解　説

IPS

非同期式プロセッサ
——超高速 VLSI システムを目指して——†

南　谷　　崇††

1. まえがき

最近のデバイス技術の進歩は著しい．すでに実験室レベルでは，スイッチング遅延が数ピコ(10^{-12})秒という超高速論理素子が実現されたという報告がいくつかある．これらの“ピコ秒論理素子”がいかに速いものであるかは，1ピコ秒では光でさえわずか 0.3 mm の距離しか進まないことを考えれば，よく分かる．これらの超高速論理素子をVLSIとして実用化するためには，熱の問題など，まだ LSI 製造技術の立場から解決すべき課題も多いと思われるが，これまでの技術の進歩を振り返ってみれば，今世紀中に実用領域に入る可能性は高いと思われる．

素子が速くなると同期式システムの設計に深刻な問題が起きる．素子遅延に比べて相対的に大きくなる配線遅延のためにクロック周波数を素子速度に見合うレベルまで上げることができないためである[1),2)]．実際すでに，従来は論理設計の一部と考えられていた同期式タイミング設計をレイアウトや実装段階まで含めて考えなければならなくなり，クロックスキュー最適化のために多くの努力が払われている[3),4)]．しかし，同期式システムの性能向上には明らかな限界があり，すでにその限界に近づきつつある．たとえば，現在の標準的なチップ寸法である 1 cm×1 cm の正方形領域に同期式システムを実現する場合，素子遅延が数十ピコ秒以下になるとその素子の高速性をそのまま享受することはできなくなる[5)]．システム領域がこれより大きくなれば状況はさらに悪くなる．

クロックを用いない非同期式システムは，原理的には，信号遷移の因果関係だけに依存して動作させることができる．システムの性能を決めるのは素子遅延，配線遅延を含めた信号伝播遅延の「平均値」である．したがって，この平均遅延を最小化するアーキテクチャとそれを支える論理設計方法論を確立すれば，ピコ秒素子の高速性をフルに享受した超高速プロセッサを実現できる可能性がある[6)]．

本稿では，ピコ秒素子が実用化された場合の同期式システムの性能限界を示すとともに，これまでの非同期式システムの理論と実際を概観し，超高速プロセッサの実現を目指して，最近の研究動向と今後の技術課題を示す．

2. 同期式プロセッサの限界

同期式システムでは，アーキテクチャと論理設計が同じであるならば，プロセッサ性能はクロック周波数あるいはクロック周期で決まる．可能な最小クロック周期は，「レジスタ間データ移動に要する時間の最大値 Ts」で与えられる．これは，素子遅延，配線遅延を含めたレジスタ間経路の最大伝播遅延とクロック分配の最大時差によって決まる．したがって，すべてのシステム内レジスタ間転送を1クロックで実行する場合，素子遅延がいかに小さくなっても最小クロック周期をシステム領域の寸法によって決まるある下限より小さくすることはできない[5)]．

一方，非同期式システムの場合は，プロセッサ性能は「レジスタ間データ移動に要する時間の平均値 T_a」で与えられる．これはレジスタ間経路と制御信号経路の信号伝播遅延のシステム動作期間にわたる期待値である．したがって，素子遅延に比べて配線遅延が支配的になった場合には，システム性能は，システム領域寸法とは無関係に，システム動作期間中の長距離データ移動に対する

† Asynchronous Processors —Toward Ultrahigh-Speed VLSI Systems— by Takashi NANYA (Department of Electrical and Electronic Engineering, Faculty of Engineering, Tokyo Institute of Technology).

†† 東京工業大学工学部電気電子工学科

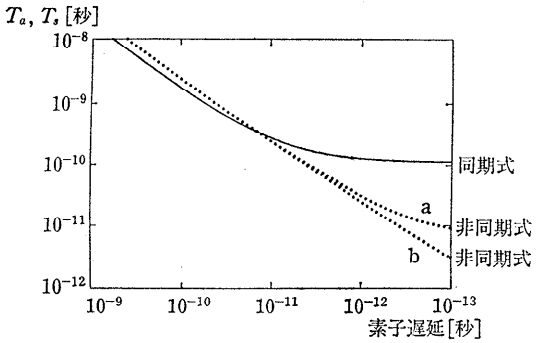


図-1 同期式システムの性能限界

近距離データ移動の相対的発生頻度によって決まる.

図-1 は, 10 mm×10 mm の正方領域にシステムを実現する場合に, 素子の高速化に対応して達成可能なシステム性能の限界 (T_s 及び T_a の下限) を示したものである. 横軸は素子遅延 (秒) を, 縦軸は同期式の T_s (秒) 及び非同期式の T_a (秒) を表す. 実線が同期式システムを, 破線が非同期式システムをそれぞれ表している. ただし, 配線の信号伝播速度は真空光速度の 1/2 とする. 同期式システムのレジスタ間経路の最大ゲート段数を 14 段, 非同期式システムのデータ移動でデータ及び制御信号が通過する平均ゲート段数を 25 段と仮定している. この図から明らかなように, 素子遅延が大きい場合には, ゲート段数の差のため同期式のほうが非同期式より高速になることは可能である. しかし, 素子遅延 20 ピコ秒付近からは配線遅延の影響が支配的となるため非同期システムのほうが速くなることが分かる. 素子遅延 10 ピコ秒以下では同期式システムの性能は飽和する. これは理想的な設計・実装を前提とした性能限界であるから, 実際にはもっと早く飽和するかもしれない.

もちろん, 非同期式システムの性能限界も配線遅延の影響を受ける. システム領域の一片の長さを L(=10 mm) としたとき, 破線 a はレジスタ間データ移動距離の期待値が $L/10$ の場合を, 破線 b は同じく期待値が $L/100$ の場合の T_a の下限をそれぞれ示している. 素子の高速化にともなって, データ移動距離の期待値を最小にするアーキテクチャが重要になることが分かる.

3. 同期式か非同期式か?

非同期式回路の歴史は古い. 1940 年代後半にはすでにリレー, 真空管を用いたいくつかの非同期式プロセッサが存在していた[7]. 1950 年代には, これらの開発経験から, 非同期回路の二つの重要な理論的モデル, すなわち, Huffman モデル[8]と Muller モデル[9]が生まれた. それ以来今日まで約 40 年にわたって, 非常に多数の研究が積み重ねられてきた[10)~15)]. 論理回路に関するほとんどの教科書には非同期式回路の記述がある[16].

しかし現実には, 第 2 世代 (トランジスタ) 以降は, Illiac Ⅱ などのわずかな例外を除けば, これまで存在したほとんどすべてのコンピュータは同期式であると言ってよい. いわゆるノイマン型コンピュータはもちろんのこと, アルゴリズムやアーキテクチャのレベルで高速化を狙って最近提案されているさまざまな非同期処理方式も, それらを実行するハードウェア基本要素としては同期式プロセッサが用いられている. その主な理由は, システム設計者, 回路設計者の次のような "常識" にあると思われる.

1) 同期式のほうが設計とテストが容易である.
2) 同期式のほうが金物量が少ない.
3) 同期式のほうが速い.
4) 同期式のほうが信頼性が高い.

常識 1 は, 現段階ではそのとおりであろう. 同期式プロセッサの設計論は確立されており, スイッチング回路理論や計算機構成論を受講した学生ならば, 半年の学生実験で, 小なりと言えども独自のアーキテクチャのコンピュータを最初から設計, 製作し, その上でプログラムを走らせることができる. それに匹敵する非同期式プロセッサの実用的かつ組織的な設計論はこれまでほとんどなかったと言ってよい.

常識 2 もまたそのとおりである. 一般に, 非同期式回路のほうが同じ論理機能を実現する同期式回路より多くの論理素子を必要とする. ただし, 配線領域も含めたシステム面積はアーキテクチャに大きく依存する.

一方, 常識 3 と 4 は, 現在の素子速度のレベルでも判定し難い. 回路速度と動作の信頼性はその設計法に大きく依存し, 評価尺度によって異なる

からである．実際，回路速度に関して，次のようなまったく相反する見解が語られることがその事情をよく示している：

「非同期式回路のほうが同期式回路より速い，という誤った先入観がある…」文献 17) より．

「ディジタル VLSI 設計における最も根強い偏見の一つは，非同期式回路は必ず同期式回路より遅い，というものである…」文献 18) より．

どちらの「偏見」が正統派であるかは別にして，前章に述べたように，「常識 3：同期式回路は速い」は，そう遠くない将来，間違いなく“非常識”になるはずである．

「適正な」遅延モデルのもとで「正しく」実現された非同期式システムは，素子及び配線におけるいかなる遅延のばらつき，変動にもかかわらず，仕様どおりに正しく動作することが保証される．その場合には，次のような特長が期待できる：

1) 高速性：設計を変更することなく，システム中の任意の素子を常に最先端の高速素子に置き換えることができる．それによって図-1 に示した性能限界を享受できる．

2) 信頼性：設計，レイアウト，実装工程の不ぐあいや，システム稼動中の環境変動に起因するタイミングフォールトがない．

3) 設計コスト：論理合成で得られた回路の正しさはレイアウトや実装の結果に影響されない．すなわち，論理設計とレイアウト・実装設計が分離されるので設計コストは大幅に削減される．

4) モジュラリティ：検証済みの回路モジュールを基本ブロックとして，タイミング関係をまったく考慮することなく，任意に大きな回路を構成することができ，システムの拡張，設計変更が容易である．

4. 遅延モデル

デジタルシステムを設計する場合，素子及び配線の遅延に関するなんらかの仮定が必要である．その仮定が現実の遅延のばらつきに対して楽観的過ぎれば，正しい動作は保証されない．一方，悲観的過ぎれば，回路性能の効率が悪くなり，コストが高くなる．したがって，設計の前提となる遅延仮定は素子技術，システム実装技術に対して適正なものでなければならない．

同期式回路では，設計時に定められる素子遅延，配線遅延はそれ以後一定であることを仮定している．この仮定によってクロック周期の設定が可能になる．したがって，もし，レイアウト，システム実装，動作環境などの要因でその遅延仮定が破れればタイミングフォールトが生じることになる．

これに対して，非同期式回路は一般に，素子遅延，配線遅延の一部または全部が可変であり設計時には未知であると仮定する．したがって，システム全体を同期させるクロックは使用できない．ただし，一定の遅延仮定のもとで，局所クロックの使用や遅延素子挿入によるタイミング調整は可能である．遅延仮定をさらに厳密に区別することによって種々の非同期回路モデルがあり得るが，以下のモデルが代表的である．

1) Fundamental Mode (FM) モデル[12]

Huffman モデル[8]とも呼ばれ，すべての素子遅延及び配線遅延は未知であるが，その上限値は既知であると仮定する．このため，回路へ入力を供給する外界はその上限値に基づいて入力タイミングを制御できる．したがって，「回路が安定状態にあるときにしか回路への入力変化は起きない」とする前提が有効になる．この前提のもとで，順序機械あるいは有限状態機械の概念に基づいて，状態割当，ハザードフリー構成などが与えられる．FM モデルはよく研究され，その設計論は確立されている[16]．しかし，現実の多くのシステムでは多数の部分回路が相互に作用しながら並行動作を行うので，上記の前提がいつも成立するとは限らない．

2) Speed-Independent (SI) モデル[11]

Muller モデル[9]とも呼ばれ，すべての素子遅延は，有限ではあるが上限値は未知である．ただし，配線遅延はゼロである，と仮定する．上限値未知のため，FM モデルと異なって外界が入力タイミングを制御できない．したがって，回路自身が出力する「完了信号」によってのみ入力変化のタイミングが制御される．配線遅延を考える場合には，配線に模擬素子を挿入すればよい．

3) Delay-Insensitive (DI) モデル[19]

最も制限の少ない，したがって遅延のばらつきに関して悲観的なモデルである．すなわち，素子遅延も配線遅延も有限ではあるが上限値は未知であると仮定する．前に述べたように，素子の高速

化にともなって配線遅延は相対的に増加しており，レイアウト，システム実装，動作環境の各要因による配線遅延変動を設計時に精密に予測することは困難である状況を考えると，このDIモデルに基づく設計は魅力的である．しかし，残念ながら，DIモデルのもとで正しく動作する回路は特殊なものに限られ，実際に役に立つ回路はほとんど設計できない[20]．そこで，DIモデルを少し修正し，「分岐する配線の信号変化はすべての分岐先へ同時に伝播する」という同時分岐の仮定[20]を加えて，これを修正DIモデルと呼ぶことにする．修正DIモデルのもとでは，ほとんどの回路を構成することが可能である．ただしこの場合，分岐配線が一定の近傍領域（**6.4** 参照）を越えないように，設計上の制約が課される．

5. 制御とデータ転送

5.1 要求・応答プロトコル

非同期式システムは互いに信号をやりとりし合う機能モジュールの集まりとみることができる．信号には制御信号とデータ信号がある．遅延の上限値は未知なので，二つのモジュールAとBの間でこれらの信号をやり取りする場合，次のような要求と応答の組からなるシェークハンド規約に従う必要がある．

「要求」：モジュールAはBへ処理を要求するための（制御またはデータ）信号を送る．

「応答」：モジュールBはAへその処理の完了を応答する（制御またはデータ）信号を送る．

この場合，同期クロックが存在しないため，これらの要求信号や応答信号がいつ送出されたか，いつ到着したか，を表すタイミング情報を信号自身がもつ必要がある．

制御信号の場合には，その信号の「無効」状態から「有効」状態への遷移がそのまま要求または応答のタイミングを示している．しかし，データ信号の場合には，2値データの表現に加えて要求・応答のタイミングを表すためになんらかの符号化が必要である．最も簡単で実用的にも有効な符号は1ビット情報を2本の信号線で表現する2線式符号である．一般的にはもっと冗長度の低い種々の符号化が考えられるが[14]，データ経路を構成する論理回路を2線式符号と同程度に簡単に実現できる望みはほとんどない．

5.2 2線式データ表現

2線式符号を用いるデータ表現として少なくとも次の三つの方式がある．

（1） 2線2相方式[12),21)]

1ビットデータDを2線信号 (d1, d0) で次のように表す．

D=0⇔(d1, d0)=(0, 1)

D=1⇔(d1, d0)=(1, 0)

データの無効状態を (0, 0) で表す．状態 (1, 1) は使用されない．データ転送は無効状態 (0, 0) から始まる．遷移 (0, 0)→(0, 1) によって「0の発生（到着）」を表す．また遷移 (0, 0)→(1, 0) によって「1の発生（到着）」を表す．したがって，データ経路のどのインタフェースでも，データ転送を実行するために無効状態 (0, 0) から符号語状態 (0, 1) または (1, 0) へ遷移する期間（稼働相と呼ぶ）と次のデータ転送に備えるために符号語状態から無効状態へ遷移する期間（休止相と呼ぶ）を交互に繰り返す．

（2） 2線遷移方式[1),22)]

2線信号 (d1, d0) のうち，d0上の信号遷移が「0の発生」を，d1上の信号遷移が「1の発生」を表す．したがって，同じ信号線上での遷移 0→1 と 1→0 は同じ意味をもつ．たとえば，信号系列 0, 1, 1, 0, 1, 0 は，初期状態を (0, 0) とすると，次の遷移系列で表される．

(0, 0)→(0, 1)→(1, 1)→(0, 1)→(0, 0)→(1, 0)→(1, 1)

この例から分かるように，現在の状態 (d1, d0) からはDへの一意的な復号はできず，それ以前の状態に依存する．したがって，2線遷移方式によるデータ経路の論理関数を実現することは可能であるが，簡単ではない．

（3） パリティ交番方式[21),23),24)]

2線信号 (d1, d0) の4つの状態に論理値0及び1を次のように割り当てる．

(0, 0)＝偶数相における論理値 “0”

(0, 1)＝奇数相における論理値 “0”

(1, 0)＝奇数相における論理値 “1”

(1, 1)＝偶数相における論理値 “1”

すなわち，論理値はd1で与えられ，状態のパリティはd1⊕d0で与えられる．データ転送のタイミング情報は偶数相と奇数相を交番することによって得る．たとえば，前記の信号系列 0, 1, 1, 0, 1, 0 は，初め偶数相だとすると，次の状態系列

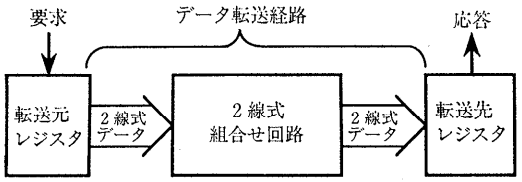


図-2　2線2相式データ転送経路のモデル

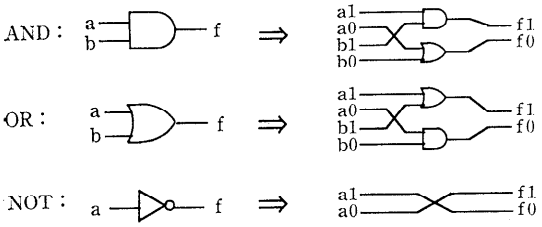


図-3　2線式論理の基本素子

で表される.

(0, 0)→(1, 0)→(1, 1)→(0, 1)→(1, 1)→(0, 1)

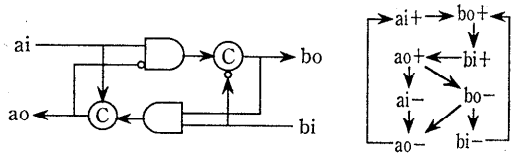
この方式も，2線遷移方式と同様，データ経路を構成する論理関数をハザードのない形式で実現することは可能ではあるが，簡単ではない.

5.3　2線2相式データ転送

三つの2線式データ表現のうち，汎用のデータ転送を実現するためには，2線2相式が最も簡単で有効である．2線2相式データ表現に基づくレジスタ間データ転送経路のモデルを**図-2** に示す.

データ経路を構成する論理回路は2線式論理によって実現される．**図-3** に2線式論理の基本論理素子を示す．NOT 演算は2線信号の交差で実現できるため，2線式論理回路は NOT 素子を含まずに実現できる．また，2相式であるため，稼働相では 0→1 方向の単調遷移，休止相では 1→0 方向の単調遷移になる．したがって，2線2相式データ経路にハザードは存在しないことが保証される.

稼働相では，外部からの要求の発生（0→1 遷移）によって2線式データ信号が転送元レジスタから読みだされ，論理回路で変換されて，転送先レジスタへ書き込まれた後，応答を発生（0→1 遷移）させる．休止相では，外部からの要求消滅（1→0 遷移）によって，転送元レジスタから論理回路を経て転送先レジスタへ至るすべての2線信号を無効状態 (0, 0) へ戻すために，いわば，データ経路の「掃除」が行われた後で，応答を消滅（1→0 遷移）させる.



（a）回路構成　　（b）信号遷移グラフ

図-4　2線2相制御モジュール

2線2相方式はデータ経路を構成する論理回路の実現が非常に簡単であり，しかもハザードフリー動作が保証される．しかしその半面，稼働相と休止相が交番するため，データ経路の処理時間のおよそ半分は実質的な仕事をしない休止相に費やされるという欠点がある．この問題を解決する一つの方法は，図-2 のデータ経路を2重化することである[25]．一方が稼働相を実行している間に他方の休止相を実行することによって，実効的に，処理時間全体が稼働相だけで費やされるようにみせることができる.

稼働相と休止相の並列実行は，自掃モジュールと呼ばれる非同期式制御回路によって効率的に制御できる[26]．自掃モジュールの回路構成を**図-4**（a）に示す．ここで，“C”で示される素子は Muller の C 素子[11]と呼ばれる2入力1出力素子である．C素子は，2入力の値が一致するとその値を出力し，一致しない場合に現在の出力値を保持する論理素子である．自掃モジュールの入出力のペア (ai, ao) は上位レベルからの要求とそれに対する応答信号であり，(bo, bi) は下位レベルへの要求とそれに対する応答信号である．これらの信号遷移の因果関係を **図-4**（b）に示す．図において，信号名に添加された＋記号はその信号の 0→1 遷移を，－記号は 1→0 遷移を表す．矢印は信号遷移の因果関係を表す．たとえば，ai の 0→1 遷移と bi の 1→0 遷移の結果として bo の 0→1 遷移が起き，その結果として外界が bi の 0→1 遷移を起こすことが記述されている．この自掃モジュールは修正 DI モデルのもとで上位レベルと下位レベルの2相動作を正しく制御する.

6. 非同期式システムの技術課題

この章では，最近の研究動向も含めて，非同期式システムを実現するために克服しなければならない技術課題を述べる.

6.1 "正しい動作"とは何か

システムの正しい動作とは,「ユーザが要求する仕様に従った出力を生成する」ことである. しかし, このような概括的な定義は非同期式プロセッサの設計, 検証, 評価にはほとんど役に立たない.

プロセッサの正しい動作とは, 一般的には, フェッチされた命令がその定義どおりに実行されることと考えられる. 同期式であれば, クロックサイクルごとにプロセッサの制御状態及びデータ転送が明確に定義できるため, クロック単位でプロセッサ内のすべての動作を記述することができる. 一方, 非同期式の場合には, 基準となる時刻が存在しない. このため, 理論的には以下のように, 最終的に一意的な安定状態へ到達するかどうかということによって正しい動作を定義している.

すなわち, Huffman モデルにおける正しい動作は, ある安定状態においてある入力変化が起きたとき, 状態遷移表に定義された"次の安定状態"に遷移することである. また, Muller モデルでは, ある初期状態から始まる可能なすべての状態遷移の系列が一意的な最終状態で終わることである.

しかし, 実際の非同期式プロセッサでは, プログラム実行が終了するまでは, 実行途中でそのような安定状態は存在しない. パイプライン方式による命令の並列実行が生じる場合も含めて, 論理合成, 設計検証で扱いやすい形式でレジスタ転送レベルでの正しい動作の定義を与える必要がある.

6.2 論理合成

最近になって SI モデルまたは修正 DI モデルのもとでの論理合成が活発に研究されるようになったが, それらはレベルの異なる三つの課題に分けられる.

第一は, 非同期システムを, メッセージによって互いに通信を行う非同期プロセスの集合体とみなし, CSP/occam ふう言語によるモジュール間の通信動作及びモジュール動作の記述を出発点とする論理合成である[26)~29)]. この上位記述を適当な生成規則を用いてあらかじめ定義された基本素子による下位記述へ変換する. 論理合成の新しい定式化であり, この方法を応用した非同期式マイクロプロセッサの設計例も報告されている[18)]. しかし, まだ生成規則を適用するヒューリスティクスの整備が十分ではない. 下位レベルでの最適設計が保証されるように生成規則を適用する組織的な方法を確立する必要がある.

第二は, 非同期式回路の動作を記述した信号遷移グラフ (STG)[30),31)] を出発点とする論理合成である[32)~34)]. 信号遷移グラフは, 図-4(b) に示されたように, ペトリネットの部分クラスである自由選択ネット[16)]において, トランジションで信号の遷移を表し, プレースで信号遷移の因果関係を表したものと考えることができる. 順序回路の表現に用いられる状態遷移図のサイズが, n 変数の回路に対して 2^n であるのに対して, 信号遷移グラフでは $2n$ のオーダである. これは, 信号遷移グラフが因果関係の記述に必要な信号遷移しか表現しないことによる. 信号遷移グラフから各信号に関する論理関数を導出することは容易である. しかし, その論理関数を SI モデルあるいは修正 DI モデルのもとでハザードなしに実現することは必ずしも容易ではない.

第三は, 図-2 に示されたデータ転送モデルにおける稼働相と休止相の繰り返しの中で要求信号に対する応答信号を正しく生成する論理回路の構成方法である. 単純に 2 線式論理を用いただけではそのような応答信号を生成できない[35)]. データ転送経路を構成する論理回路の性能はプロセッサ性能を決める大きな要素の一つであり, 高速かつ経済的な論理回路の構成が必要である.

6.3 テスト

非同期式システムでは, システムの稼働開始時点ですでに存在している永久故障と, 稼働中に発生する一時的故障を明確に区別する必要がある. 一時的故障としては過渡故障, 間欠故障, あるいはそのまま永久的に存在するものがあり得る.

永久故障のモデルとして代表的な縮退故障を考えてみよう. SI モデルまたは修正 DI モデルのもとで 2 線 2 相式データ経路が正しく実現されていると仮定する. もし, データ経路に 1 縮退故障が存在すれば, 休止相を完了することはできない. また, もし 0 縮退故障が存在すれば, その故障を活性化するような入力遷移に対してデータ経路は稼働相を完了させることができない. いずれの場合にも, 適当な完了信号生成回路とタイムアウト

機構によって検出することができる．したがって，ほとんどの場合，すべての永久縮退故障は検出可能である．

一方，稼働中に生じる一時故障や環境からの雑音に対しては，同期式システムの場合より問題は複雑である．同期式の場合，組合せ回路の一時故障がシステムに影響を及ぼすのはクロック時刻での発生のみである．また，記憶素子であれば，内部状態が反転するだけであるから，他へ影響が及ぶ前に誤り検出・訂正を行うことができる．これに対して，非同期式の場合には，組合せ回路における任意の時刻の一時故障がシステムに影響を及ぼす．また，記憶素子への一時故障は，単に状態を反転させるだけではなく，その結果がただちに誤った動作シーケンスの開始となって現れる．したがって，記憶素子が反転する誤りは適当な手段で検出できたとしても，その後の対処の仕方が大きな問題である．同期式回路の場合とは全く異なったセルフチェッキング設計[36]の概念が必要になる．

6.4 非同期式アーキテクチャ

素子遅延に対して配線遅延を無視できる程度の広さをもつ物理的空間を近傍領域と呼ぶことにしよう．近傍領域の範囲は，素子速度や配線材料によって異なる．素子のスイッチング遅延と同程度の時間に光が到達できる範囲を近傍領域とすれば，たとえば，現在の標準的な1ナノ秒素子の場合には直径 30 cm 程度，1ピコ秒素子で直径 0.3 mm 程度の領域になる．

多数の異なる近傍領域からなる大規模システムではいかにして計算の局所依存性を実現するかが超高性能を達成する鍵になる．実際，信号処理や画像処理の分野では局所依存性を利用した並列計算アルゴリズムを実現するパイプラインアレイやウェーブフロントアレイ[37]にうまく写像できる問題がある．しかし，残念ながら，このようなアレイ構造にうまく適合する応用分野は限られている．

非同期式システムの最も著しい特徴は，その基本性能が計算に必要なデータ転送で生じる信号伝播距離の平均値で決まることである．このことは，ほとんどのデータ移動が近傍領域内で行われれば，個別には，近傍領域を越えた大きな範囲のデータ移動を任意に行うことができることを意味する．したがって，計算局所依存性の要求には柔軟性がある．また，同期式の場合には困難であった非常に処理遅延の大きい高機能モジュールを自由にデータ経路へ組み込むことができる．さらにパイプライン処理において各段階での処理時間が均一である必要はない．超高速プロセッサの実現へ向けて，このような非同期式システムの特徴を最大限に生かしたアーキテクチャを研究する必要がある．

6.5 メタステーブル動作

双安定素子が二つの安定状態のどちらとも判定できない中間状態に異常に長くとどまることをメタステーブル (metastable) 動作という[38),39)]．互いに独立な入力信号の組合せで双安定素子の状態を定める場合には，常にメタステーブル動作の起きる可能性が存在する．したがって，メタステーブル動作は，同期式か非同期式かにかかわらず，すべてのディジタルシステムに共通の問題である．たとえば，入出力部からプロセッサへの割り込み信号，異なるクロック系同士の交信，資源共有などはその顕著な例である．メタステーブル信号を異なる回路が受けた場合，その信号値に関して回路同士で互いに矛盾した認識が生じる可能性がある．その結果がシステムの誤動作を招く．

いかなるディジタルシステムの設計もメタステーブル動作発生の可能性をゼロにすることはできない．したがって，メタステーブル動作の発生確率を許容可能な水準まで低下させたアービタ回路の実現，及び，信号衝突機会を最小にするシステム構成を研究する必要がある．

7. むすび

スイッチング遅延1ピコ秒程度の超高速素子が実用化された場合の同期式プロセッサの性能限界を示し，その壁を打破する一つの可能性としての非同期式システムの理論と実際を概観し，超高速プロセッサを実現するために解決すべき研究課題を示した．

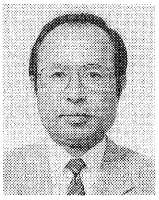
デバイス技術者の努力によって生み出されたピコ秒素子の高速性をフルに享受するシステムを設計することはシステム技術者，研究者の責任である．早急にそのような設計論を用意して，来るべき超高速素子の実用化を待つべきである．

参考文献

1) Seitz, C. L.: System Timing, in Introduction to VLSI Systems (Mead and Conway), Ch. 7, Addison-Wesley (1980).
2) 南谷 崇：論理回路の設計技法［V］；非同期式論理回路の設計，電子情報通信学会誌，Vol. 67, No. 3, pp. 301-307 (Mar. 1984).
3) Fishburn, J. P.: Clock Skew Optimization, IEEE Trans. Computer, Vol. 39, No. 7, pp. 945-951 (July 1990).
4) Kue, E. S. et al.: Timing Driven Layout, Proc. SASIMI '90, pp. 263-270 (Oct. 1990).
5) 南谷 崇：同期式プロセッサの限界と非同期式プロセッサの課題，電子情報通信学会技術報告，FTS 90-45, pp. 49-56 (Dec. 1990).
6) Nanya, T.: Challenges to Dependable Asynchronous Processor Design, Proc. Int. Symp. on Logic Synthesis and Microprocessor Architecture, pp. 132-139 (July 1992).
7) Langdon, G. G.: Logic Design, Academic Press, New York (1974).
8) Huffman, D. A.: The Synthesis of Sequential Switching Circuits, J. Franklin Inst., Vol. 257, No. 3, pp. 161-190 (Mar. 1954).
9) Muller, D. E. and Bartky, W. S.: A Theory of Asynchronous Circuits, Proc. Int. Symp. on Theory of Switching, pp. 204-243 (1959).
10) 木村 泉：非同期式回路の理論［I］-［III］，情報処理，Vol. 2, pp. 99-104, pp. 152-157, pp. 206-217 (1961).
11) Miller, R. E.: Switching Theory; Volume 2, John Wiley & Sons (1965).
12) Unger, S. H.: Asynchronous Sequential Switching Circuits, John Wiley & Sons (1969).
13) 南谷 崇：非同期式論理回路の理論［I］-［IV］，電子通信学会誌，Vol. 63, pp. 624-631, pp. 752-759, pp. 828-835, pp. 933-939 (1980).
14) Varshavsky, V. I.: Self-Timed Control of Concurrent Processes, Kluwer Academic Publishers (1990).
15) Meng, T. H.: Synchronization Design for Digital Systems, Kluwer Academic Publishers (1991).
16) たとえば，当麻，内藤，南谷：順序機械，岩波書店 (1983).
17) 小林幹典：同期式回路設計の考え方，トランジスタ技術，pp. 400-407 (1990. 11).
18) Martin, A. J. et al.: The Design of an Asynchronous Microprocessor, Decennial Caltech Conference on VLSI, MIT Press, pp. 351-373 (1989).
19) Udding, J. T.: A Formal Model for Defining and Classifying Delay-Insensitive Circuits and Systems, Distributed Computing, 1, Springer-Verlag, pp. 197-204 (1986).
20) Martin, A. J.: The Limitations to Delay-Insensitivity in Asynchronous Circuits, Advanced Research in VLSI (Proc. 6 th MIT Conf.), pp. 263-278 (1990).
21) Armstrong, D. B. et al.: Design of Asynchronous Circuits Assuming Unbounded Gate Delays, IEEE Trans. on Computers, Vol. C-18, No. 12, pp. 110-1120 (Dec. 1969).
22) Sutherland, I. E.: Micropipelines, C. ACM, Vol. 32, No. 6, pp. 720-738 (June 1989).
23) Dean, M. E., Williams, T. E. and Dill, D. L.: Efficient Self-Timing with Level-Encoded 2-Phase Dual-Rail (LEDR), Advanced Research in VLSI (Proc. 1991 UCSC Conf.), pp. 55-70 (1991).
24) McAuley, A. J.: Four State Asynchronous Architectures, IEEE Trans. Computers, Vol. 41, No. 2, pp. 129-142 (Feb. 1992).
25) 上野，南谷：2線2相2系方式による非同期式レジスタ間転送，電子情報通信学会技術研究報告，FTS 91-23, pp. 31-38 (July 1991).
26) 籠谷，南谷：プロセス記述による非同期式制御回路合成の一手法，情報処理学会設計自動化研究会 DA 60-10 (Dec. 1991).
27) Martin, A. J.: Compiling Communicating Processes into Delay-Insensitive VLSI Circuits, Distributed Computing, 1, Springer-Verlag, pp. 226-234 (1986).
28) van Berkel, C. H. and Saeijs, R. W. J. J.: Compilation of Communicating Processes into Delay-Insensitive Circuits, Proc. ICCD, pp. 157-162 (Oct. 1988).
29) Brunvand, E. and Sproull, R. F.: Translating Concurrent Programs into Delay-Insensitive Circuits, Proc. ICCAD-89, pp. 262-265 (Nov. 1989).
30) Chu, T. A.: Synthesis of Self-Timed Control Circuits from Graphs: An Example, Proc. ICCD, pp. 565-571 (Oct. 1986).
31) Chu, T. A.: Synthesis of Self-Timed VLSI Circuits from Graph-Theoretic Specifications, Ph. D Thesis, MIT (June 1987).
32) Meng, T. H.-Y. et al.: Automatic Synthesis of Asynchronous Circuits from High-Level Specifications, IEEE Trans. on CAD, Vol. 8, No. 11, pp. 1185-1205 (Nov. 1989).
33) Lavagno, L., Keutzer, K. and Sangiovanni-Vincentelli, A.: Algorithms for Synthesis of Hazard-Free Asynchronous Circuits, Proc. 28th DAC, pp. 302-308 (June 1991).
34) Vanbekbergen, P. et al.: Optimized Synthesis of Asynchronous Control Circuits from Graph-Theoretic Specifications, Proc. ICCAD '90, pp. 184-187 (Nov. 1990).
35) Nanya, T. and Kuwako, M.: On Signal Transition Causality for Self-Timed Implementation of Boolean Functions, Proc. 26th Hawaii Int. Conf. on System Sciences (Jan. 1993).
36) 南谷 崇：フォールトトレラントコンピュータ，オーム社 (1991).
37) Kung, S. Y. et al.: Wavefront Array Processors —Concept to Implementation, IEEE Computer,

Vol. 20, pp. 18-33 (July 1987).
38) Chaney, T. J. and Molner, C. E.: Anomalous Behavior of Synchronizer and Arbiter Circuits, IEEE Trans. Computers, Vol. C-22, No. 4, pp. 421-422 (Apr. 1973).
39) Marino, L. R.: General Theory of Metastable Operation, IEEE Trans. Computers, Vol. C-30, No. 2, pp. 107-115 (Feb. 1981).

(平成4年6月29日受付)



南谷 崇 (正会員)

1946年生. 1969年東京大学工学部計数工学科卒業. 1971年同大学院修士課程修了. 日本電気(株)中央研究所勤務を経て, 1981年東京工業大学工学部情報工学科助教授. 1989年同電気電子工学科教授. 主として, 論理設計方法論, 非同期システム論, フォールト・トレラント・コンピューティングに関する研究に従事. 工学博士. 1986年度電子情報通信学会論文賞. 著書「順序機械」岩波書店,「フォールトトレラントシステムの構成と設計」槇書店,「フォールトトレラントコンピュータ」オーム社など. IEEE, 電子情報通信学会, 情報通信学会各会員.