解説

GHzマイクロプロセッサ

青木直明　H. Peter Hofstee　Sang Dhong
IBMコーポレーション

KHzからGHzへ

1971年に，世界初の4ビットマイクロプロセッサ4004(750KHz)が登場して以来，今日まで多くのアーキテクチャが開発され，また半導体技術の進歩により，性能は着実に向上してきた．今年のISSCCで我々は，PowerPCのサブセット命令を実行する1GHz(ギガヘルツ)のマイクロプロセッサ“guTS(Gigahertz Unit TestSite)”の発表を行った[1]．動作周波数だけを見れば，27年で約1500倍になったことになる．

guTSプロジェクトの目的は，回路に焦点を当て，どこまで性能を上げられるか，を示すことであった．さらにいえば，将来の，マイクロアーキテクチャや半導体プロセス技術に改善の余地を残しつつも，回路設計によって，いかにマイクロプロセッサの性能を引き上げることができるかを実証することであった．guTSは，コアの部分だけであり，マイクロプロセッサ全体としての性能を引き出すには，メモリとのインタフェースや周辺の回路だけでなく，さらに複雑なシステムとして，最適化する必要がある．ただ，コアとして非常に大きな可能性を持っているといえるだろう．本稿では，まずguTSプロジェクト全般について，開発経緯について述べた後，マイクロアーキテクチャ，回路設計，CAD，テストの点から解説する．最後に今後の課題，guTSの与えた影響などを考察する．

guTSプロジェクト

1995年に戻る．多くのコンピュータが，まだ100MHzから133MHzで動作しているころに，IBMは，将来の技術的な種として，1GHzのアイデアを採用し，研究所をAustinに設立した．だが実際，プロジェクトがスタートした時，山のような難題が目の前に置かれていた．図-1は，マイクロプロセッサを取り巻く技術的課題を示している．

回路，クロック，テスト，消費電力など，1GHzという動作周波数を考えた場合，そのどれもが大きな壁であった．実際，最も難しかったのは，起こりうる可能性のある膨大な問題の中から，本当の問題を区別することであった．“解決しなければならない本当の問題は何か”，具体的な姿が浮かんでくるまでには，かなりの議論と時間を要した．

また，これは技術的課題ではないが，スタッフィング(人材の獲得)も大きな問題であった．Austinに研究所を設立した理由は，この地がIBMを代表するワークステーションRS6000の開発拠点であると同時に，またAMD，MOTOROLAなどを始めとするマイクロプロセッサ関連の人材が豊富だからである．にもかかわらず，すぐには，人材は揃わなかった．このことは，この業界の状況を示している同時に，このアイデアが，あまりにも現実離れしていたためともいえる．実際，“できるはずがない，いや仮に設計できたとしても動かないだろう”と考える者は少なくなかった．マイクロプロセッサの設計といえば，通常，大プロジェクトである．あるプロセッサの開発では，数百人の設計者を動員したという報告もある．もちろんプロセ

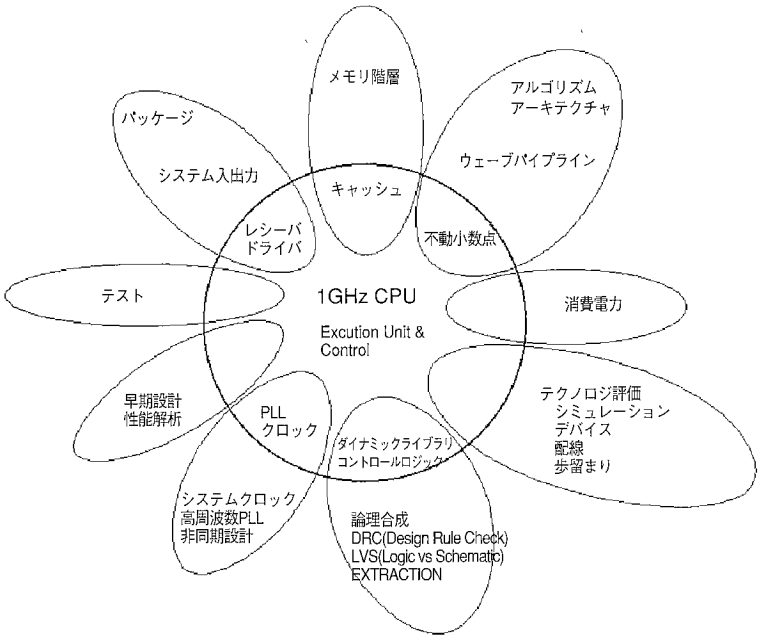

図-1　マイクロプロセッサの技術的課題

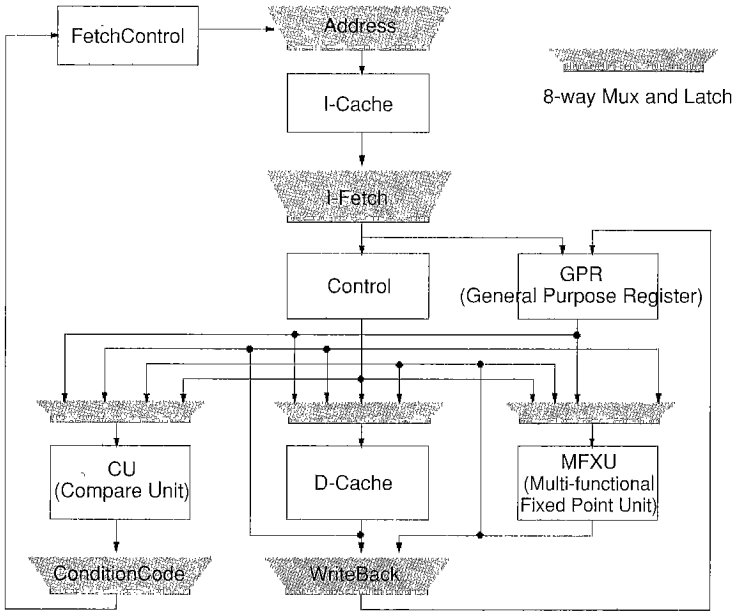

図-2　guTSの構成

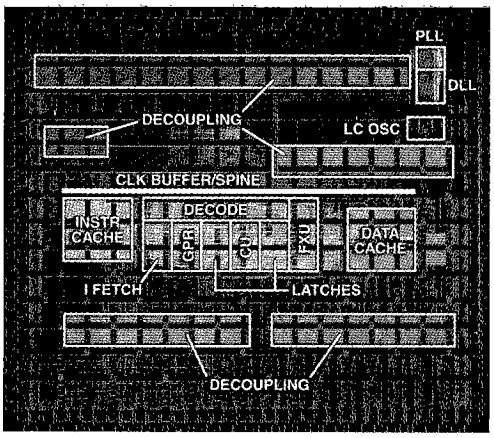

図-3　1000MHz 64-bit Interger Processor

ッサの規模にもよるが，guTSの設計は，High Performance VLSI Designのグループと他グループのエンジニアを含め，わずか十数人のチームで行われた．

当初は，各マクロの評価が目的であったが，年が明けて1996年から，IBMの他のサイトや大学から徐々に人材が揃い，1996年の夏には，プロセッサとして設計することが決定した．そして1997年の1月末，テープアウトし，Eastfishkill, NYのASTC(Advanced Semiconductor Technology Center)で製造された．その年の夏，Yorktown Heights, NYの T.J. Watson Research Centerでテストを行い，今回の発表に至った．

■アーキテクチャ

guTSは，PowerPCの64ビット命令セットの内，96個の命令を実現している．それらは，算術演算，論理演算，シフト・ローテート，比較，分岐，ロード・ストア等である．省略されたのは，乗算，除算のような数個の整数演算命令と，“Load String”のようなロード・ストア命令，それとアドレス変換・入出力・浮動小数点などの命令である．guTSの命令セットは，多くのアプリケーションで実行されている命令の少なくとも90%はカバーしている．

guTSのマイクロアーキテクチャは，以下の目標を目指して決められた．

- 条件分岐が成立する場合を除き，すべての命令を1サイクルで実行する．
- 条件分岐が成立した場合のみ，ストールを行う．
- 4段パイプライン(フェッチ，デコード，実行，書き込み)

また直前の命令によって計算された条件に依存する分岐によるストールは，わずか1サイクルに抑えられている．

さらに，いくつかのフォワーディング(またはバイパス)経路を省略することによって，性能を犠牲にしてまで，動作周波数を上げるということはしたくなかった．したがって，すべてのフォワーディングを実現している．このように周波数だけでなく，複雑な命令に対しても，高い目標設定を行った．コントロールはマルチプレクサ・ラッチへのセレクト入力という形で与えられている．これは，確かに論理設計の上で制約を課すことになるが，多くの利点を持っている．

- クロック・スキューに対するペナルティーは，サイクルに一度だけ払われる．
- ラッチからラッチへの経路が，回路設計者に最適化の経路として与えられる．
- 設計上のインタフェースが簡略化される．
- サイクルの終わりでのセレクトは，コントロールに最大限の時間を与えられると同時に並列性を高められる．
- スキャンのためのペナルティーは，マルチプレクサ・ラッチのわずか1ポートである．

図-2に，guTSの構成，図-3にチップのフロアプランを写真で示す．guTSは，シングルイシュー・インオーダーで4段のパイプライン構造をとる．キャッシュは，データ，インストラクションとも4KBで，ダイレクト・マップド方式を採用している．レジスタファイルは，32エントリ，3リード・2ライトである．

■回路

“最初に回路ありき．”これが，このプロジェクトを通して一貫していることであった．実際，すべての論理設計は，デバイスレベルの回路(CMOS回路)に何

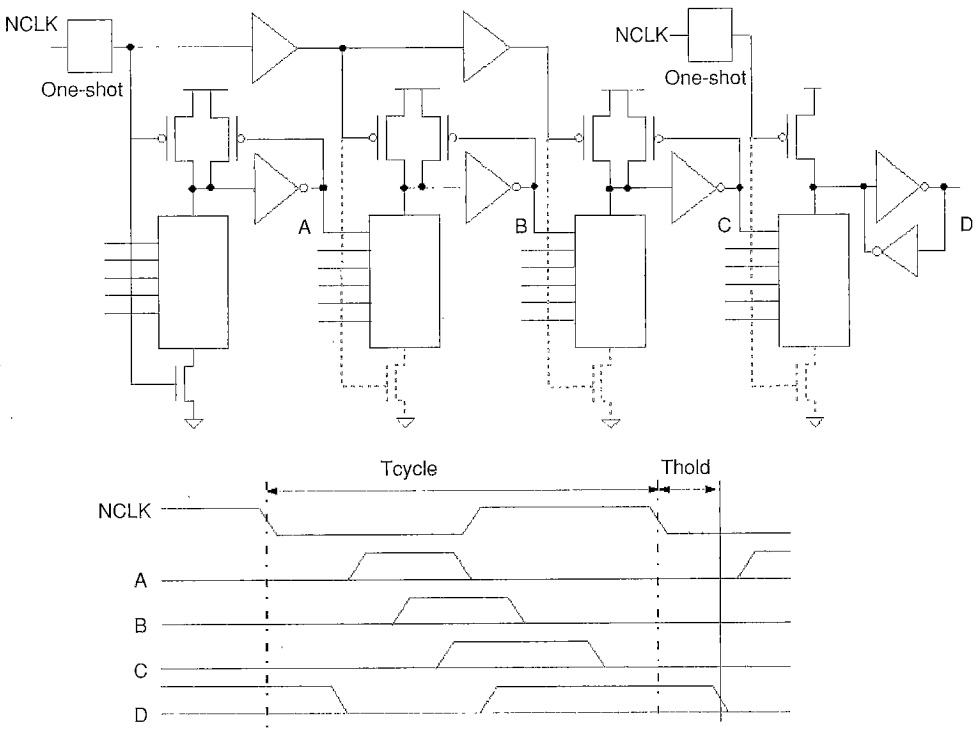

図-4 Delayed-Reset Domino回路

ができるか，を十分に知っている回路設計者によって行われた．これは，データフローのマクロを，スケマティック・エントリ(デバイスレベルの回路図入力)と回路シミュレータを使って設計するのが回路設計者の役割，一方，抽象的な“ロジックレベル”を設計のかなり後半まで，ハードウェア記述言語を使用しながら，最後にシンセシス(論理合成)によってコントロールの回路を設計するのが論理設計者の役割といった現場とは，多少異なるかもしれない．

マイクロプロセッサの設計の場合，マイクロアーキテクチャ，論理設計，回路設計，レイアウト設計を各々独立したグループとして最適化する傾向がある．これは，確かに各グループの生産性を高めるかもしれないが，guTSは，一般にこうしたアプローチが，与えられたテクノロジーの最大到達可能点まで到達していないことを示したといえるだろう．典型的な設計において，コアの周波数は2倍くらいまで高められると我々は，見積もっている．

GHzを超えるような，きわめて高い動作周波数を達成するには，コントロールロジックの設計者が早い段階で，他の回路のインプリメンテーションについて十分理解していなければならない．コントロールロジックの設計において，シンセシスを用いる問題としては，生成されたトランジスタのサイズやレイアウトが必ずしも十分ではないということ，また，一般に論理設計者が，それらが定義されている回路の物理的な側面から離れている場合が多いことなどがあげられる．このことが，ロジックの変更，シンセシス，配置配線の頻繁な繰り返しを起こしているのかもしれない．guTSは，センスアンプを使った回路群[2]と先人のいくつかの仕事を基に，半年間と数十人で高度の最適化を行いながらも，設計時間を犠牲にすることなく，また法外なデザイン・リソースを必要とすることもなかった．言い換えれば，いかに限られたリソースで，効率よく設計を行うかもまた，我々にとってチャレンジであった．センスアンプを使った回路群では，設計時間を有効に使うため，デコーダ，プリチャージ回路，コントロール回路等，多くの回路を大幅な変更なしに共有している．

1nsのサイクルタイムでは，せいぜい14のインバータ，あるいはダイナミック回路において7段のロジックが可能であった．チップ全般において，ダイナミック回路を広範囲に使用している．ダイナミック回路は，高速かつ面積効率的な設計を可能にするからである．図-4に基本的な回路を示す．段数を減らすためには，各段で，最大限の機能が実現されなければならなかった．したがって，いくつかの機能を1段で実行できるよう，ロジック，トランジスタおよびワイヤを，共有できるところは共有させ，できるかぎりコンパクトな設計になるよう注意を払った．

ダイナミック回路の性格上，信号はパルスになる．パルス幅，リセット(プリチャージ)，そして評価のタイミングを注意深く，取り扱いながら，通常ドミノ回路において用いられている接地との間のインタラプト

デバイス(図-4の点線部)をできるだけ取り除いた．これで10%から15%程度の改善が得られた．しかし，一方でこのことは，タイミングを誤れば，短絡の問題を引き起こす．設計では，この問題を回避しながら，またレイアウトの違いによって生ずる拡散層の容量を考慮しながら，最大限の性能を引き出すために，多くの時間を費やし，調整，最適化を行った．最早，ナノ秒(10億分の1秒)という単位は，設計の現場では使われていない．ピコ秒(一兆分の1秒)が，最小単位である．guTSの中で使われている回路の多くは，タイミングがすべてである．タイミングが誤れば，完全な動作は望めない．回路シミュレーションには，プロセスが変化しても動作を保証するため，ASXと呼ばれるIBMの回路シミュレータを使用した．

ダイナミック回路が，スタティック回路よりも消費電力が多いというのは，きわめて高い動作周波数領域でなければ正しい．しかし，この議論は，非常に高い周波数を実現する上では，あまり意味をなさなくなる．スタティック回路では，プロセスあるいはデバイスの技術革新なしに，このような高い動作周波数の要求を満たすことは，ほとんど不可能だからである．guTSの消費電力は，1GHz，1ミリオントランジスタで6.3ワットである．この値は，毎サイクル，キャッシュにアクセスしていることと，性能を追求するために，きわめて大きなトランジスタをチップ全般にわたり使用していることを考えると，妥当な値だと信じている．

ハイパフォーマンス回路の設計とCAD

理想的な世界では，回路設計者に課せられる制約は，次のような外から見えるマクロの動作に関するものだろう．

- 機能仕様
- 入力信号タイミング，ノイズ耐性
- 出力信号タイミング，駆動能力
- 消費電力，面積
- 外部環境，温度特性，電源電圧特性
- 信頼性

だが，現実には，上で述べたような直接的な制約以外に，設計する上で，どういった回路(ダイナミックかスタティックか)を使うか，ある特定のライブラリの使用や，クロックの使用に関する制限，設計マージン等の制約がある．またタイミングやレイアウトおよびインテグレーションといった多くのツールが，個々のツールに固有の事前に設定されている付随的，間接的制約を課す．たとえば，タイミングベリフィケーションやタイミングルールの生成などにおける共通の制約は，1つの入力の変化が，唯一の出力の変化を起こすということである．このことは，1つのクロックあるいはデータから，いくつかのパルスを生成するような回路の使用を制限する．また，ディジタル回路であっても，動作周波数が高くなればアナログ的な振る舞いが出てくる．データの依存性や他の信号が早くきているか否か，あるいは同時にくるのかといった違いによって出力信号のタイミングが数十ピコ秒も変化するケースがある．タイミングルールでこれらのことをチェックするのは，非常に困難だ．

もう1つの例は，ノイズの解析である．個々の回路から，ノイズ源となり得る個所を特定し，どのように伝播するか，回路間のインタフェースを解析するというよりは，一律に最悪値を想定し設計を行う．

こうした制約が，回路を最適化する上で，設計者にとって，時に取るに足らない問題でありながら，そのことが，達成可能な性能に深刻に影響を与えることがある．

guTSは，もちろん理想的な世界で設計されたわけではない．目標は，回路設計上，特に性能に大きく影響を与える制約は，できるだけ少なくするということであったが，実際にはいくつかの制約を課した．主な制約は，すべての出力信号は，疑似スタティック(マクロ内での信号はパルスだが，ラッチへの出力は，伸長されている)で，クロック周波数に依存することなく，ラッチのホールドタイムを満たすことを保証するよう設計された(図-4のThold)．信号到達時間とデータパターンにおける不確実性をチェックする方法は欠いたが，遅延対パルス幅のチェックは行った．これは，たとえば，プリチャージの期間中に，適切な時間内にダイナミックノードがプリチャージされているかをチェックする．他のチェックはPMOSとNMOSの比率などである．理想的な方法なら，信号の規準を直接チェックできるだろう．電源バスの解析はデザインフローには統合されていなかったが，一律10%の低下を回路設計の際に適応した．こうした制約のなか，さまざまなタイプの回路を使っている．Delayed-Reset Domino(図-4)，Self-Resetting Domino，センスアンプ，シンプルなスタティック回路等である．こうしたさまざまな回路を使用することは，間接的な制約をチェックするよりも時間がかかる傾向がある．しかしながら，単純な疑似スタティックインタフェースにより，グローバルなチップのタイミングベリフィケーションにも，通常以上の時間はかからなかった．これまで述べてきたように，グローバルな制約として，どういった回路方式を使うか，ということに制約はいっさい課さなかった．

また，これらのマクロがいかに複雑であろうとも，インテグレーション時には，この複雑さがほとんど外からは見えないということも重要であった．多くの最適化は，通常適切なツールによってその恩恵を受けるところだが，今回このプロジェクトでは，そうしたツールは開発されていなかった．にもかかわらず，最も複雑なマクロのいくつかは，スクラッチから，6カ月

以内で設計された．

■クロックの設計

guTSでは，1本のクロックバスが，データフローのマクロとコントロールのマクロの間を背骨のように走っている．そこから，各マクロへクロックが，魚の骨のように分配されている．PLLの最大周波数は1,560～1,570MHz，ジッター(位相差を比較補正する際に生ずる微妙な周波数の揺らぎ)はPk-Pkで71ピコ秒に抑えられている．クロックスキューは，中央のクロックバスで50ピコ秒である．

最終的な変更で，各マクロへ分配されるクロックのスキューは，手作業により，当初の半分にまで抑えられた．また，ラッチへの配置，配線プログラムによって，意図せず生じたクロックの遅延のいくつかは，結果的に非常に洗練されたサイクルスティーリング(各実行ユニットの処理時間の違いを，ラッチのタイミングを調整することによって，各パイプラインのオーバーヘッドを除く技術)として適用された．このことは，スキューをゼロにするために，独立してクロックの分配を最適化することが，必ずしも最善の結果を生むとは限らないことを述べている．

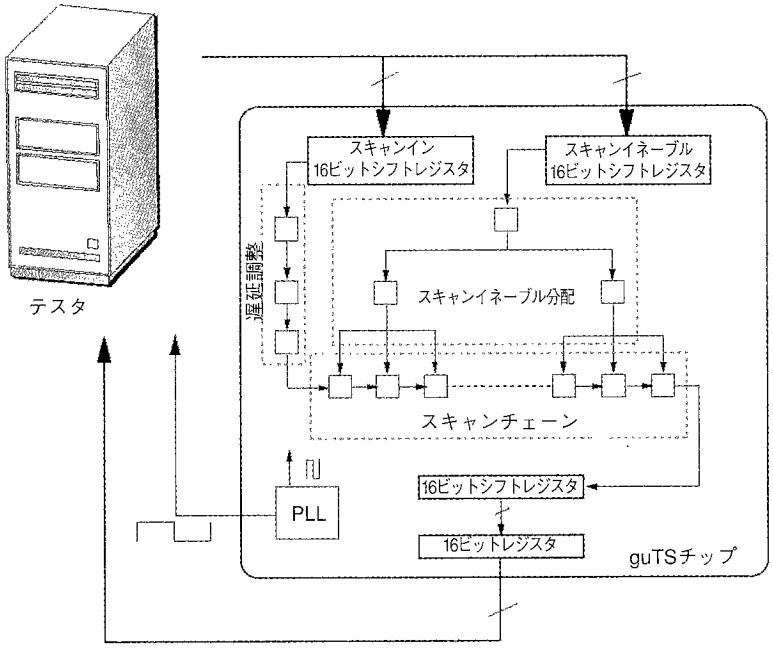

図-5　テストインタフェース

■テスト

guTSは，IBMのCMOS6Xのラインで製造された．CMOS6Xテクノロジーの主な特徴は以下の通りである．

- リソグラフィ：0.25um
- 実効チャネル幅：0.15um
- 酸化膜圧：4.0nm
- 電源電圧：1.8V
- 配線層：6層アルミニウム

CMOS6Xは1997年時におけるIBMの量産ラインである．最初に述べたように，このプロジェクトの目的は，先進的な回路による性能向上を狙ったものであり，先進的な製造ラインを通してのものではない．

テストは，ウェファー・プローバーで(ウェファーに直接，専用の針を接触させることによって)行われた．guTSの目的は，1GHz以上での動作確認であったため，新たにテスト方法が考案された[3]．我々が使えるテスタの最大周波数は660MHzで，そのままでは，使えなかったからである．

基本的には，パラレルーシリアルインタフェースを採用している．**図-5**に，テストインタフェースを示す．チップ内部のPLLから生成された1GHzのクロックを16分の1にして，テスタへ送る．テスタはこのクロックに同期して，テストパターンを生成する．チップ内には，2個の16ビットロード機能付きのシフトレジスタを用意しておき，テスタから送られてくる16ビットのデータを16ナノ秒ごとにロードする．シフトレジスタは，他の回路同様1GHzで動作し，一方のシフトレジスタは，スキャンインにデータを転送する．他方のシフトレジスタは，各ラッチのスキャンイネーブルに，パイプラインのようにいく段かのラッチを通して，コントロールの信号を転送する．

guTS内の各マクロは，スキャン機能を使い，当該データをロードし，次のサイクルで機能テストを実行後，スキャンアウトを通して確認された．また，キャッシュにデータをスキャンロード後，スキャンアウトまで，数千サイクルを要するテストも行われた．また，guTSとは別に，各マクロは必要な回路と共に，単独で評価が可能なよう同じチップに集積され，これらもテストされた．

最後に，“e”の最初の13桁を計算するテストを，1ビットのエラーに非常に敏感な，整数演算のみのアルゴリズムを使って行った．このことは，ウェファープローバーという，インダクタンスやコンタクトといった電気的な点で，必ずしも十分とはいいきれないテスト環境下で，guTSが間違いなく動作していることを確信させた．

図-6に，主要なタイミングのテスト結果を示す(図-2参照)．

上図において，Repower and Distributeは，重い負荷を駆動するために挿入したバッファ等の回路を指す．Source Jitterは，クロックの設計で述べたPLLのジッターのことを指す．

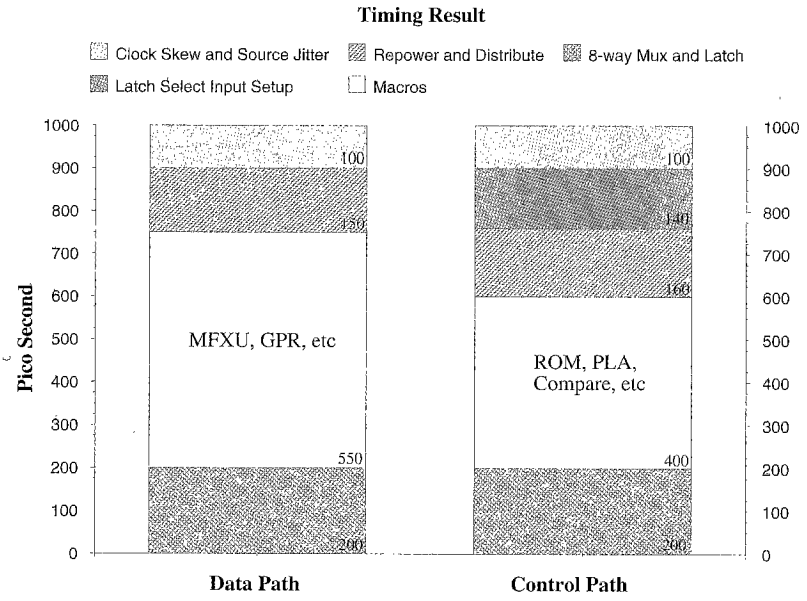

図-6　タイミングのテスト結果

今後の展開

guTSは，整数演算ユニットのみのインプリメンテーションであり，いわゆるマイクロプロセッサとしては不十分である．浮動小数点ユニットやバスインタフェース等の周辺回路，またスーパースカラ，VLIW等のアーキテクチャも含めてGHzプロセッサの実現について考えていかなくてはならない．同時に，今回開発したguTSの，コアとしての利用を考えていく必要がある．たとえば，メモリを強化してサーバ向けに，また消費電力と周波数のバランスをとることよって，組み込み用途向けにさまざまな応用が考えられる．

言い換えれば，guTSのような高い動作周波数のプロセッサの長所を最大限に利用しながら，いかにシステムとしての性能を引き出すかである．たとえば，高い動作周波数を維持しながら，スーパースカラ等の複雑なアーキテクチャを実現するのは，容易ではない．自ずと単純なアーキテクチャになる．しかし，このことはビジネス用途のマシンを考えるならば，それほど短所にはならない．ループ文の繰り返しが少なく，ループ文以外の分岐命令が多く，スーパースカラの恩恵を受けにくいからである．またビジネス用途のアプリケーションは，スーパースカラに合わせて，プログラムを最適化するのが，相対的に難しい[6]．より重要なのは，メモリの階層である．guTSのキャッシュは，インストラクション，データとも4KBであり，十分とはいえない．性能を引き出すには，大容量のキャッシュが必要であり，連想性も高める必要があるだろう．L1キャッシュ，L2キャッシュ，L3…，主記憶，そしてハードディスク装置といったメモリ・サブシステムまで，いかに実現するかを考える必要がある．つまり，マイクロプロセッサとメモリの周波数のギャップを，いかに無駄なく埋めるかである．具体的には，どこまでオンチップ化するのか，インタフェースをどうするのかが，今後，検討すべき課題である．

もちろん，複雑なアーキテクチャを，高い動作周波数で実現することも重要な課題の1つである．最後に，CADへの期待も大きい．guTSは，ほとんどのマクロ，末端のクロックのラインですら，手作業によって最適化された．半導体プロセスの進歩と共に，チップ上のトランジスタ数もますます増加し，設計コストの上昇，消費電力も大きな問題となっている．設計の複雑度は，集積度と周波数の向上と共に増している．各設計段階での適切なツールと上位設計から下位設計までシームレスな環境が望まれる．

GHzを超えて

guTSは，量産ラインを使ってわずか十数人というチームで設計された．さらに多くの人数を使って，最適化を行えば10%程度の改善は可能だろう．また次世代のCu配線等の先進的プロセスを使えば，20%程度の改善が見込まれる．guTSは，半導体プロセス，アーキテクチャの進歩だけでなく，回路とその最適化，設計方法によっても，将来のマイクロプロセッサに，かなりの性能改善の余地があることを示した．また，GHzという数字が，考えられていたような難しい壁ではないこと，マイクロプロセッサのクロックスピード，サイクルタイムは1GHzを超えて，今後も伸びていくことを証明したといえる．

コンピュータは，少なくとも今後10～20年は継続して半導体技術の発展と共に進歩しつづけ，特に，現在コンピュータの性能によって制限されている解決困難な領域の問題を解くと共に，より我々の日常生活の中に入り込んでくるであろう．

参考文献
1)Silberman, J. et al. :A 1.0GHz Single-Issue 64b PowerPC Integer Processor, IEEE International Solid-State Circuits Conference 1998 Digest of Technical Papers, IEEE Press, pp.230-231(1998).
2)Takahashi, O., Aoki, N., Silberman, J. and Dhong, S. : 1GHz Logic Circuits with Sense Amplifiers, Proc. VLSI Symposium on VLSI Circuits, to appear(1998).
3)Heidel, D. et al. : High Speed Serializing/De- Sirializing Design-For-Test Method for Evaluating a 1GHz Microprocessor, Proc. 16th IEEE VLSI Test Symposium, to appear(1998).
4)Hofstee, H. P. et al. : Designing for a GigaHertz, IEEE MICRO 1998, to appear.
5)Microprocessor Report, pp.9-10(Mar. 9 1998).
6)日経エレクトロニクス, No.627(Jan. 30 1995).

（平成10年4月30日受付）