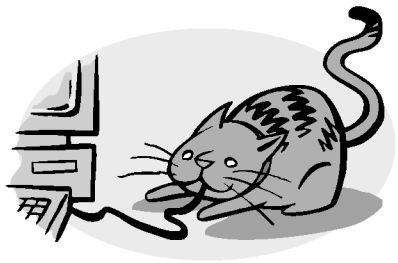
オンチップバスの話



NEC エレクトロニクス 第一カスタムLSI事業部
k-anjo@cw.jp.nec.com
安生 健一朗

バスとは何か？ ―

バスは，ハードウェアを構成するブロック間の通信手段として，さまざまな装置内のチップ間やチップ内部で使用されている．チップ間の接続に使用されているバスはオフチップバス，チップ内部のバスはオンチップバスと呼ばれている．本来，チップ間またはチップ内部の機能ブロック間で通信するための仕組みとしては，通信が生じるであろうブロック間に個別の配線を用意すれば十分である．しかしながら，専用の配線を個別に用意すると配線数が膨大となり，ボードやチップの面積が増加し，コストが高くなってしまう．そこで，バスと呼ばれるブロック間で共有する配線を用い，配線コストを抑えることになる．しかしながらバスを用いると，複数のブロックからの転送が配線上で衝突しないように排他制御したり，すべてのブロックが同じ手法でデータをやりとりするためのデータ転送のタイミングや手順（プロトコル）を定めたりする必要が生じる．

図-1 を用いてバスの基本的な転送プロトコルを説明する．バスに接続される機能ブロックは，CPUやDMAコントローラといったリード／ライトなどのリクエストを発行するユニット（マスター）と，メモリのようにリクエストを受けてデータを返す動作をするユニット（スレーブ）に分類される．バスプロトコルは，(1) マスターへのバス使用権の付与，(2) アクセスするスレーブの選択，(3) データ転送，(4) バス使用権の解放のステップで実行される．

(1) マスターにおいては，たとえばCPUのロードストア命令などによってバスへのアクセス要求が発生する．このときマスターでは，アクセスの対象となるアドレスと要求の種類を示すコマンド，およびライトコマンドであれば書き込むデータも保持する．マスターは，バスを介してスレーブにアクセスするため，調停によってバスの使用権を獲得する．調停は，バスがどのマスターにも使用されていないアイドル状態の時，定められた優先度に基づいて，アービタと呼ばれる回路が唯一のマスターに対しバスの使用権を与える仕組みである．

(2) 次に，マスターはアクセスするスレーブを選択する．スレーブの選択方式はバスの種類によって異なるが，マスターがアドレスをバス上に出力し，その一部を選択信号に割り当てることでスレーブを選択（デコード）する方式が一般的である．このデコードは，専用のデコーダで集中的に行う場合と，各スレーブが自分自身で行う場合とがある．

(3) スレーブを選択すると同時に，マスターは定められた手順でリード／ライトなどのコマンドを出力し，ライトコマンドならば書き込むデータも転送する．選択されたスレーブが，リードコマンドを受信した場合には当該アドレスのデータを返送，ライトコマンドならば当該アドレスに受信したデータを書き込む．

(4) スレーブから完了メッセージが返送され，マスターはバス権を解放してデータ転送を終了する．コマンドがリードの場合にはデータの返送をもって完了メッセージとすることができる．一方，ライトの場合には完了メッセージは必ずしも必要ではないためメッセージ返送を省略することで，無駄なバスの占有を減らし性能劣化を避ける場合もある．

以上が基本的なバスプロトコルであるが，実用化されているバスのプロトコルはそれぞれ用途や要求性能によって詳細な仕様は異なる．実用化されているバスの中でも特にオフチップバスでは，異なったベンダで

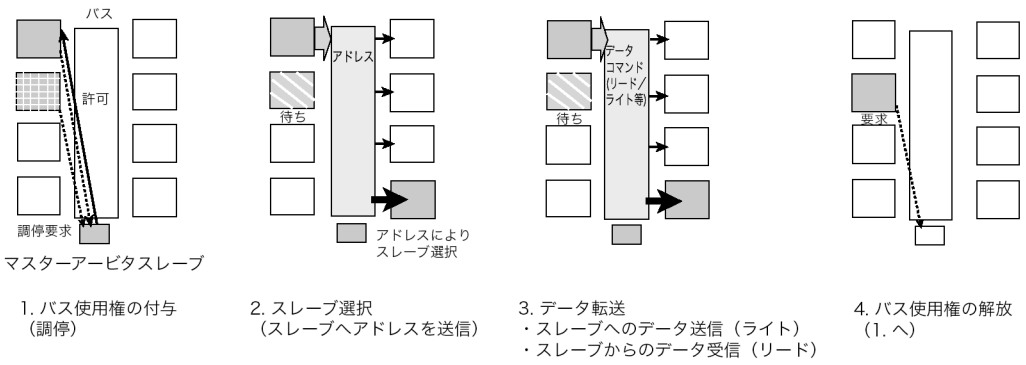


図 -1　バスの基本プロトコル

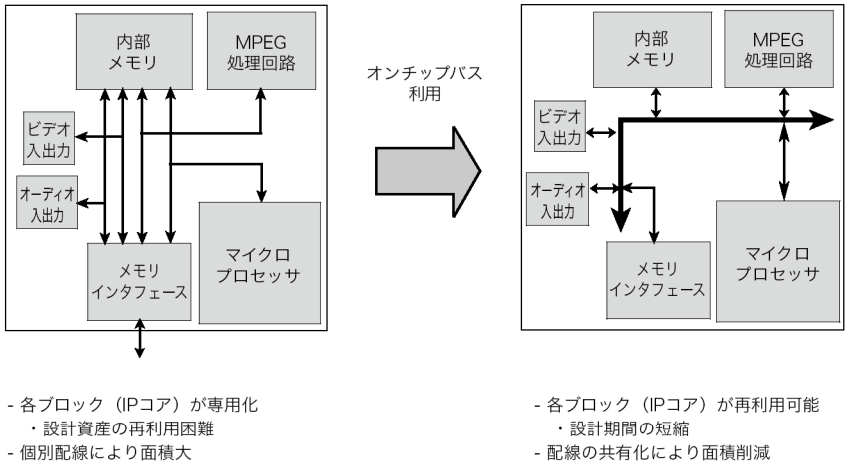


図 -2　オンチップバスを用いたシステムオンチップへのシフト

製造されたチップ間を接続するため，仕様が標準化されているものが多い．装置内の汎用バスとしては古くは ISA，PCI，PCI の性能改善版の PCI-X，さらに最近では PCI とソフトウェア互換で大幅な高性能を実現した HyperTransport や PCI Express などがある．また，ディスク接続用規格として SCSI，ATA，高速版ディスクインタフェースとして Fibre Channel，Infiniband などがある．一方，オンチップバスは，チップベンダが自社または顧客のチップ設計を効率化するために独自に規格を定めていることがほとんどであり，ARM 社 の AMBA[1]，IBM 社 の CoreConnect[2]，NEC の NECoBus[3]，SONICS 社の μNetworks[4] などが挙げられる．

オンチップバスのこれまで ──

過去，1980 年代にはチップ上にオンチップバスは存在しなかった．この時代には，最も高機能な半導体チップでも 8/16bit マイコンなどであり，ハードウェアの機能はボード上でトランジスタや論理ゲートなどを組み合わせて実現されていた．

集積度は 1990 年代にかけて著しく向上し，従来ボード上で実現されていた機能もチップ上で実現することが可能となってきた．この頃には，半導体チップ上にたとえば 8bit CPU，メモリ，I/O インタフェースなどの機能を搭載することが可能であった．すると，異なる設計者の開発した回路ブロック間で正しくデータ転送するための仕組みが必要となり，また設計した機能ブロックを別のチップにも流用したいといった要求が生まれ始めた．しかしながら，搭載可能なブロック数が少ないためバスのように配線を共有化する必然性はなく，個別にブロック間に通信用の配線を用意すれば十分であった．

1990 年代後半から 2000 年代になると集積度がさらに高まり，32/64bit CPU，SDRAM インタフェース，ネットワークインタフェースなど多彩な機能を持ったチップが実現可能となった．しかしながら，実現可能なチップ規模が大きくなり 2 つの問題が生じた．1 つ目は個別

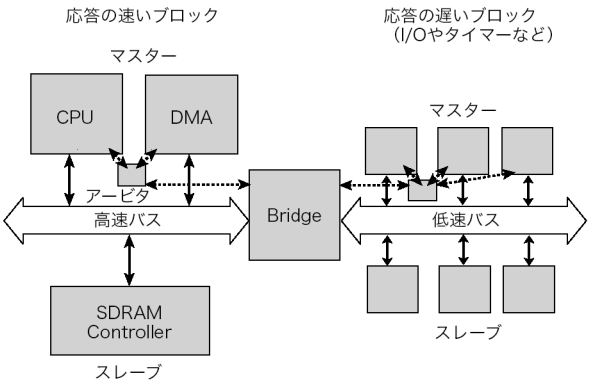


図 -3　オンチップバスの構成例（AMBA，CoreConnect）

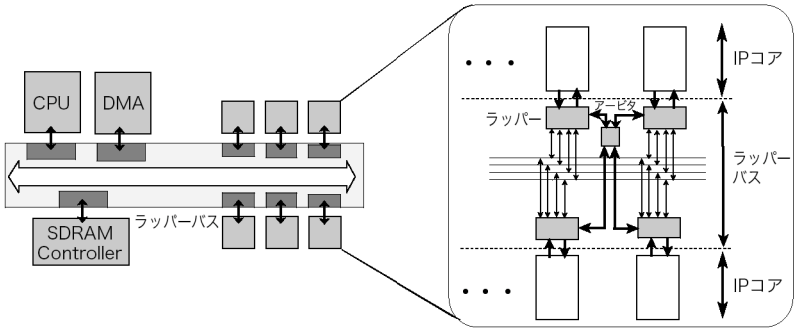


図 -4　ラッパーバスの構成例（uNetworks，NECoBus）

配線によるチップ構造の複雑化，配線面積の増大である．2 つ目は，設計者数の増加による担当ブロック間の仕様不整合や検証する機能の増加による設計の長期化である．そこで，これらの問題を解決するため，LSI 設計にオンチップバスが用いられるようになってきた．図 -2 にオンチップバスを LSI 設計に採用した例を示す．図に示すように，元々各ブロックが互いに通信する際には専用に配線を用意していたが，バスを採用することで配線数を削減し，ブロック間通信プロトコルも 1 つに定めることができる．さらに，各機能ブロックを IP（Intellectual Property）コアと呼び，これ単体を流通させて設計コストを削減することも可能となった．このような背景から，AMBA や CoreConnect などのオンチップバスを用いた SoC 設計は標準的な設計フローとして認識され始めてきた．

現在実用化されているオンチップバスの例として AMBA，CoreConnect の構成を図 -3 に示す．AMBA と CoreConnect はほぼ同様な構成を持ち，応答の速いユニットのみが接続される高速バスと，応答の遅いユニット用の低速バスとで構成されている．この両バスの間に存在するブリッジと呼ばれるユニットが，高速なユニットから低速なユニットへのアクセスを肩代わりし，その間高速バスが占有されて性能が劣化しないように工夫されている．なお，AMBA と CoreConnect の違いは詳細な仕様にあるため，本稿では特に述べない．

ラッパーによるオンチップバス設計 ──

前述のように，オンチップバスでは配線を共有化し，バスプロトコルの規格化による IP コア間の接続性向上，および IP コアの再利用により設計コストを削減することを目指している．しかしながら実際には，IP コアの再利用性は向上したものの，規格を厳守してバスプロトコル動作を行う制御部を設計する必要が生じ，IP コアの設計負担はむしろ増大している．

オンチップバスよりもさらに設計コストを改善するため，バスを自動合成するという考え方が新たに提案されている．この合成されたバスをラッパーバスと呼び，IP コアとして取り扱う．このバスでは，ラッパーというバスプロトコル制御部に相当するハードウェアを自動生成して使用する．図 -4 にラッパーバスの構成を示す．図 -4 に示すように，ラッパーバスは従来のオンチップバスとは異なり，プロトコル制御を担当するラッパーをバ

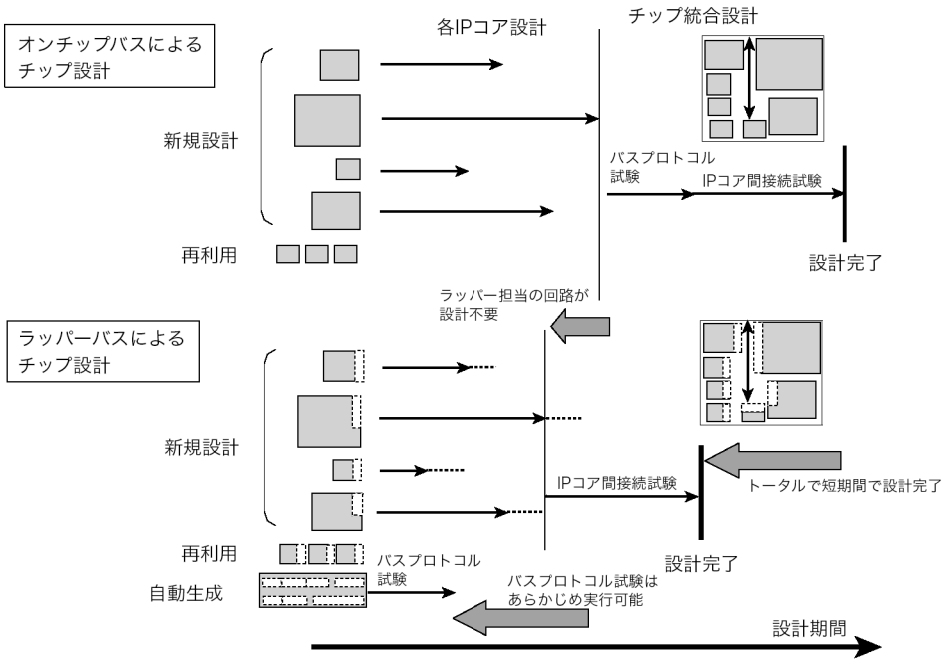


図 -5　ラッパーバスと従来のオンチップバスを用いた場合の設計期間の比較

スに含む構成である☆1.

LSI 設計にこのラッパーバスを使用することで 2 つの利点が得られる．1 つは，IP コア側で従来設計する必要のあったアービタやバス配線のタイミング制御が不要となるため，設計が容易化されて設計コストが低減することである．もう 1 つは，バス配線の構成やプロトコルが IP コアとは独立となるため，さまざまな性能ニーズに適合したバス構成を選択することが可能となる点である．たとえば，高い性能は要求しないが，面積は小さい方が望ましいチップでは，1 つのバス配線を持つラッパーバスを使用し，面積よりも高い性能が重要な場合には，複数のバスを使用したラッパーバスを用いることもできる．

実用化されているラッパーバスとしては，SONICS 社の μNetworks，NEC の NECoBus 等がある．μNetworks では，ブロックごとに割り当てるバス配線の使用権を時分割にすることで，要求された性能を必ず満たすバスを，ラッパーとともに自動合成する．NECoBus では，IP コアのインタフェースを可能な限り簡素化して IP コア設計を容易化し，通常のオンチップバスよりも設計コストを削減することを狙っている．

ここで，ラッパーを用いたバスの設計コスト削減の効果について**図 -5** を用いて説明する．図 -5 は従来のオンチップバスとラッパーバスとの設計期間の比較である．一般にチップ設計は，IP コア設計とそれらを統合したチップレベル設計とに分かれる．従来のオンチップバスを用いた場合には，まず再利用可能な IP コアは流用し，残りの IP コアは新規に設計する．これらの IP コアはオンチップバスのプロトコル制御部を内部に搭載している必要がある．その後チップレベル設計では，IP コア同士を繋ぎ，まずバスプロトコルの検証をした後に，IP コア間の協調動作の確認など統合的な機能を試験する．一方，ラッパーバスを用いて設計した場合には，IP コアではプロトコル制御部の設計は不要であり，設計しなければならない回路量は減る．また，バスのラッパーなどの構成要素は自動合成されるため，IP コア設計期間中にバスの生成および動作確認をすることも可能となる．そのため，チップレベル設計期間も短くなる．

オンチップバスの将来 ──

本稿を通じて，オンチップバスは，LSI の集積度の向上に伴った SoC の設計コストの増加を抑制するための技術であることを説明した．今後，LSI の集積度がさらに向上することで実現すべき機能が増加し，また高い処

☆1 オンチップバスで用いられていたブリッジは，低速バスが高速バスの性能を劣化させないためであった．ラッパーバスでは，ラッパーを変更すればバスのプロトコルを最適化し，性能を改善することができるため，オンチップバスで用いられていたブリッジは不要である．

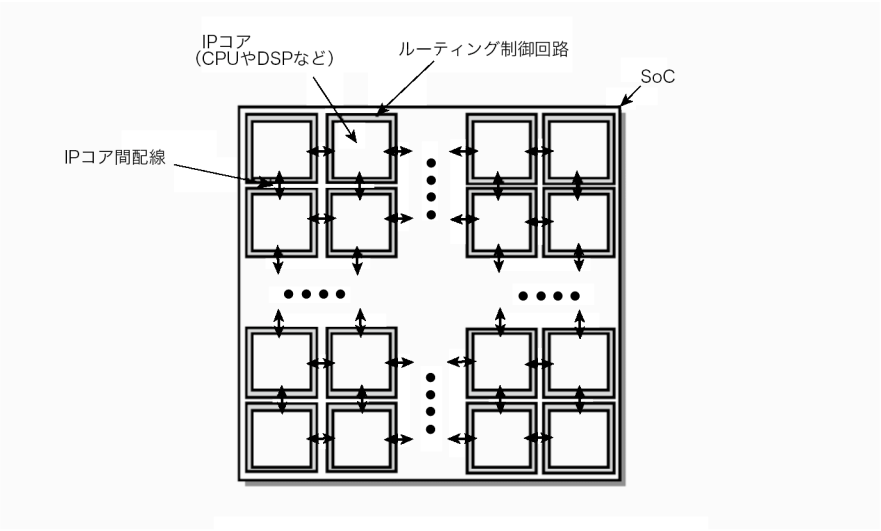


図 6　オンチップネットワークを適用したチップ

理性能も要求されてくる．このとき，SoC の設計においては以下の 2 点が重要となる．1 つ目は，より多機能化，高性能化するチップを実現可能な，バスに代わる効率のよい転送方式の構築，もう 1 つは多くの機能を持ったチップを短期間で容易に設計する手法の確立である．

オンチップバスでは，配線数を削減してチップ面積を小さくし，さらに設計コストを低減することを狙っていた．しかしながら，配線は共有して使用するため，同時に複数の IP コア間で通信できず，性能向上の障害になる．将来，たとえば高性能なネットワークルータでは，数十 Gb/s といった大容量な転送レートを実現し，また PC においてもプロセッサの動作周波数が数十 GHz という時代になるであろう．また，携帯電話などに用いられる LSI では低電力でありながらも，より多様なサービスを実現するため高い性能が要求されるであろう．このような LSI では，オンチップバスのように配線を共有する手法では要求を満たすことが難しくなる．そこで，より効率のよい方式として，オンチップネットワークと呼ばれる構成[5]が提案されている．**図 -6** にオンチップネットワークを適用したチップの構成の一例を示す．CPU や DSP 等の各 IP コアは，周囲にルーティング制御回路を搭載し，通信に使用されるデータであるパケットを転送する機構を持つ．IP コア間の通信には，互いに隣接する IP コア間で 1 対 1 接続した配線が使用される．離れた IP コア間では，途中に配置された IP コアのルーティング制御回路がパケットを適切に転送することで通信できる．このオンチップネットワークを用いると，コンピュータ間のネットワーク接続のように数多くの IP コアを接続し，高いデータ転送レートを実現することが可能となる．

一方，LSI の集積度はますます高まり，大規模かつ多機能化するチップ設計は長期化し続けるであろう．そのため，このようなチップを容易に設計する手法が望まれる．したがって，あらゆる用途に用いられるチップを，統一されたブロック間通信規格で設計し，IP コアの流通をさらに促進することが重要である．本稿で述べたラッパー技術は，IP コア間の接続部分を IP コア設計から独立化させることができるため，従来のオンチップバスから将来のオンチップネットワークまで，多岐に渡るラッパーバスを実現することが可能である．また，さらにこのフレームワークを支える設計環境として，たとえば要求する性能や電力，面積を入力すると，自動的にラッパーバスが生成されるような支援ツールがあれば，さらに設計コストを削減することも可能となる．これにより，さまざまな用途間で IP コアが流通し，最先端の IP コアをどのようなチップでも簡単に組み込むことが可能となることを期待する．

参考文献

1）http://www.arm.com/armtech/AMBA?OpenDocument, AMBA .
2）http://www-3.ibm.com/chips/products/coreconnect/, CoreConnect™ Bus Architecture .
3）Anjo, K. et al.: NECoBus: A High-end SOC Bus with a Portable & Low-latency Wrapper-based Interface Mechanism, Proc. of IEEE Custom Integrated Circuits Conference, pp.315-318（May 2002）.
4）Wingard, D. et al.: Integration Architecture for System-on-a-Chip Design, Proc. of IEEE Custom Integrated Circuits Conference, pp.85-88（May 1998）.
5）J. Dally, W. and Towles, B.: Route Packets, Not Wires: On-Chip Interconnection Networks, Proc. of IEEE Design Automation Conference, pp.684-689（June 2001）.

（平成 14 年 10 月 2 日受付）