのペリフェラルとして動作するように設計されたハードウェアと高性能 APU 上で実行される高度なソフトウェアを組み合わせた、プログラマブル システム オン チップへと進化しています。これがザイリンクスの All Programmable SoC と呼ばれるアーキテクチャです。

この新しいフローを利用するには、初期の FPGA のようなトップダウンの RTL から、IP コアの開発と ARM® の Advanced eXtensible Interface (AXI) などの標準化された接続を中心とするボトムアップのフローへと、設計手法を切り替える必要があります。カスタム インターフェイスから共通インターフェイスへの進化が進めば、データ パスとプラットフォーム デザイン間の相互作用の検証に費やされる労力を軽減できます。

ザイリンクスの System Generator for DSP も進化しています。Vivado® Design Suite の一部であるこのツールは、Vivado IP インテグレーター ツールで構築されるプラットフォーム デザインに DSP データパスを組み込むことで、新しいボトムアップ設計手法を統合します。この記事では、System Generator を使用したデザイン オートメーションにより、高性能なデザインが、どのようにしてプラットフォーム コネクティビティを利用するかを詳しく説明します。

## All Programmable プラットフォーム フレームワークの構築

新しいデザイン フローでは、最初に、データ パスを収容するプラットフォーム フレームワークを定義します。Vivado Design Suite はボード認識型ツールなので、設計者は利用可能なボード オートメーションを利用して新しいプラットフォーム デザインを構築します。

図 1 に示したプラットフォーム デザイン（すなわち、プラットフォーム フレームワーク）は、プロセッサ レベルおよびボード レベルのインターフェイスと、それらのインターフェイスを結合するロジックの基本的な集合体です。筆者らはこのプラットフォーム フレームワークをシステム レベルのデザインの基礎（シェル）として使用します。データ パスについては自由に設計する余地があります。ブロック オートメーションとコネクティビティ オートメーションは、プロセッシング システムを IP ペリフェラル経由でボード レベルのインターフェイスに接続します。IP カタログにパッケージされた DSP データ パスまたはソフトウェア アクセラレータは、ザイリンクスの設計者支援オートメーションを利用して、プロセッサと（拡張機能による）外部デバイスへのインターフェイスで構成されるプラットフォーム フレームワークに容易に接続できます。

## インポート可能な IP コアとして データ パスを作成

筆者らの最終的な目標は、データ パスが All Programmable プラットフォーム フレームワークにアクセスできるようにすることです。デザインをゼロから始めたければ、標準化されたインターフェイスを使用してデータ パスを作成できます。図 2 に示すように、ゲートウェイ ポートを AXI4-Lite インターフェイスとして容易にマークするか、あるいは単に、Simulink® ダイアグラム上で AXI4-Stream のような標準接続ポートに合わせてポートの名前を付けておけば、System Generator が、Vivado IP カタログ用にデザインをパッケージするときに追加のロジックをデザインに付加し、共通の信号をインターフェイスに集めてくれます。

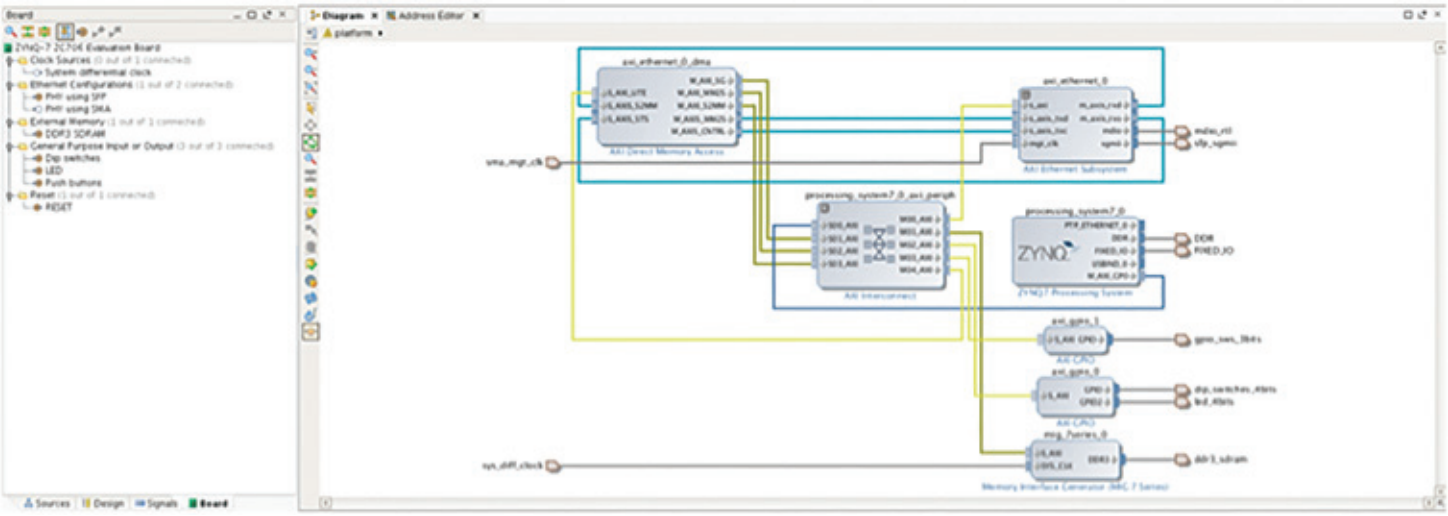


図 1 – プロセッシング システムをボード レベルのインターフェイスに接続するプラットフォーム フレームワークの例

しかし、新しい手法では、プラットフォーム フレームワークを使用して、All Programmable デザインに統合されるプラグインを調整できます。オートメーションを使用して、どのインターフェイスがプラットフォーム デザイン内に存在するか、どのインターフェイスがボードに関連付けられ、どのインターフェイスが DSP データパス用のプラグインを作成するかを決定します。目標は、このデータ パスを、プラットフォーム フレームワークに接続される IP コアへと変換することです。したがって、ボード レベルのインターフェイスにではなく、標準化された AXI インターフェイスに焦点を合わせる必要があります。ボード レベルで関連付けられていない各インターフェイスは、System Generator ゲートウェイに変換されます。これらのゲートウェイは、System Generator 環境の内部でシンプルな信号として機能する一方、プラットフォーム デザインが IP カタログにエクスポートされるとき、プラットフォーム デザインに接続される AXI インターフェイスを生成します。

1 つの例として、AXI4-Lite インターフェイスは、Vivado Design Suite (Vivado ツール スイート) にエクスポートされるとき、共通してアドレス指定できるレジスタ インターフェイスを共有する、互いに独立した読み出し信号と書き込み信号を作成します。簡単なコピーと貼り付けの操作で、同じインターフェイスを介してアドレス オフセットの位置でプロセッサから利用可能な、直接レジスタの数を増やすことができます。同時に、これらのレジスタの読み出し / 書き込み用のソフトウェア ドライバー API が自動的に生成されます。

AXI4-Stream インターフェイスがプラットフォーム デザインから利用可能である場合、System Generator は、適合するゲートウェイをモデルに追加します。AXI4-Stream インターフェイスは極めて柔軟性が高く、多くの信号を含んでいます。ACLK は、このインターフェイスに関連付けられるクロック ソースですが、この信号には、データ パスのこの部分の抽象化されたシステム クロックとしての直接の意味が与えられます。TVALID は、インターフェイスが有効になったことを示す信号です。その他の信号はオプションです。System Generator は発信元のストリーム インターフェイス内に存在する信号をモデルに追加しますが、設計者は内部の要件に合わせて信号を削除または追加できます。

図 2 に示したモデルでは、筆者らのデータ パスは S_AXIS インターフェイス上の TDATA (インターフェイス上に送信されるデータ) および TVALID のみを考慮に入れていることがわかります。IP インテグレーター内では、デフォルト値で信号接続が駆動されるので、不要な信号を削除するために、このモデルに使用されないゲートウェイをコメント アウトします。

AXI4-Lite 信号および AXI4-Stream 信号は、ボトムアップ手法によって容易にシミュレートして検証できます。AXI4- Lite インターフェイスは、多数の Simulink ブロックからソースおよびシンクされるシンプルなゲートウェイとしてモデル化されます。これはゲートウェイの一方の側からもう一方の側へのデータの受け渡しをシンプルに抽象化したものです。同様に、AXI4-Stream インターフェイスは、1 つの IP コアからもう

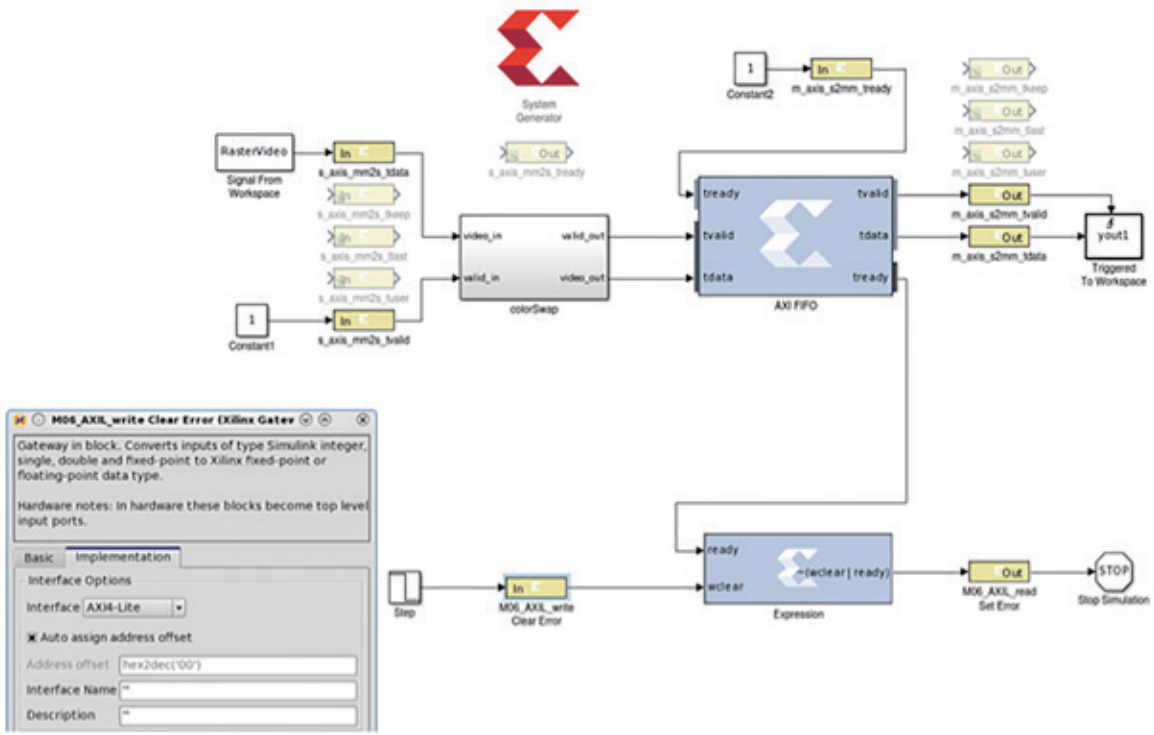



図 2 – AXI4-Lite および AXI4-Stream インターフェイスに変換される自動化ゲートウェイ

1つの IP コアへデータを渡すための簡単なハンドシェイク規則に従う信号の単なる集合体です。

筆者らのシミュレーション モデリングで問題となるのは、インターフェイス上で使用されるオプションのポートだけです。1 サイクルごとにデータを受け入れ、データ パスを介して中断なくデータを処理する場合は、TREADY ハンドシェイク信号は不要です。この簡素化されたモデルは、Simulink の Signal トークンから Workspace トークンへ、TDATA ゲートウェイを介してベクトルの各要素を送信します。完全なハンドシェイクが必要な場合は、図 2 の M_AXIS インターフェイスに示すように、AXI4-Stream FIFO を使用してそれをモデル化し、データをバッファーに入れます。

調整オートメーションは、プラットフォーム フレームワークに対してインターフェイスする IP コアの作成の出発点として機能します。しかし、System Generator は柔軟性が高く、AXI インターフェイスの一部 ( または全体 ) の追加や削除が可能です。最終的なデザインでは、データ パスが IP コアに変換され、複数のシステム レベル デザインに再利用できるようになります。

ロジックを追加した後、最後の手順として、System Generator for DSP で構築したデータ パスを Vivado IP カタログに逆にエクスポートします。この操作により、RTL を使った作業または IP インテグレーター内での作業でも、インターフェイスを容易に接続できます。さらに、SDK 内で使用するドライバーを生成し、ゴールデン テスト ベクトル データによるシミュレーション モデルを IP コアに付加します。また、DSP データ パスを作成したときにプラットフォーム フレームワークの知識を事前に得ていたため、図 3 の完成したシステムに示すように、モデル化とプラットフォーム デザインの統合が自動的に行えます。

## シミュレーション リスクの軽減

ハードウェア アクセラレータ、DSP データ パス、またはカスタム ロジックを統合した完全なシステム オン チップを生成することは容易ではありません。ボトムアップ方式のシミュレーションによってデータ パスが予想どおり機能することを確認するには、大きなリスクが伴います。また、データ パスをサポートするプラットフォーム インターフェイスの帯域幅を確実に維持するには、同じくらい手間がかかります。

標準化されたインターフェイスを利用することで、シミュレーション リスクを軽減する IP コアを開発できます。リスクが軽減されるのは、インターフェイス レベルの相互作用が抽象化されるため、設計者は内部データ パスの検証に専念できるからです。また最後に、ボード、ブロック、コネクティビティのスマートなオートメーションを利用することで、ユーザーの要件を満たし、カスタム データ パスを統合した、プラットフォーム ベースのシステムを生成できます。

System Generator for DSP の詳細は、http://japan.xilinx.com/products/design-tools/vivado/integration/sysgen.html を参照してください。その他不明な点や質問等ありましたら、筆者 Daniel Michek (Tel : (858) 207-5213、e-mail : daniel.michek@xilinx.com) までお問い合わせください。

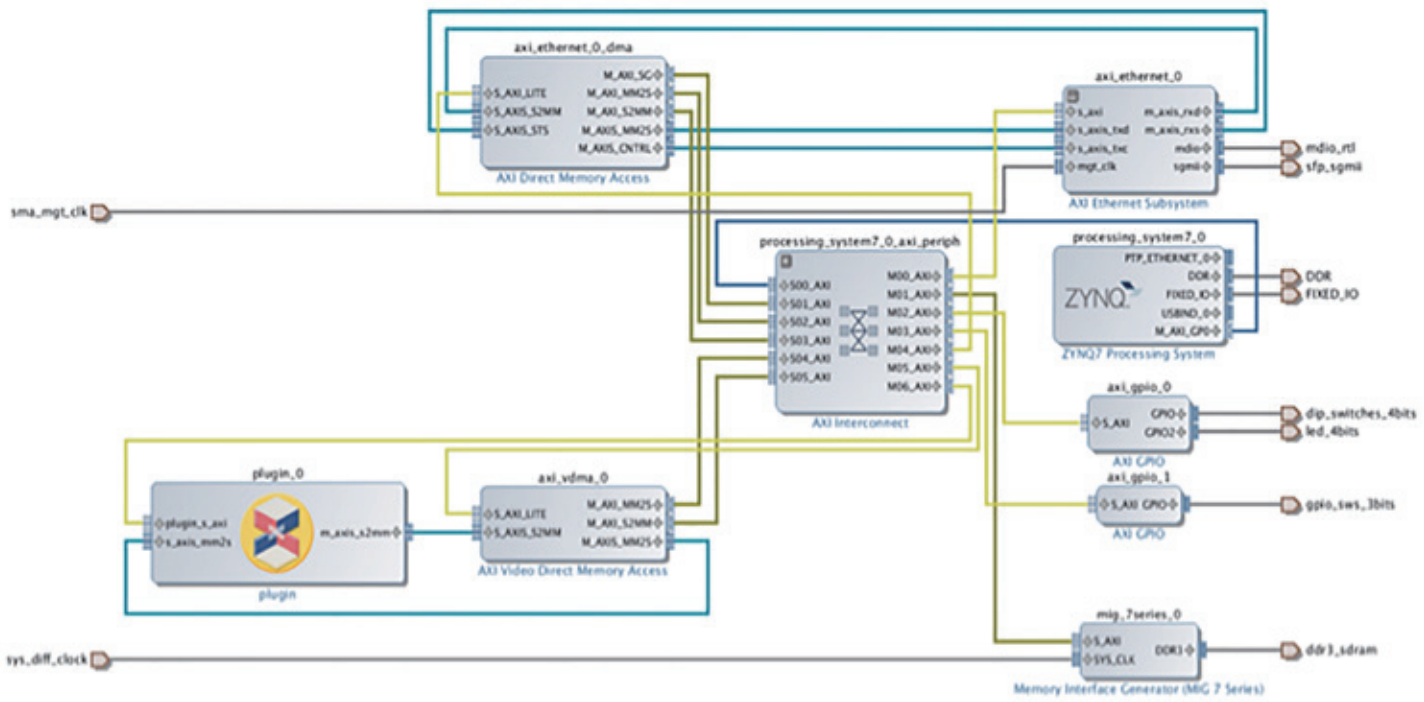



図 3 – DSP データ パスをプラットフォーム フレームワークに接続するプラットフォーム ベースのシステム

# Rethinking Digital Downconversion in Fast, Wideband ADCs

# 高速ワイドバンド ADC 内のデジタル ダウンコンバージョンの再検討

**Ian Beavers**
Contributing Technical Expert
Analog Devices
*ian.beavers@analog.com*


ワイドバンド GSPS ( ギガサンプル / 秒 ) アナログ / デジタル コンバーター (ADC) は、高速データ収集システムに性能面で多くのメリットをもたらします。これらの ADC は、高いサンプル レートおよび入力帯域幅で広帯域の周波数スペクトラムを可視化します。ただし、アプリケーションによっては、ワイドバンド フロント エンドを必要とするものと、ナローバンド スペクトラムへのフィルタリング / チューニング機能を必要とするものがあります。

ナローバンドのみを必要とするアプリケーションで、ADC がサンプリング、処理、電力の消費を最大限に行ってワイドバンド スペクトラムを送信するのは、基本的に非効率です。データ リンクがザイリンクス FPGA 内の大規模な高速トランシーバー バンクを消費し、それ以降の処理でワイドバンド データのデシメーションとフィルタリングを行うと、システムに無用な負担をかけることになります。その代わりに、ザイリンクス FPGA のトランシーバー リソースを上手に割り当てて、より狭いその被測定帯域を受信し、複数の ADC からのデータをチャネル化できます。その後のフィルタリングは、周波数分割多重 (FDM) アプリケーション用の FPGA の多相フィルター バンク チャネライザー内で行えます。

GSPS ADC の高性能化により、デジタル ダウンコンバージョン (DDC) 機能を信号チェーンの上流に配置し、ザイリンクス FPGA ベースのデザイン ソリューションの ADC 内に配置することが可能になります。この手法は、高速システムの設計者に複数の新しい設計オプションを提供します。ただし、これは比較的新しい ADC の機能なので、GSPS ADC 内の DDC ブロックの動作に関して、技術者がデザインに関連する疑問を感じることがあります。この記事では、この新しい手法を安心して設計に使えるように、よくある一般的な疑問に回答していきます。

## デシメーションとは

簡単に定義すると、デシメーションとは、ADC 出力サンプルの周期的な一部だけを観察し、その他のサンプルを無視する手法です。その結果、ダウンサンプリングによって ADC のサンプル レートが実質的に下がります。たとえば、ADC 出力を decimate-by-M モードにすると、M 番目ごとのサンプルのみが出力され、中間のサンプルはすべて廃棄されます。サンプル数が M の倍数に達するたびに、この操作が繰り返されます。

サンプル デシメーション単独では、ADC のサンプル レートを実質的に下げるローパス フィルターとして機能するだけです。周波数変換やデジタル フィルタリングを使用しない場合、デシメーションは単に、周波数ドメイン内で基本信号の高調波および他のスプリアス（偽の）信号をフォールド（折り重ねる）します。

## DDC の役割

デシメーションだけでは、帯域外の信号がフォールドされるのを防ぐことはできません。DDC はどのようにこれを実現するのでしょうか。

DDC がもたらす性能面のメリットをフルに生かすには、デシメーション機能と組み合わせて使用されるフィルター / ミキサー コンポーネントをデザインに組み込む必要があります。デジタル フィルタリングは、デシメーション比によって設定され定義される狭い帯域から、帯域外ノイズを実質的に除去します。DDC 用の標準的なデジタル フィルターは、有限インパルス応答 (FIR) フィルターとしてインプリメントされます。フィードバックが存在しないため、このフィルターは過去の入力にのみ影響されます。FIR フィルターの通過域は、デシメーション後のコンバーターの実効周波数スペクトラムの幅と一致している必要があります。

## DDC フィルターの幅の選択

DDC のデシメーション比は、通常は 2 のべき乗である整数係数 (2、4、8、16 など) に基づきます。しかし、デシメーション係数は、実際には DDC アーキテクチャに基づいて分数デシメーションを含む任意の比に設定できます。分数デシメーションの場合、有理分数の比が得られるように、通常はデシメーションの前に補間演算ブロックが必要です。

理想的には、デジタル フィルターはデシメーション周波数の帯域幅と完全に一致し、その帯域外のすべての信号をフィルタリングするはずです。しかし、実際の実効フィルター幅は、デシメーション比の全帯域幅と完全には一致しません。したがって、フィルターの幅は、デシメーション周波数の何 % かになります (85%、90% など)。たとえば、デシメーション係数 8 のフィルターの使用可能な帯域幅は、実際にはサンプル レートを 10 で割った値 ( すなわち、fs/10) になることがあります。DDC フィルタリング ステージは、低域通過フィルターのリップルと高域阻止フィルターのエイリアスの除去手段を提供する必要があります。

## 固定周波数と可変周波数

次の問題は、DDC フィルターの周波数は固定されているか、すなわち、特定の被測定帯域上での DDC フィルターのチューニングとセンタリングが可能かということです。

ここまで DDC のデシメーション ステージとフィルタリング ステージについて説明してきました。しかし、この説明は、希望の周波数が DC からのフィルター通過域の範囲内に入る場合にのみ有効です。通過域を外れている場合は、周波数スペクトラムの他の部分に

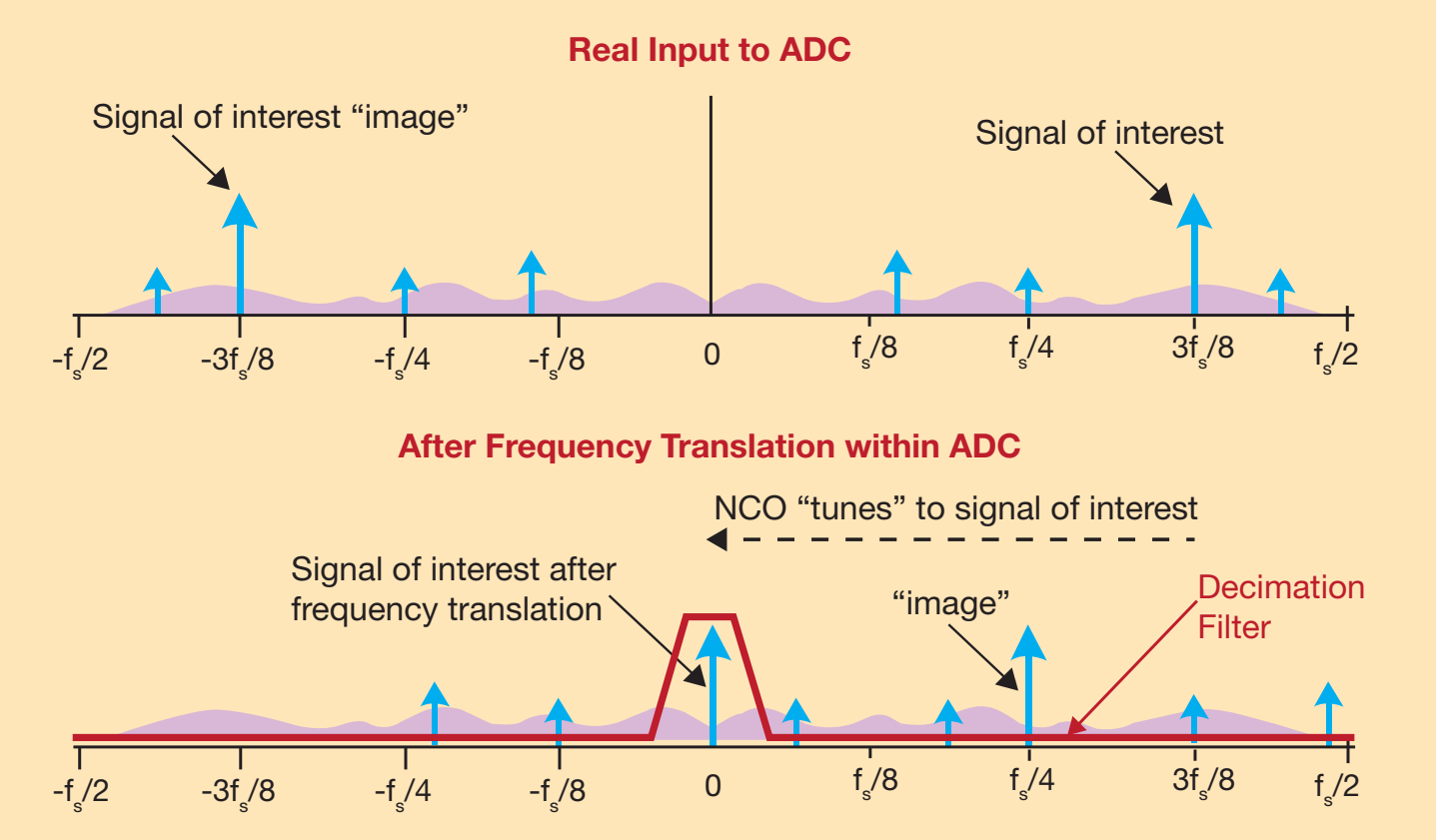



図 1 – ローパス フィルターを使用した周波数変換と NCO により、被測定周波数の通過域フィルターを効果的に実現できます。周波数プランニングにより、不要な高調波、スプリアス信号、イメージを帯域外に配置できます。

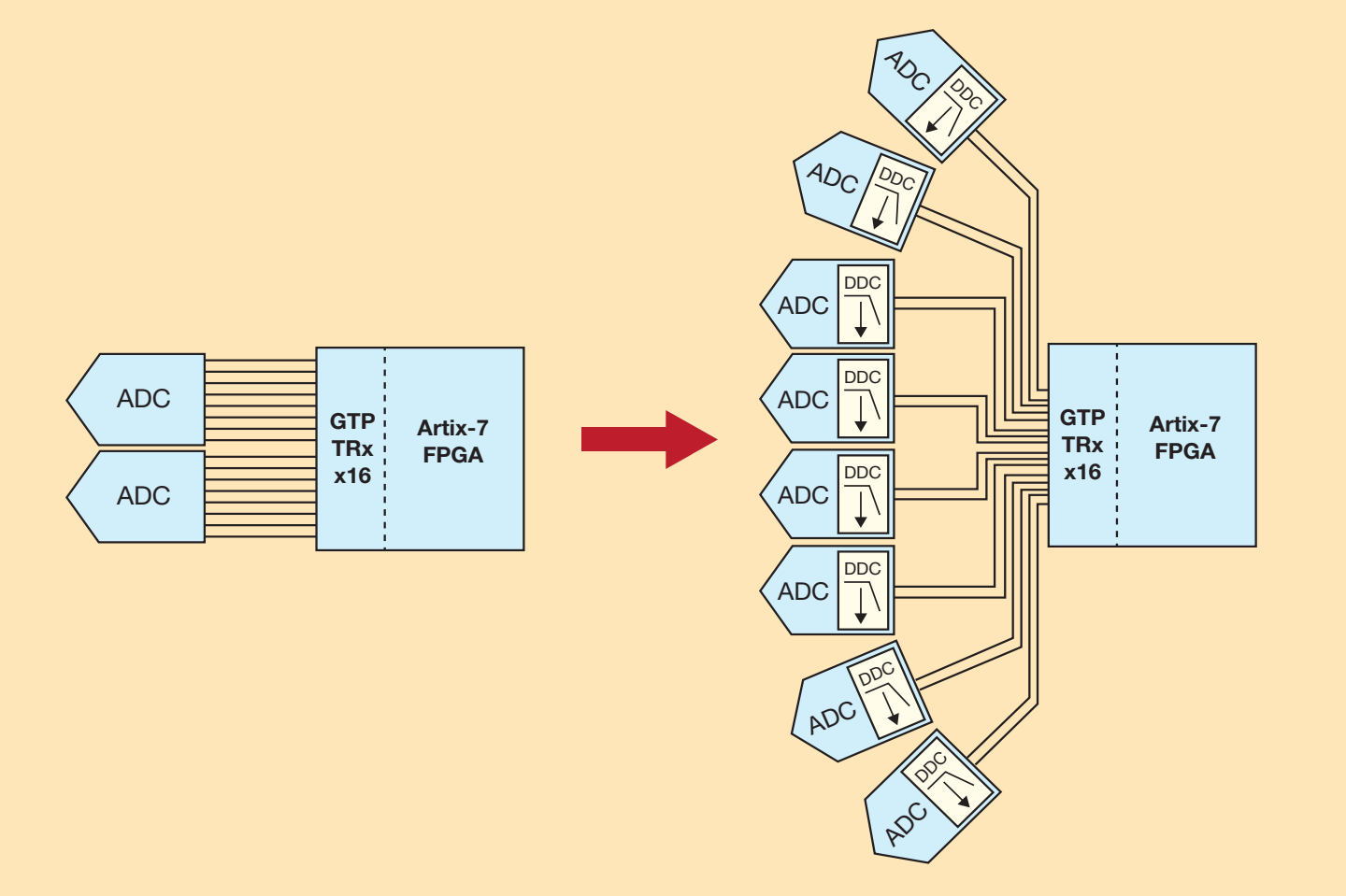



図 2 – デシメーション係数 8 の DDC を使用して、ザイリンクス Artix-7 の 16 個の GTP 6.6Gbps トランシーバーはそのままで、8 個の ADC (各 ADC が 2 レーンの JESD204B 上にデシメーション済みの I/Q データを出力) をサポートする場合と、2 個の ADC (各 ADC が 8 レーンを介して全帯域幅を出力) をサポートする場合の比較。

合わせてフィルターをチューニングし、被測定信号を観察する必要があります。この狭帯域は、数値制御オシレーター (NCO) によって第 1 または第 2 ナイキスト ゾーン内でチューニング可能です。NCO は、ワイドバンド スペクトラムの異なる部分に合わせたフィルター帯域のチューニングとミキシングの方法を提供します ( 図 1)。

デジタル チューニング ワードは、そのデジタル チューニング ワード内で使用される、被測定帯域のミキシングを可能にするビット数によって定義される周波数配置分解能を備えた、サンプル レートの分数分周器を提供します。チューニング ワードは、必要な箇所にフィルターを配置するためのチューニング範囲と分解能を持ちます。標準的な NCO チューニング ワードの分解能は、サンプリングされる周波数の 2 つのナイキスト帯域にわたる最大 48 ビットまで可能であり、ほとんどのアプリケーションにはこれで十分です。

NCO には、ミキサーが付属しています。アナログ直交ミキサーとよく似た動作をするこのデバイスは、NCO の周波数をローカル オシレーターとして使用することにより、実数および虚数入力信号のダウンコンバージョンを実行します。

DDC フィルターの後は周波数変換ステージに進みます。被測定キャリア バンドが DC にチューニング ダウンされた後、DDC フィルターは、サンプル レートを実質的に下げると同時に、チューニングされた被測定帯域の周囲の不要な隣接搬送波に対する十分なエイリアス除去手段を提供します。

実数入力信号をベースバンドにミックスダウンする場合、負のイメージのフィルタリングのために 6dB の信号損失が発生します。また、NCO によって追加の小さな挿入損失が発生します。ベースバンドにミックスダウンされた実数入力信号の総損失は通常、6dB を多少上回ります。NCO により、入力スペクトラムを DC に合わせてチューニングし、それ以降のフィルター ブロックによって入力スペクトラムを実質的にフィルタリングしてエイリアシングを防ぐことができます。また DDC は、独立して制御されるデジタル ゲイン ステージを含むことがあります。ゲイン ステージにより、システムは、+6dB またはそれ以上のゲインを実現し、出力ビットのフル スケールの範囲内で信号のダイナミック レンジをセンタリングできます。

## プロセッサ間の割り込み

ADC サンプルのデシメーションにより、最終的に放棄される不要な情報を信号チェーンの下流に送信する必要がなくなります。このデータが除去されるため、ADC のバックエンドに必要とされる出力データ帯域幅は削減されます。この削減された量は、I データ出力と Q データ出力の両方からのデータの増加によって埋め合わされます。たとえば、I データと Q データの両方を出力するデシメーション係数 16 のフィルターでは、ワイドバンド出力データが 8 分の 1 に減少します。

このようにデータ レートを最小限に抑えると、ADC からの出力 JESD204B レーン数を減らすことで、システム レイアウトの複雑性が抑えられます。ADC 出力帯域幅の削減により、通常では実現できないようなコンパクトなシステム デザインが可能になります。たとえば、システムの電力とサイズの制限のためにボード上で 1 個の FPGA しか使用できない場合、DDC を使用しないと、サポートされる高速シリアル トランシーバーの数によって ADC の数が制限されることがあります。

このシステム内で狭帯域のみを観察する場合は、ADC 内のデシメーションによって、この制限を取り払うことができます。デシメーション係数 8 の DDC を 1 個使用して、ADC の出力帯域幅を 2 つの出力データ レーンのみに減らすことにより、同じザイリンクス Artix®-7 FPGA システムで 4 倍の ADC をサポートできます。特にこの場合は、Artix-7 FPGA 内の既存の 16 個の GTP トランシーバーをそのまま使用して、DDC を使用する 8 個の ADC を設計できます ( 図 2)。この方法で、ザイリンクス FPGA のリソースを、一連の FDM チャネル用のマルチチャネル デジタル レシーバーとして効率的に利用できます。

## DDC フィルターによる SNR および SFDR への影響

次に検討するべき問題は、DDC フィルターがオンの状態とオフの状態で、信号対ノイズ比 (SNR) とスプリアスフリー ダイナミック レンジ (SFDR) のアナログ性能がどのように変化するかということです。

コンバーターのワイドバンド ノイズが除去され、狭帯域スペクトラムのみが観察されるため、観察されるノイズに対する信号電力の比は大きくなると予想されます。ADC のダイナミック レンジは、フィルターの通過域内では向上するでしょう。DDC の使用による SNR の向上は、ワイドバンド スペクトラムのデシメーションとフィルタリングの基本的な利点の 1 つです。

DDC によるデジタル フィルタリングを使用して、より小さな帯域幅の外側のノイズをフィルタリングできます。この場合、ADC の SNR の計算には、フィルタリングされたノイズの処理ゲインを表す、このフィルタリング用の補正係数を含める必要があります。完璧なデジタル フィルターを使用した場合、帯域幅が 2 分の 1、4 分の 1、8 分の 1、16 分の 1 ... に削減されるたびに、フィルタリングされたノイズによる処理ゲインは +3 dB ずつ増えます。

*理想的 SNR (処理ゲインを含む) = 6.02*N+1.76 dB+10log10(fs/(2*BW))*

DDC を使うことの明確な利点は、基本信号の高調波を被測定帯域の外側に配置できることです。適切な周波数プランニングを行えば、デジタル フィルタリングによって DDC の狭い帯域内で高調波が検出されなくなり、システムの SFDR 性能が向上します。

狭い帯域幅で十分なシステムでは、DDC はワイドバンド ノイズを除去することによって ADC 処理ゲインを提供します。これにより、被測定帯域幅内で検出される信号対ノイズ比が向上します。もう 1 つのメリットは、適切な周波数プランニングにより、通常は優勢となる基本信号の 2 次高調波および 3 次高調波が、チューニングされた被測定帯域の外側に配置され、デジタル方式でフィルタリングされることです。これにより、システムの SFDR が向上します。

サンプリングの原理によると、高調波やシステムの高次スプリアス信号は、各ナイキスト帯の周辺にフォールドバックします。これは DDC にも当てはまり、不要な 2 次または 3 次高調波が通過域にフォールドバックし、SFDR を低下させる可能性があります。したがって、このようなサンプリングの問題を回避するには、DDC 通過域フィルターの幅と NCO のチューニング位置を考慮したシステム周波数プランをインプリメントする必要があります。

## 外部フィルターの必要性

内部の DDC を使用するシステム ADC は、DDC フィルタリングを使用しない場合に通常行われているように、追加のアナログ フィルターを使用できます。ワイドバンド アプリケーションでは、DDC は ADC のフロントエンドで必要とされるフィルタリングを多少軽減します。

DDC 内のデジタル フィルタリングは、フィルタリング処理の一部を実行し、フロントエンドの厳格なアンチエイリアス アナログ フィルターに必要となる処理を軽減します。一方、ワイドバンド フロント エンドは複数の用途に DDC を使用できるようになり、複数の帯域を同時に観察することや、NCO で被測定帯域を掃引し、変化する入力信号を検出することが可能になります。

## ADC による 複数の DDC の利用

FPGA を使用した内部デジタル ダウンコンバージョンを検討している技術者にとって、最後の疑問は、ADC は 1 つの DDC しか利用できないのかということです。答えは「いいえ」です。実際に、複数の帯域の観察が可能です。

ADC 内の複数の DDC に対して、ナイキスト ゾーンの別々の帯域にチューニングされた、各 DDC 専用の NCO を実装できます。この方法で、複数の周波数帯を同時に観察し、システムの FPGA トランシーバー数とデシメーション ブロックにかかる負担を軽減できます。これらのリソースは、FDM システム用に複数の ADC をチャネル化するなど、他の処理に再割り当てできます。

高速 ADC の処理能力の向上により、信号チェーンの上流に DDC 機能を実装できるようになりました。ワイドバンド ナイキスト レート ADC の全帯域幅を使用する必要がないシステムでは、DDC 処理によって不要なデータとノイズをフィルタリングできます。これにより、信号収集の SNR と SFDR が向上します。帯域幅の低減により、Artix-7 などの FPGA のトランシーバーに対するデータ インターフェイスの負担が軽減され、複雑な信号収集システムのデザインが容易になります。

# ザイリンクス トレーニング カリキュラムパス

XILINX

**基本のFPGAアーキテクチャ E-Learning**
- スライス および I/Oリソース
- メモリ および クロッキングリソース
- Architecture Wizard および Foorplan Editor

開発言語

| VHDL 基礎 | Verilog 基礎 | System Verilog での設計 |
|---|---|---|
| VHDL 中級 | Verilog 中級 | System Verilog での検証 |

Vivado デザインツールフロー

VivadoでのFPGA設計導入

VivadoでのFPGA設計実践

- 7シリーズデザイン
  - Chip Scope Pro ツールを使用したデバッグ法
  - FPGA 低消費電力設計のコツ
  - アナログミックスシグナル(AMS)設計
    - Vivado でのアドバンスド FPGA設計
      - パーシャルリコンフィギュレーション
- コネクティビティデザイン基礎
  - メモリインターフェイス設計
    - マルチギガビットシリアルIO
- DSPデザイン基礎
  - Simlink基礎(MathWorks社開催)
    - SystemGenerator を使用したDSPデザイン
      - Cコードベース設計 : Vivado HLS を使用した高位合成
- エンベデッドデザイン基礎
  - Zynq システムアーキテクチャ
    - Zynq エンベデッドシステム開発
      - Zynq エンベデッドシステムソフトウェア開発
        - エンベデッドマイクロブレーズ Linux開発

入門 / 初級 / 中級 / 上級

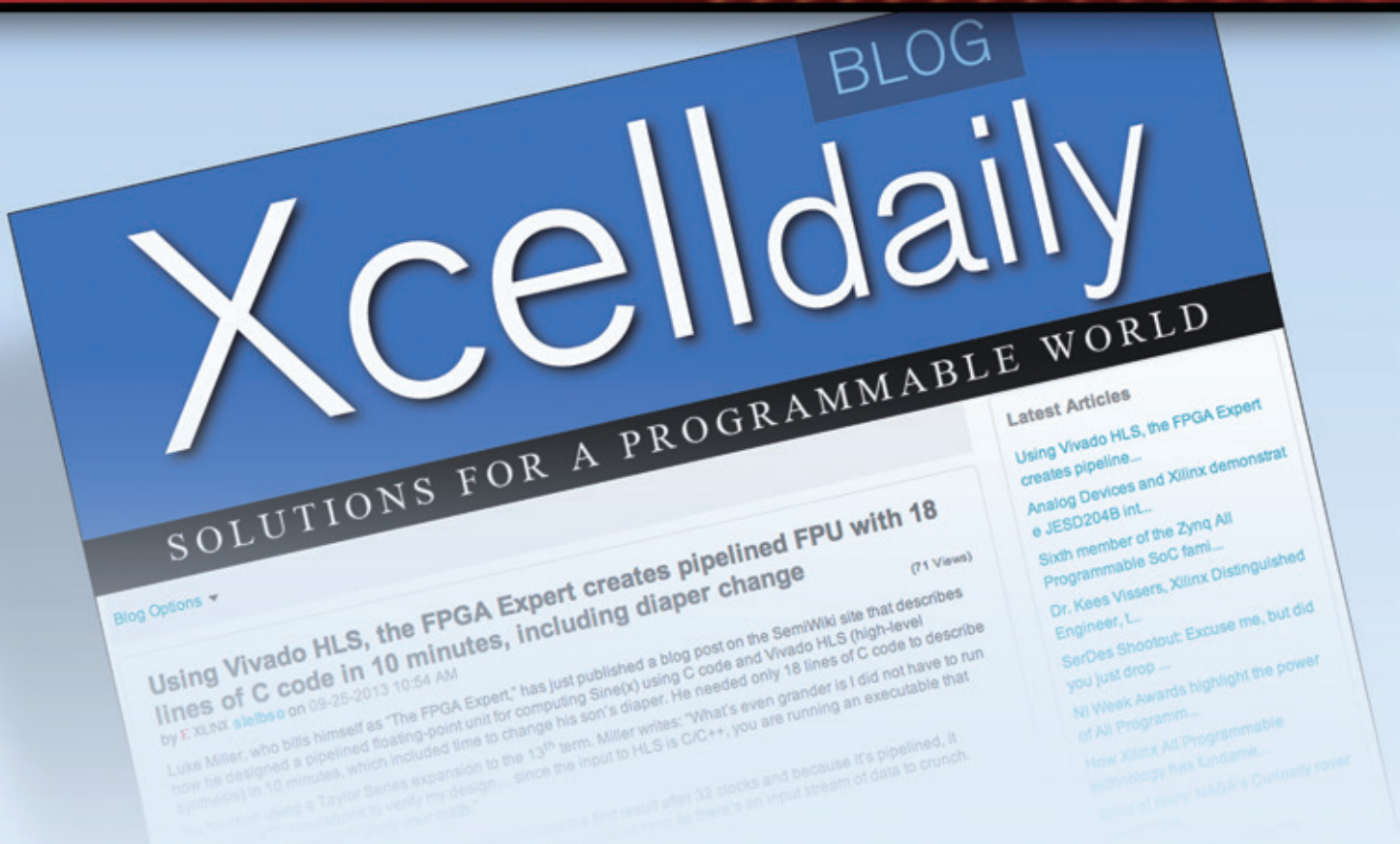
ザイリンクス販売代理店 / 認定トレーニングプロバイダ **オリジナル トレーニング**

オリジナルトレーニングの内容およびスケジュールは、各社の Web サイトをご覧ください。

| | | |
|---|---|---|
| AVNET | アヴネット(株) | www.avnet.co.jp/training.aspx |
| 新光商事 Shinko Shoji Co.,Ltd. | 新光商事(株) | xilinx.shinko-sj.co.jp/training/index.html |
| PALTEK | (株)パルテック | www.paltek.co.jp/seminar/index.htm |
| hdLab | (株)エッチ・ディー・ラボ | www.hdlab.co.jp/web/x500x/ |

詳細とご登録はこちらから **Japan.xilinx.com/training/**

# Xcell Journal を拡充。新たに Daily Blog を追加




ザイリンクスは、数々の受賞歴がある Xcell Journal をさらに拡充し、エキサイティングな Xcell Daily Blog（英文）を始めました。このブログでは、コンテンツを頻繁に更新し、技術者の皆様がザイリンクスの製品とエコシステムの多岐にわたる機能が活用でき、All Programmable システムおよび Smarter System の開発に役立つ情報を提供します。

## Recent（最近の記事）

- *Adam Taylor's MicroZed Chronicles Part 88: SDSoC Part 4—a look under the hood*
- *How do you design backplanes for 25+ Gbps operation? Teraspeed's Scott McMorrow tells you how in two videos*
- *Adam Taylor's MicroZed Chronicles Part 87: Getting SDSoC up and running Part 3*
- *Warning! Only days left before Xcell Journal's latest caption contest deadline. Your prize: a Zynq-based Digilent ZYBO dev board.*
- *Hyperspectral GigE video cameras from Photonfocus see the unseen @ 42fps for diverse imaging applications*

ブログ : www.forums.xilinx.com/t5/Xcell-Daily/bg-p/Xcell

XILINX
ALL PROGRAMMABLE™