拡張ユニバーサルボード／拡張FPGAボード入門

T-Engineによるハードウェア開発

T-Engineフォーラムでは、組込み向けリアルタイムOSであるT-Kernelの「強い標準化」により、ミドルウェアの流通を可能にし、組込みシステムの開発効率向上を目指してきました。T-Engineは、T-Kernelを用いた組込みシステムの標準開発プラットフォームとして規格化されました。T-Engineには、携帯情報機器向けの標準T-Engineと、より組込み向けに小型化したμT-Engineがあり、すでにSH、M32R、FRなどの国産のCPUコアをはじめとして、ARMやMIPSといった世界的にシェアの高いCPUコアを用いた製品が数多く製品化されています。

標準T-Engine、μT-Engineには、メモリやUSBやカードインタフェースなどの組込みシステムに必要なインタフェースを備えていますが、最終製品に応用したり、実験に用いる場合には、標準T-Engine、μT-

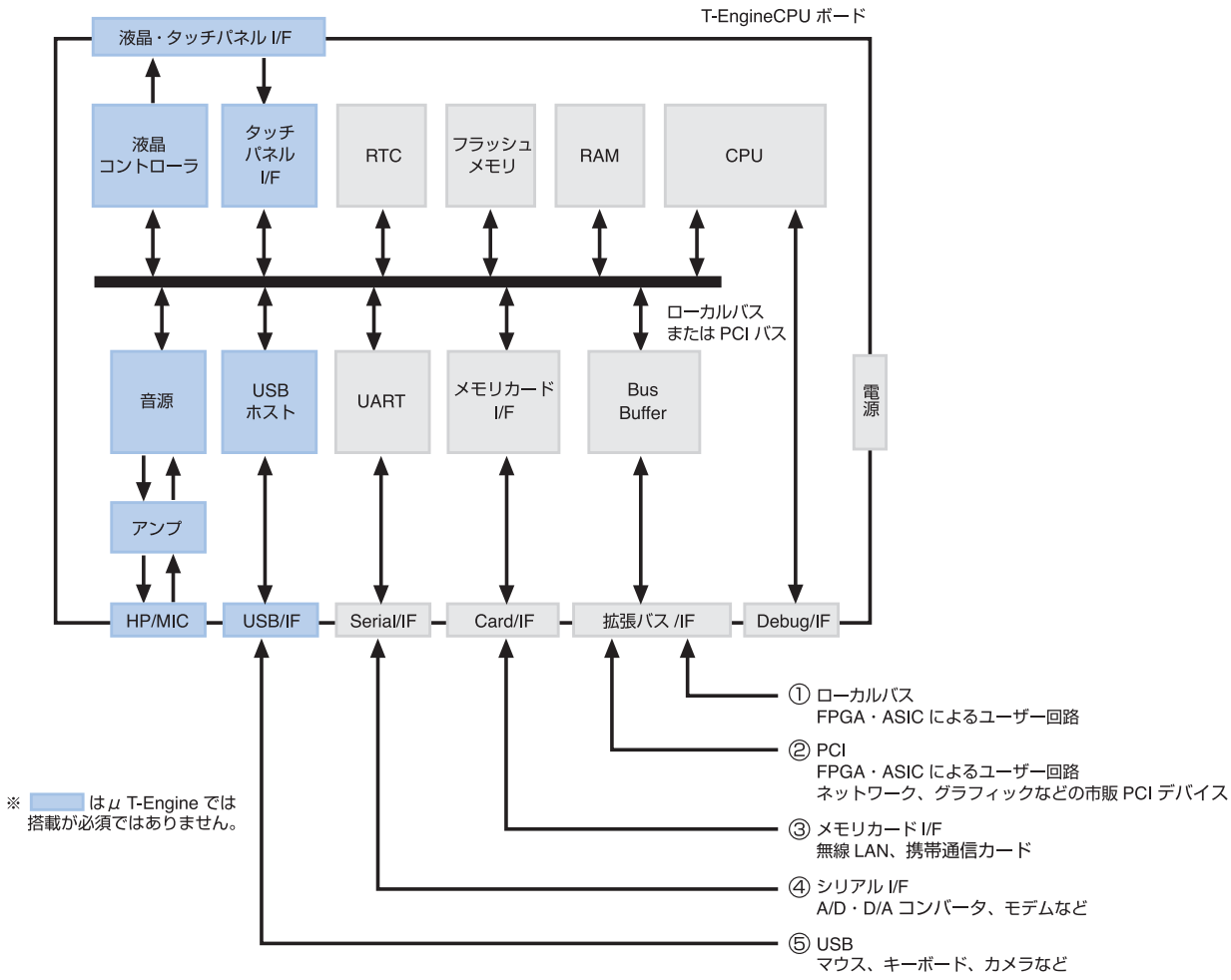

図1　T-Engineのハードウェアと拡張インタフェース

Engineを中心としてユーザー回路を接続したシステムを構築する必要があります。

T-Engineを中心とした組込みシステムの試作用に、拡張コネクタに接続する拡張ユニバーサルボードや機能拡張FPGA開発ボードが開発されています。本章では、これらを使用したハードウェアの設計手法について説明します。

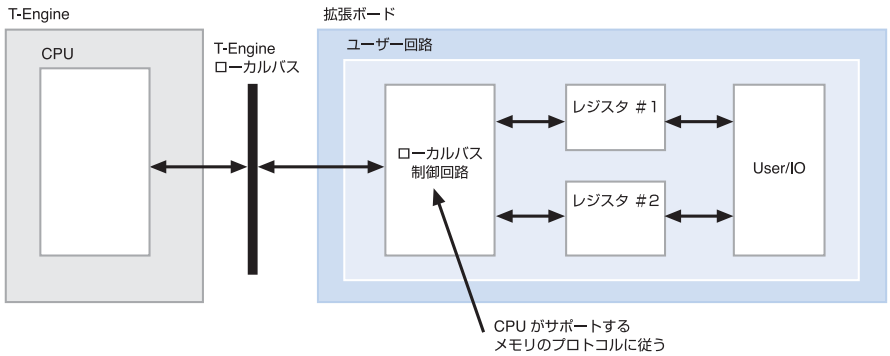

図2　ローカルバスを用いた拡張ボードの構成例

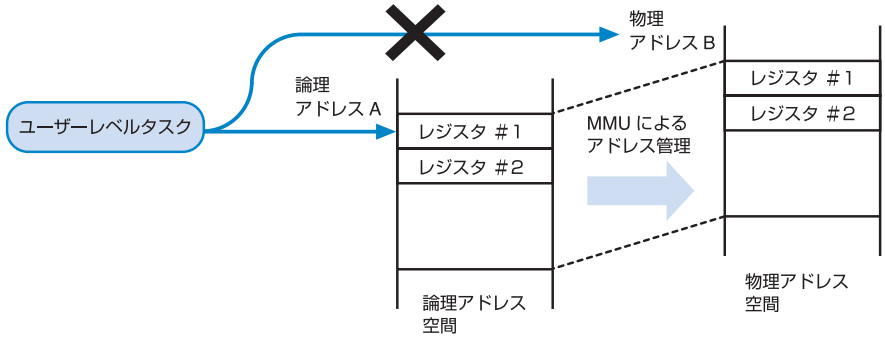

図3　ユーザーレベルのタスクは物理アドレス空間を直接アクセスできない

T-Engineをハードウェア開発に用いる利点

●ソフトウェア開発工数の削減

CPUを中心とした組込みシステムをフルスクラッチで設計する場合や、市販のワンボードマイコンを使用してハードウェアを開発する場合、これらで動作するOSをポーティングする必要があります。また、ミドルウェアやデバイスドライバを新規に開発しなければなりません。

しかし、T-Engineを使用すれば、最初からT-Kernelが動作する状態で提供されますので、OSのポーティング作業は不要です。また、流通しているミドルウェアを利用することにより、開発工数を削減することが可能です。さらに、T-Kernelはデバイスドライバのダイナミックローディングが可能であるため、新しく追加したハードウェアのデバイスドライバを独立モジュールとして開発・ロードすることが可能です。

●豊富な周辺インタフェース

図1に標準的なT-Engineのインタフェース構成を示します。

標準T-Engineは、ユーザーがハードウェアを拡張できるインタフェースとして、シリアルインタフェースやUSBホスト機能、メモリカードインタフェース、拡張コネクタを備えます。μT-Engineでは、メモリカードインタフェースがコンパクトフラッシュインタフェースやSDカードインタフェース等のより小型なインタフェースになっており、USBホスト機能等は必須ではありません。要求性能や回路規模に応じて、これらのインタフェースを選択し、ハードウェアを拡張することが可能です。

さらに、標準T-Engineにはスイッチやタッチパネル付き液晶ディスプレイなどのユーザーインタフェースを備えています。これらを使用しないシステムであっても、デバッグに使用することでデバッグ効率を向上させることが可能です。

拡張コネクタによるハードウェア拡張

●ローカルバスによるハードウェア拡張

拡張コネクタには、T-Engineに搭載されたCPUのローカルバスが出力されています。図2にローカルバスに接続したユーザー回路の概略を示します。

CPUのローカルバスは、メモリ空間またはI/O空間にアドレスマッピングされるデバイスを接続するためのインタフェースです。ローカルバスに接続される最も一般的なデバイスはメモリです。したがって、ユーザー回路は、そのCPUがサポートするメモリのプロトコルに従うことで、T-EngineのCPUからアクセスが可能になります。

T-KernelはMMUによるメモリ管理をサポートしています。メモリの保護機能により、物理アドレス空間は、デフォルトではユーザーレベルで動作するプログラムからはアクセスできません（図3）。拡張ボード

上のレジスタなどのリソースをユーザーレベルでアクセスする場合には、デバイスドライバを作成するか、T-Kernel/SM（System Manager）のアドレス空間管理機能を用いて拡張ボードが使用する領域をユーザーレベルでアクセス可能な論理空間にマッピングする必要があります。

ローカルバスは、バス幅も広く、DMAや割込みも使用できますので、最も性能が出しやすく、また自由度の高いインタフェースです。

T-Engineフォーラムでは、拡張コネクタのピンアサインを規定していません。というのも、ローカルバスのプロトコルはメーカーやCPUの品種により異なるからです。そのため、あるT-Engine向けに作られた拡張ボードは、他のT-Engineで動作するとは限りませんのでご注意ください。

他のT-Engine向けに作られた拡張ボードを誤って挿入してT-Engineや拡張ボードを破損してしまうことがないように、T-Engineの拡張コネクタの両端に誤挿入防止キーが付いています。このキーは、CPUの種別や拡張コネクタに出力されているバスの種類により、T-Engineフォーラムが割り当てています。

●PCIバスによるハードウェア拡張

T-Engine製品の中には、拡張コネクタにローカルバスとともにPCIバスを出力しているものもあります。PCIバスはCPUと周辺デバイスを基板上で接続するための標準規格です。PCIを使用することでCPUと周辺デバイスのメーカーや品種を問わず接続できますし、何よりも多種多様なデバイスがPCやワークステーション向けに製品化されていて、部品の選択肢がたいへん広いのが魅力です。

PCIを搭載するT-Engineの拡張コネクタのPCI部分のピンアサインは統一されています。したがって、PCIを使用した拡張ボードはPCIを持つT-Engine同士で共通化することが可能です。

PCIはバス構造ですので、バス上に接続されたバスマスタデバイス同士の調停を行う必要があります。この調停を行うデバイスはPCIホストまたはセントラルリソースと呼ばれます。PCIバスを備えたT-Engineは、このPCIホスト機能を搭載しており、最大3個までのPCIデバイスを接続することができます（図4）。

開発用拡張ボード

拡張ユニバーサルボードや、機能拡張FPGA開発ボードは、T-Engineを中心とした組込みシステム開発において、小規模な試

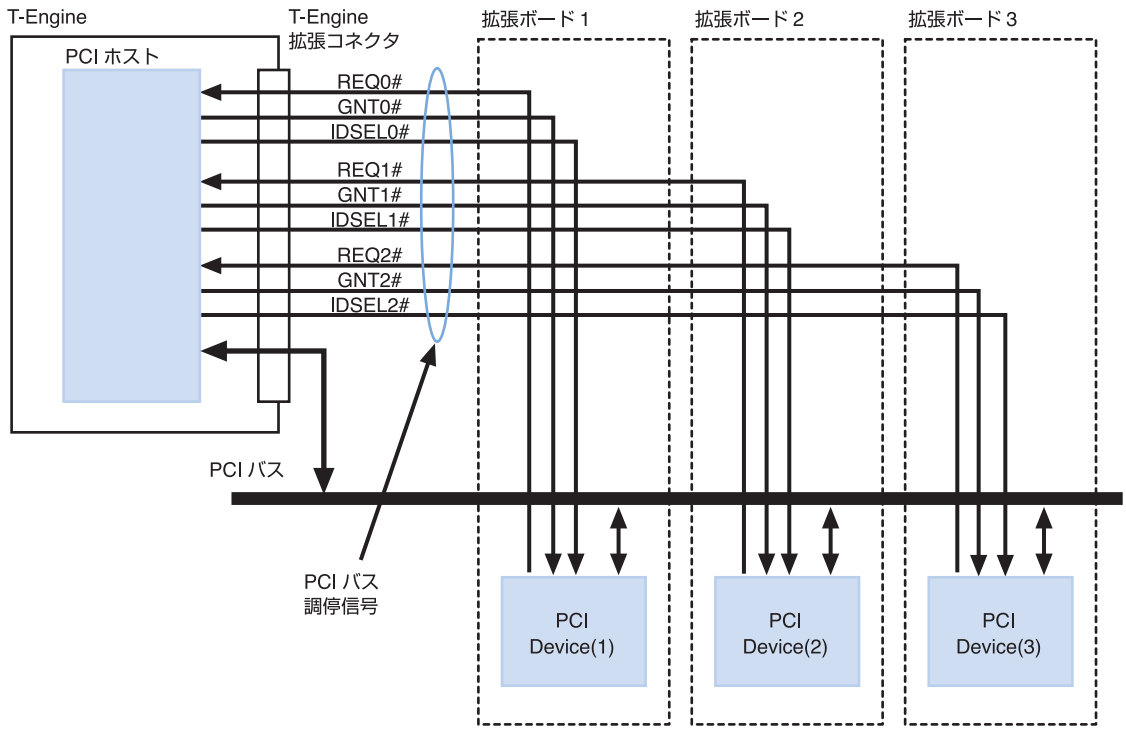

図4　PCIを用いた拡張ボードの構成

作を安価に行うことを可能にするものです。

●拡張ユニバーサルボード（ルネサステクノロジ）

拡張ユニバーサルボードは、手配線が難しいT-Engineの拡張コネクタの信号を、2.54mmピッチのスルホールに引き出し、ユーザーが簡単にハードウェアを拡張できるようにしたボードです（図5）。拡張コネクタに出力されているバスの種類により、4種類のボードがセットになっており、SH7760、SH7751R、SH7727、SH7145、M32R、VR5500、VR4131、TX4956の各標準T-Engine、μT-Engineで使用できます。

ボード上には2.54mmピッチのユニバーサルエリアが設けられており、任意の回路を手配線で接続することが可能です。小規模な実験・評価回路を接続するのに最適です。

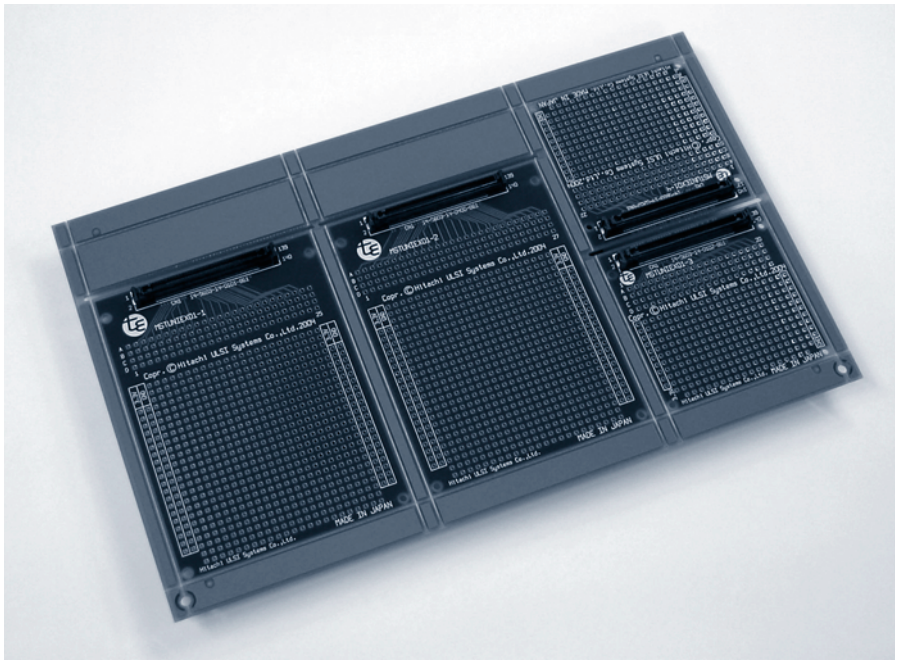

図5　拡張ユニバーサルボードの外観

●機能拡張FPGA開発ボード（アルティマ）

FPGA（Field Programmable Gate Array）は、内部回路をユーザーがプログラム可能なLSIです。FPGAの中には、Logic Elementと呼ばれるフリップフロップを中心とした回路ブロックがあらかじめ多数配置されていて、外部から配線をプログラムすることで任意の論理回路を実現します。

T-Engine向け機能拡張FPGA開発ボードの構成を図6に、T-Engineに接続した外観を図7に示します。FPGAはAltera社のCyclone EP1C20を搭載しています。このFPGAは、約20000個のLogic Elementを搭載しており、約25万ゲート規模の回路をプログラムすることができます。また、基板上にSDRAMを搭載していますので、高速で大容量のメモリを使用するような大規模システムにも対応します。

このFPGAボードは、T-Engineの拡張コネクタの端子すべてをFPGAに接続し、バスプロトコルの違いをFPGAの内部論理で吸収することで、全T-Engineに対応します。

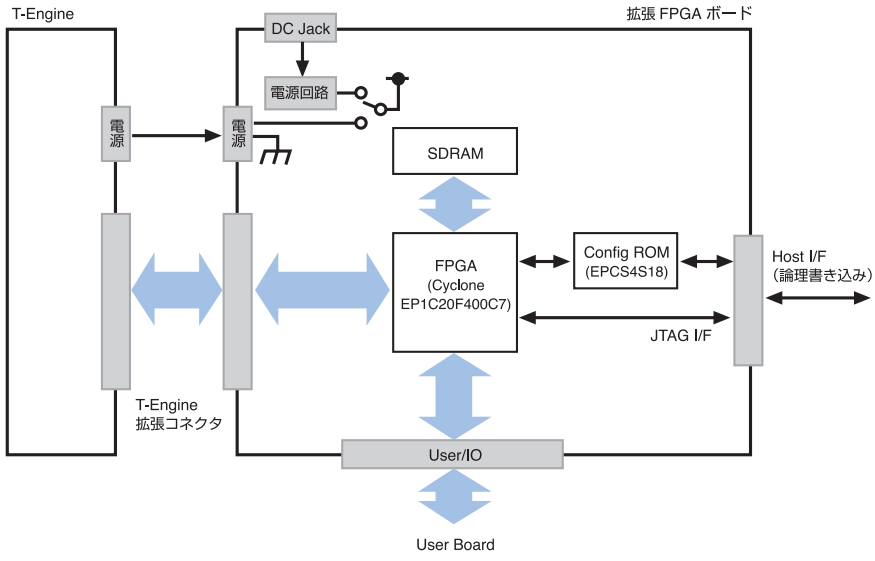

図6　T-Engine用拡張FPGAボードの構成

拡張コネクタの誤挿入防止キーも、すべてのT-Engineで使用できるマスターキーになっています。

FPGAの内部論理は、再プログラミングすることですぐに変更が可能なので、設計変更から実機検証までの時間（Turn Around Time）を短縮することが可能になります。一般に、ASICなどのLSIの製造には数ヶ月単位の長い期間が必要ですが、FPGAを試作環境として使用することで論理設計が完了した段階で実機でデバッグが可能になり、設計・検証効率が劇的に改善します。

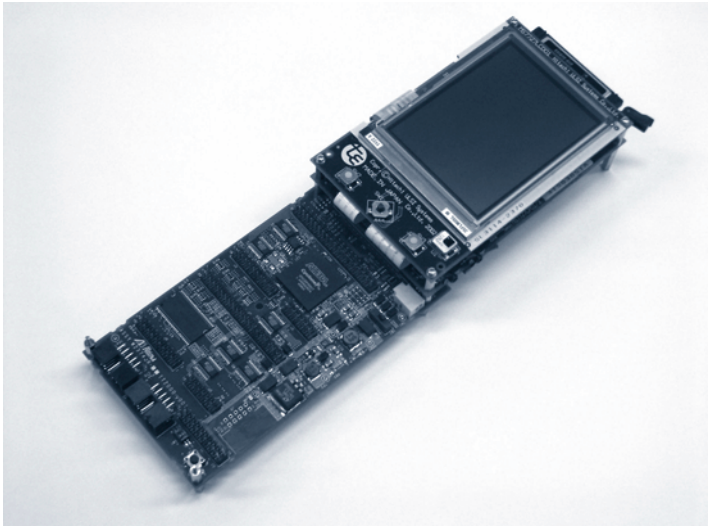

図7　T-Engine用拡張FPGAボードの外観

FPGAボードを使用した拡張回路の設計

●T-Engineのバスインタフェースの設計

近年、組込みシステムは大規模化しており、CPUを中心として多くの周辺機能を搭載するようになってきました。しかし、回路規模は増大する一方で、開発期間はますます短縮することが求められています。T-Kernelの強い標準化によってミドルウェアの流通、再利用性を高めるのと同じように、ハードウェアについても論理記述の流通、再利用性を高めることで設計効率を向上させることができます。

CPUのローカルバスにユーザー回路を設計する場合、そのCPUのバスプロトコルに合わせたバスインタフェースを持つ必要があります。しかし、特定のバスに依存した回路は、別のCPUのローカルバスに接続するたびに再設計が必要です。これでは再利用性が高いとはいえません。そこで、図8に示すように、ユーザー回路を特定のCPUバスに依存しないFPGA内部の共通バスに接続します。CPUのローカルバスとFPGA内

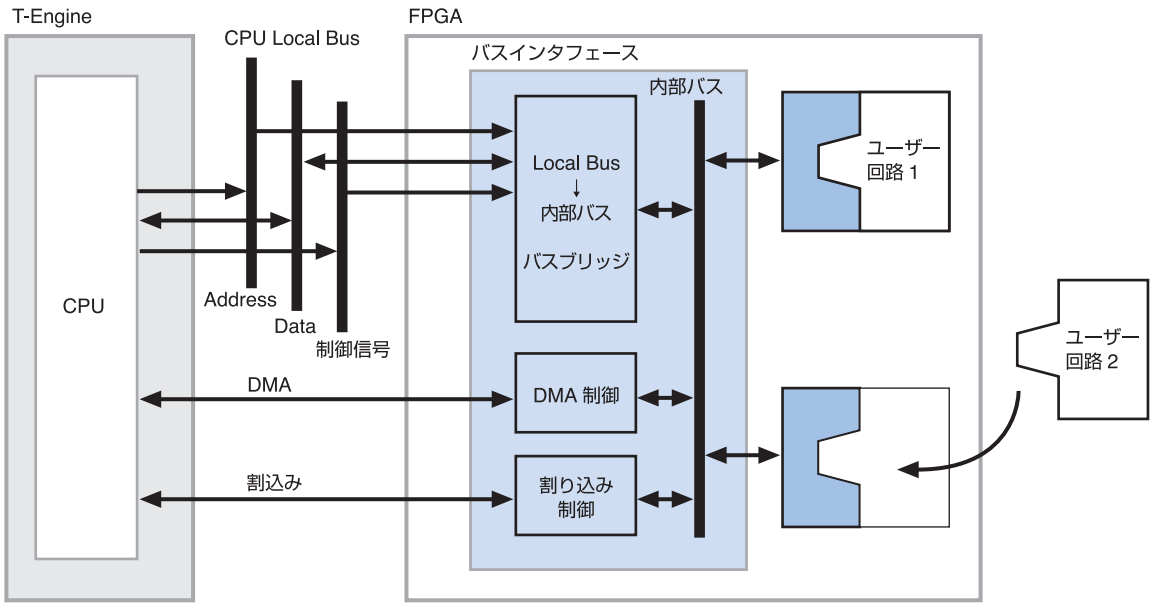

図8　FPGA内部バスにユーザー回路を接続

部バスとの間は、バスブリッジ回路により接続します。このバスインタフェースは、CPUのローカルバスのプロトコルの違いを吸収する役目を果たします。異なるCPUを搭載したT-Engineにこのユーザー回路を接続する場合には、このバスインタフェース回路のみを開発すれば、ユーザー回路の再利用が可能になります。

●FPGAの開発フロー

図9にFPGAを使用した一般的なハードウェアの開発フローを示します。FPGAの論理開発には、伝統的な回路図を用いてゲート設計することもできますが、近年はVerilogやVHDLといった、ハードウェア記述言語（HDL）で設計するのが一般的です。HDLは回路図よりも抽象的な記述が可能であり、可読性が高く、メンテナンスしやすいからです。できた論理記述は論理シミュレータを使用して検証します。シミュレーションによって動作を確認したら、論理合成によってゲートに変換します。次に、配置配線により論理回路を対象となるFPGAの内部論理にマッピングします。

この際、論理記述とは別に、制約と呼ばれる情報を与える必要があります。制約とは、配置配線をする際の条件になる情報です。制約には、どの信号をFPGAのどの端子に割り当てるか指定する端子制約や、回路がどの程度の速度で動作しなければならないかを指定するタイミング制約などがあります。

端子制約を誤ると、回路が正しく動作しないだけでなく、T-Engineの拡張コネクタの出力端子とFPGAの出力端子が衝突するなどして回路を破壊することもありますので端子制約の作成は慎重に行う必要があります。タイミング制約では、たとえば目的の回路が100MHzで動作するものであれば、フリップフロップ間のすべてのパスのゲート遅延が10ns以内でなければならないという制約を与えます。

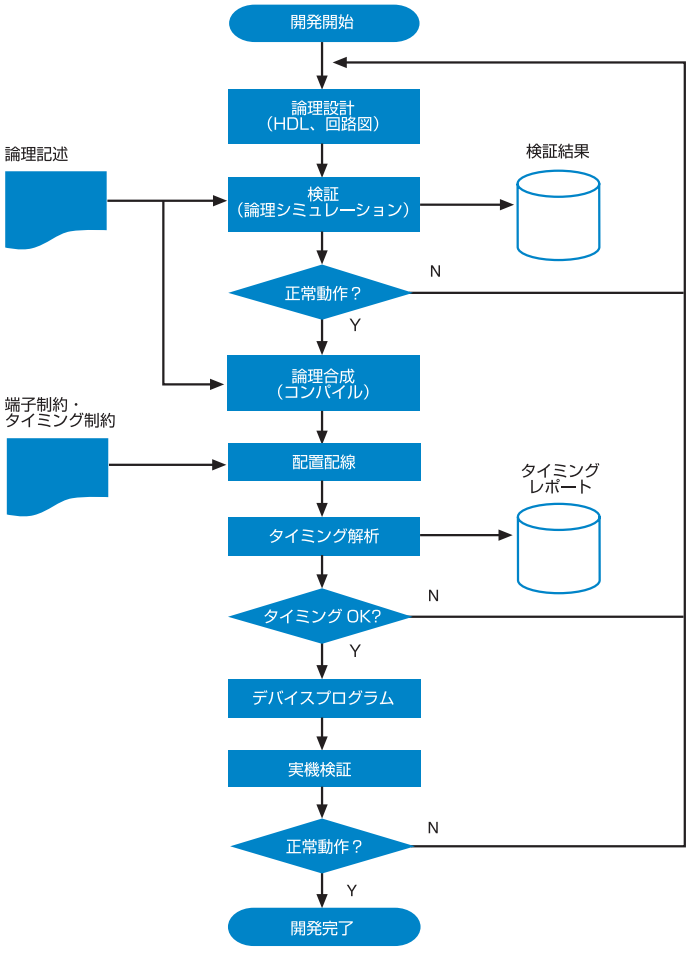

図9　FPGA開発フロー

配置配線後のタイミング解析結果に問題がなければ、FPGAに論理をダウンロードして実機で検証を行います。

これらの一連の流れは、FPGAベンダーより提供される開発環境上で行うことができます。設計変更がハードウェアの変更なしにできるため、設計・検証効率がソフトウェア並みに向上します。

まとめ

以上に述べたように、小規模な試作でも簡単にハードウェアを設計することができるユニバーサルボードやFPGAボードの整備により、T-Engineをハードウェア開発のプラットフォームとしても使用できる環境が整ってきました。T-Engineを使用することで、OSやCPUまわりのトラブルで悩むことなく、開発初日からソフトウェアとハードウェアの開発に着手できる利点があります。

T-Engineは、ボードの回路図やOSのソースコードまで公開されており、組込みシステムの動作を学習するのに最適な環境です。開発現場だけでなくこれから組込みシステム開発技術者を志す方にもぜひご活用いただけたらと思います。