計算機アーキテクチャ 126－20
（1997．10．29）

データパスの特性を考慮した非同期式制御回路の一設計手法

桑子雅史[†] 石川誠[††]
上野洋一郎[††] 南谷崇[†]

実際の非同期式プロセッサにおける制御回路の設計仕様は、QDIモデルに基づく回路を実現するのには不完全なことが多い。QDI回路を実現するためには、冗長な仕様を追加する必要がある。本稿では、冗長な仕様を追加する段階においてデータパス回路の処理遅延特性を考慮することより、信頼性と速度性能の両立した制御回路を設計する一手法を示す。

A design method for asynchronous controllers using data-path delay information

MASASHI KUWAKO,[†] MAKOTO ISHIKAWA ,[††] YOICHIRO UENO[††]
and TAKASHI NANYA[†]

Specifications for designing asynchronous controllers are often "incomplete" to implement QDI circuits. We show a modification method to obtain a complete specification from an incomplete one. In order to assure both high-speed operation and high timing-reliability in the resulting circuits, some causal relations are added to a given incomplete specification based on timing information expected.

1. はじめに

近年、高速なスイッチングデバイスの素子性能を十分に享受する一手法としてクロック信号を用いない非同期式論理設計が注目されている[1]。

我々は非同期式論理設計によって高速なプロセッサを実現する方法を明らかにするために32bit汎用非同期式プロセッサ"TITAC-2"[2)3)4)]を0.5μmルールのスタンダードセル方式C-MOSを用いて設計・製作した。

TITAC-2全体のパイプライン制御では、FIFO構造のステージ間ラッチを利用することによって、データパス回路そのものが制御回路としての機能も持っている分散制御方式を採用している[2]。これに対してTITAC-2内の命令キャッシュでは、一般的に用いられているような、データパス回路と制御回路が分離しており ある機能を実現する複数のデータパス資源(演算回路や記憶回路)を単一の制御回路によって制御する集中制御方式を採用している。

† 東京大学 先端科学技術研究センター
Research Center for Advanced Science and Technology, University of Tokyo
†† 東京工業大学 情報理工学研究科
Graduate School of Information Science and Engineering, Tokyo Institute of Technology

本稿では、TITAC-2の命令キャッシュ内のブロック転送回路を例として、TITAC-2の設計にも用いた、信頼性と速度性能を両立させる非同期式集中制御回路の一設計手法を示す。

2. 非同期式回路の遅延仮定

非同期式回路を考える上では、論理ゲートや配線の遅延について何らかの仮定を設けなければならない。この仮定は遅延仮定と呼ばれる。遅延仮定にはさまざまなものが提案されている。

設計された回路のタイミング・フォールト(素子・配線の遅延時間が回路の設計段階や実際の動作時において設計者の設けた仮定から外れること)に対する耐性、即ち、タイミング信頼性は、どのような遅延仮定に基づいて論理設計を行なうかに影響される。

一方、遅延仮定は速度性能にも影響を与える。一般に、タイミング信頼性の高くなるような厳しい遅延仮定に基づいた非同期式回路の速度性能は低くなり、回路量は多くなる。

従って、非同期式回路の設計においては速度性能とタイミング信頼性とのトレードオフによって遅延仮定を決定する必要がある[5]。

最も厳しい遅延仮定として知られているのは、DI(Delay Insensitive)モデル[6]である。このモデル

は純粋遅延[7]と慣性遅延[7]とが混ざり合った浮遊遅延が論理ゲートの出力及び配線に存在する。そしてこれらの遅延は有限ではあるが、その上限値は未知であると仮定される。この遅延仮定に基づいて設計した回路は、タイミング・フォールトを起こさない。しかし、Delay–Insensitive モデルの下で実現可能な回路は、ごく限られた機能のものだけである。

DI モデルに等時分岐 (Isochronic Fork) の仮定を設けることで任意の機能を持つ回路を設計することができる[8]。等時分岐とは、配線に分岐がある場合その分岐した先の配線部分の伝搬遅延時間は全て等しいとする仮定である。DI モデルに等時分岐の仮定を加えたモデルは QDI(Quasi–Delay–Insensitive) モデルと呼ばれている[9]。

3. 要求–応答方式と 2 相式動作

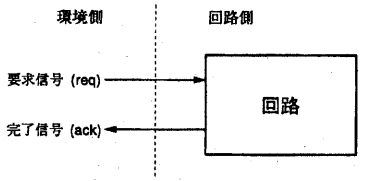

図 1　要求–応答方式

QDI モデルのような厳しい遅延仮定に基づく回路は、図 1に示すように動作要求信号 req が入力されると要求された処理を行ない、処理が終了すると動作完了信号 ack を返すという要求–応答方式で動作する必要がある。

要求–応答方式の実現の一つに 2 相式と呼ばれる方式がある。req が 0 から 1 になると回路が動作を開始し、それが完了すると ack が 0 から 1 になる。これは稼働相と呼ばれる。環境が ack=1 を確認して req=0 とすると回路は ack=0 を出力して次の稼働相を実行する準備ができたことを環境に知らせる。これは休止相と呼ばれる。このように稼働相と休止相を交互に繰り返すことで処理を実行するため 2 相式と呼ばれている。TITAC–2 では、プロセッサのほとんどの部分で 2 相式を採用している。

4. 非同期式制御回路の設計手法

QDI モデルに基づく制御回路を設計する手法として、STG(Signal Transition Graph)[10] と呼ばれる動作仕様記述から回路を合成する手法が多く研究されている。

STG では、変数 x の 0→1 の遷移を x^+ と表わし、1→0 の遷移を x^- と表わす。そして各変数の遷移間の因果関係を有向枝を用いて表わす。例えば、変数 x が 1 になると変数 yが 0 から 1 に遷移するのであれば、$x^+ \rightarrow y^+$ と表わす。

STG から回路を生成する研究のほとんどは、回路を生成するために必要な Complete–State–Coding と呼ばれる性質を満たす STG を対象としている。

Complete–State–Coding[11]　STG が以下のいずれかの条件を満たしているとき、その STG は Complete–State–Coding の性質を満たしていると言う。

- 全ての到達可能状態が異なる 2 進コードを持つ。
- 2 つ以上の状態が同一の 2 進コードを持つ場合、それらの状態において遷移可能な内部信号と出力信号は等しい。

実際のプロセッサにおける制御回路の設計においては与えられる要求仕様がこの条件を満たしていることは少なく、回路を実現するために設計者が冗長な仕様を追加する必要がある。しかし、冗長な仕様を追加する方法に関する研究は少ない。

我々が今回提案する非同期式回路の設計手法は、STG を仕様として QDI モデルに基づいて制御回路を設計する手法をベースとするものである。冗長な仕様を追加する段階において、制御される対象の回路の遅延特性を考慮することにより信頼性と速度性能を両立させる。

提案手法による設計の大まかな流れは以下のようになる。

(1)　制御対象となる論理演算回路·記憶回路などの回路資源の個々の動作仕様と、それらを結合したときの全体の動作仕様を決定する。
(2)　全体の動作仕様を、各資源の動作開始要求信号、動作完了応答信号などを用いて STG で記述する。
(3)　データパス回路の遅延特性を考慮しつつ仕様の STG に冗長な因果関係を追加することで、回路を生成することが可能な STG を得る。
(4)　QDI モデルに基づく制御回路を (既存の手法を用いて)STG から論理合成する。

以下では、TITAC–2 の命令キャッシュ内のブロック転送回路を具体例として、提案する設計手法を説明する。

5. TITAC–2 の命令キャッシュのブロック転送回路

命令キャッシュでキャッシュミスが発生したときには、メイン·メモリからキャッシュ·メモリへキャッシュ·ラインサイズに相当する回数のデータの転送が行なわれる。TITAC–2 においては図 2に示した回路でブロック転送を実現している。

この図は、ブロック転送に関連する信号のみしか表記していない簡略化したものであり、実際の TITAC–2

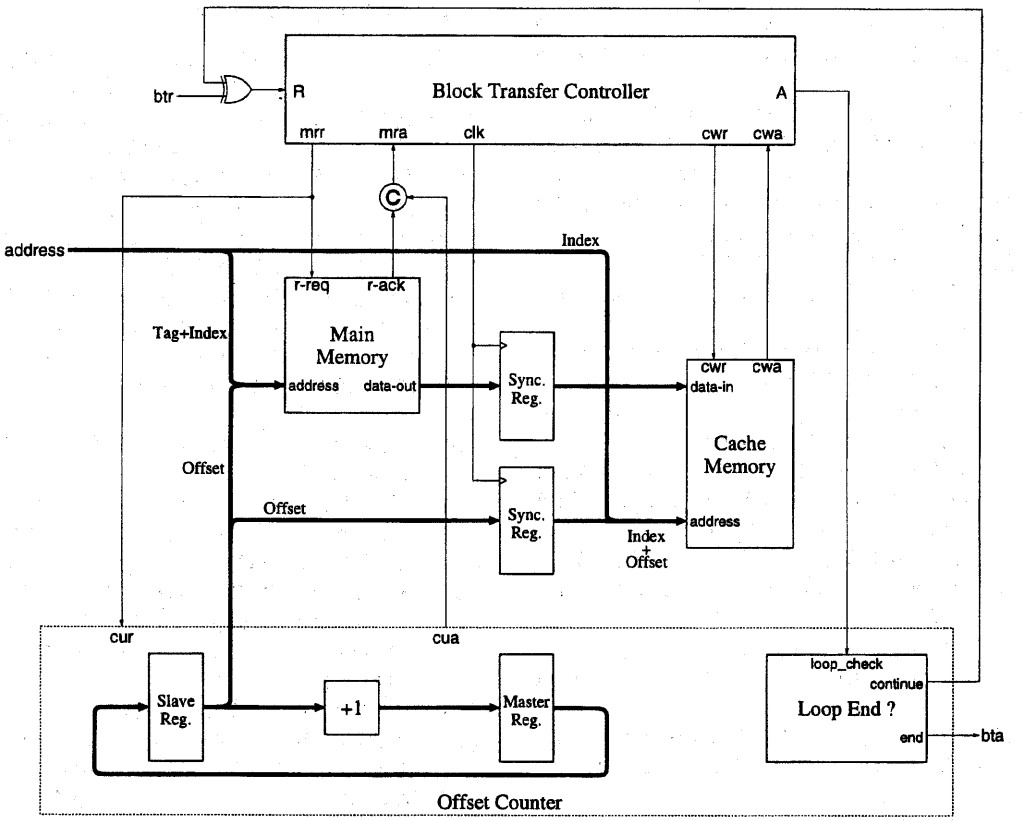

図2　ブロック転送回路の構成

では図2よりも複雑な構成になっている。

図中の Main Memory はチップ外部に接続された主記憶と、これを要求-応答方式で動作する回路と見なせるようにするためのチップ内部のインターフェース回路が一体となった回路ブロックである。入力にアドレスを与えた状態で r-req=1 とすると読み出し動作が行なわれ、出力データが安定したことを示す信号として r-ack=1 が出力される (メイン・メモリの稼働相)。r-ack=1 の後、r-req=0 とすると r-ack=0 となり、次の読み出し動作の準備ができたことを制御回路に知らせる (メイン・メモリの休止相)。

図中の Cache Memory はチップ内部の同期式RAMマクロ[12]と、これを要求-応答方式で動作する回路と見なせるようにするためのインターフェース回路が一体となった回路ブロックである。入力にアドレスと書き込みデータを与えた状態で cwr=1 とすると書き込み動作が行なわれ、書き込みが完了したことを示す信号として cwa=1 が出力される (キャッシュ・メモリの稼働相)。cwa=1 の後、cwr=0 とすると cwa=0 となり、次の書き込み動作の準備ができたことを制御回路に知らせる (キャッシュ・メモリの休止相)。同期式RAM マクロの特性により、cwr=1 から cwa=1 まで と、cwr=0 から cwa=0 までの時間は、それぞれ約 7ns 必要となっている。

図中の Offset Counter は、キャッシュのラインサイズに相当する回数のブロック転送を行なうためのカウンタである。このカウンタはマスタースレーブ方式で実現されている。メイン・メモリやキャッシュ・メモリに与えるアドレス入力の下位ビットを生成するとともに、制御ループの終了判定の機能も持つ。cur=1 となると、Slave Register の値がインクリメントされ Master Register へ転送される。転送が終了すると cua=1 となる (オフセット・カウンタの稼働相)。cur=0 となると、Master Register から Slave Register への転送が行なわれて Offset Counter の出力値が更新される。更新が終了すると cua=0 となる (オフセット・カウンタの休止相)。

図 2の回路では、メイン・メモリからの読み出し及びオフセット・カウンタのインクリメントと、キャッシュ・メモリへの書き込みが 2 段パイプライン化してある。このパイプライン動作を実現するためのステージ間ラッチが図中の 2 つの Syncronous Register であ

る。メイン・メモリから読み出されたデータ、このデータを書き込むべきキャッシュ・メモリ・アドレスのオフセット部分をそれぞれ記憶する。clk はレジスタのクロック入力である。

また、図中で©で表わされている素子は、Muller の C 素子[13]と呼ばれるものである。これは、全ての入力が1となると出力も1となり、全ての入力が0となると出力も0となるような記憶を持つ素子であり、非同期式回路においてよく用いられる。

図2の回路全体の動作は以下の通りである。

(1) ブロック転送開始の要求入力である btr が1となる。
(2) メイン・メモリの読み出しとオフセット・カウンタの稼働相が行なわれる。
(3) メモリから読み出されたデータとオフセットがステージ間ラッチに記憶される。
(4) メイン・メモリとオフセット・カウンタの休止相が実行され、オフセット・カウンタの値が更新される。これと並行してキャッシュ・メモリへの書き込みの稼働相及び休止相が実行される。
(5) 繰り返しループの終了判定が行なわれ、終了ならば bta=1 となる。そうでなければ、(2) へ戻りブロック転送を続ける。
(6) bta=1 となると、btr=0 となる。そして bta=0 となる。

以上の動作を実現する Block Transfer Controller の動作仕様は、図3の STG で表わすことができる。

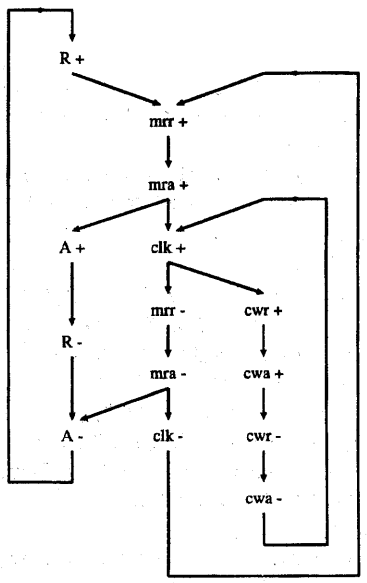

図3 Block Transfer Controller の動作仕様を表わす STG

6. STG からの制御回路の実現

図3の STG は、Complete-State-Coding の性質を満たしていないため、そのままでは QDI モデルに基づく回路を実現できない。回路を実現するためには、この "不完全な仕様" に新たな因果関係を追加することで "完全な仕様" に修正する必要がある。

因果関係を追加する箇所の発見には、STG が Persistency と呼ばれる性質を満たすことが、回路を生成できる十分条件であることを利用する。

Persistency[14] 遷移 s^* が遷移 t^* の原因遷移となっており、s^* の相補的な遷移 $\overline{s^*}$ が t^* の発火の後にのみ起こるとき、s^* は persistent であると言う。

因果関係を追加する箇所の発見手順は以下のようになる。

(1) 任意の分岐箇所に着目する。未検査の分岐箇所が無ければ終了。
(2) その分岐において persistency が満たされているかどうかを検査する。満たしていればこの分岐箇所については終了。(1) に戻る。
(3) 着目している箇所の原因遷移以外の遷移によって、結果遷移の発火条件を維持させられないかを検査する。可能ならばこの分岐箇所については終了。(1) に戻る。
(4) 因果関係を追加する箇所の候補を探索。
(5) 追加箇所の候補のそれぞれについて、その因果関係を追加したものとして (1) に再帰する。

例えば、mra^+ から A^+ と clk^+ への分岐を見てみると、図3の STG のままでは A^+ が発火する前に $clk^+ \rightarrow mrr^- \rightarrow mra^-$ と進行して、A^+ の発火原因である mra=1 が消失してしまう危険があり、この分岐に関して persistency が成立していないことがわかる。これを避けるためには、$A^+ \rightarrow clk^+$、$A^+ \rightarrow mrr^-$、$R^- \rightarrow clk^+$、$R^- \rightarrow mrr^-$ のいずれかの因果関係を追加すればこの分岐に関して persistency が成立する。

このようにして STG 中の全ての分岐箇所を調べて、全体が persistency を満たすような因果関係の追加箇所を発見する。

この方法によって得られる因果関係を追加する箇所の選び方は幾通りにもなる。図3の STG の場合、R,A のループと mrr,mra,clk のループとの間に関しては、

- $A^+ \rightarrow clk^+$
- $A^+ \rightarrow mrr^-$
- $R^- \rightarrow clk^+$
- $R^- \rightarrow mrr^-$

の4通りのいずれかである。mrr,mra,clk のループと cwr,cwa のループとの間に関しては、

- $cwr^+ \rightarrow mrr^-$ と $mrr^- \rightarrow cwr^-$
- $cwr^+ \rightarrow mrr^-$ と $mra^- \rightarrow cwr^-$
- $cwr^+ \rightarrow mrr^-$ と $clk^- \rightarrow cwr^-$

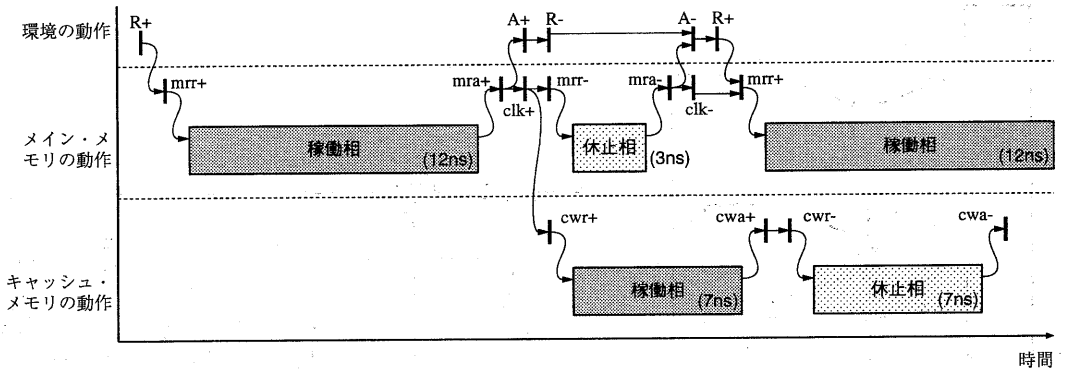

図4 ブロック転送回路の期待される動作タイミング

- $cwr^+ \rightarrow clk^-$ と $clk^- \rightarrow cwr^-$
- $cwa^+ \rightarrow mrr^-$ と $mrr^- \rightarrow cwr^-$
- $cwa^+ \rightarrow mrr^-$ と $mra^- \rightarrow cwr^-$
- $cwa^+ \rightarrow mrr^-$ と $clk^- \rightarrow cwr^-$
- $cwa^+ \rightarrow clk^-$ と $clk^- \rightarrow cwr^-$

の8通りのいずれかである。即ち、その組み合わせによって計32通りの選び方があることになる。

以上の中から最適な因果関係の追加箇所を選ぶためにブロック転送回路に期待される動作タイミングを考える。メイン・メモリの読み出しの稼働相は約12ns、メイン・メモリの読み出しの休止相は約3ns (オフセット・カウンタの休止相に律速されるため、比較的長い時間となっている)、キャッシュ・メモリの読み出しの稼働相は約7ns、キャッシュ・メモリの読み出しの休止相は約7nsである。以上からブロック転送回路は、図4に示すタイミングで動作することが期待される。

図4の動作タイミングより、

- clk^+が発火した時点では、それとほぼ同時にA^+が発火しており、$A^+ \rightarrow mrr^-$の因果関係の追加は回路の速度を低下させない。
- mra^-が発火した時点では既にcwr^+が発火しており、$cwr^+ \rightarrow clk^-$の因果関係の追加は回路の速度を低下させない。
- cwa^+が発火した時点では既にclk^-が発火しており、$clk^- \rightarrow cwr^-$の因果関係の追加は回路の速度を低下させない。

ということがわかり、図5のように因果関係を追加しても、これによる回路速度の低下はないと言える。

図5の"完全な仕様"からは、STGからQDIモデルに基づく回路を得る既存の各種の方法によって回路を生成することができる。例えば、Parkの方法[15]を用いると図6の回路が得られる。こうして生成される回路はQDIモデルに基づいているためタイミングフォールトに対する耐性が高く、等時分岐の仮定が満たされている限り、実際の回路の動作タイミングが図4の仮定からどれほど変動しても速度性能が低下することはあってもこの回路自身が誤動作を起こすことはない。

7. おわりに

非同期システムの制御回路の動作仕様は、そのままでは回路を生成できない"不完全な仕様"として与えられることが多い。この"不完全な仕様"に対して回路を生成するために必要な"冗長な仕様"を追加する段階で、データパス回路の処理遅延特性を考慮することにより速度性能とタイミングフォールトに対する耐性の高い制御回路を実現する設計手法を提案した。この手法を、非同期式プロセッサTITAC-2の命令キャッシュ内のブロック転送回路を例にとって説明した。

この設計手法はブロック転送回路の以外にもTITAC-2の命令キャッシュ中の制御回路で採用しており、この手法で設計した回路が設計者の期待通りに動作することは実際のチップで確認されている。

今回我々が提案した手法では、因果関係を追加する箇所の候補を見つける段階で全探索を行なっている。また追加箇所の候補の中から、速度性能の低下が最も少ないものを見つける段階ではヒューリスティックな手法によっている。これらの段階をより系統的に行なう方法については今後の課題である。

なお、本研究の一部は科研費補助金基盤研究(B)09480049、及び(株)半導体理工学研究センターとの共同研究によるものである。

参考文献

1) 南谷崇: 非同期式プロセッサ — 超高速VLSIシステムを目指して —, 情報処理, Vol. 34, No. 1, pp. 72–80 (1993).
2) 高村明裕, 桑子雅史, 南谷崇: 非同期式プロセッサTITAC-2の論理設計における高速化手法, 信学論(D–I), Vol. J80-D-I, No. 3, pp. 189–196 (1997).
3) Nanya, T., Takamura, A., Kuwako, M., Imai, M., Fujii, T., Ozawa, M., Fukasaku, I., Ueno, Y., Okamoto, F., Fujimoto, H., Fujita, O., Ya-

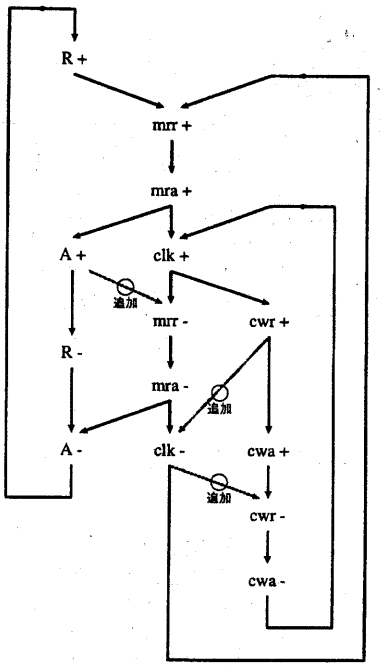

図 5　回路実現が可能なように因果関係を追加した STG

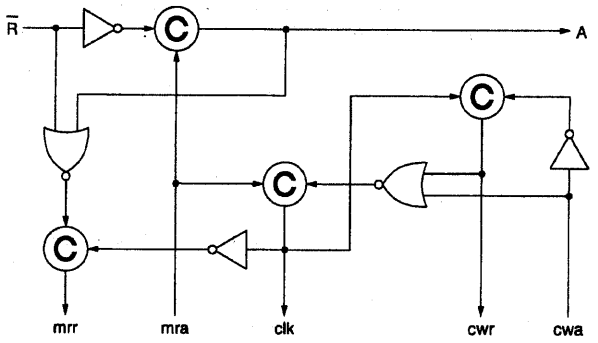

図 6　図 5から設計した回路

mashina, M. and Fukuma, M.: TITAC-2 : A 32-bit Scalable-Delay-Insensitive Microprocessor, *HOT CHIPS IX*, Stanford, pp. 19–32 (1997).

4) Takamura, A., Kuwako, M., Imai, M., Fujii, T., Ozawa, M., Fukasaku, I., Ueno, Y. and Nanya, T.: TITAC-2 : An asynchronous 32-bit microprocessor based on Scalable-Delay-Insensitive model, *Proc. International Conf. Computer Design (ICCD'97)* (1997).

5) Kuwako, M. and Nanya, T.: Timing-reliability evaluation of asynchronous circuits based on different delay models, *International Symposium on Advanced Research in Asynchronous Circuits and Systems*, Salt Lake City, IEEE Computer Society, pp. 22–31 (1994).

6) Udding, J. T.: A Formal Model for Defining and Classifying Delay-Insensitive Circuits, *Distributed Computing*, Vol. 1, No. 4, pp. 197–204 (1986).

7) 当麻喜弘, 内藤祥雄, 南谷崇: 順序機械, 岩波書店 (1983).

8) Martin, A.J.: Synthesis of Asynchronous VLSI Circuits, *FORMAL METHODS FOR VLSI DESIGN* (Staunstrup, J.(ed.)), Elsevier Science Publishers B. V., chapter 6, pp. 237–283 (1990).

9) Nanya, T., Ueno, Y., Kagotani, H., Kuwako, M. and Takamura, A.: TITAC: Design of a Quasi-Delay-Insensitive Microprocessor, *IEEE Design & Test of Computers*, Vol. 11, No. 2, pp. 50–63 (1994).

10) Meng, T. H.-Y., Brodersen, R.W. and Messerschmitt, D. G.: Automatic Synthesis of Asynchronous Circuits from High-Level Specifications, *IEEE Transactions on Computer-Aided Design*, Vol. 8, No. 11, pp. 1185–1205 (1989).

11) Moon, C. W.: *Synthesis and Verification of Asynchronous Circuits from Graphical Specifications*, PhD Thesis, Univ. of California at Berkeley (1992).

12) NEC: CB-C8 ファミリ セルベース IC メモリマクロ編 (1995).

13) Miller, R. E.: *Combinational Circuits*, Switching Theory, Vol. 1, John Wiley & Sons (1965).

14) Chu, T.-A.: Synthesis of Self-Timed VLSI Circuits from Graph-Theoretic Specifications, *Proc. International Conf. Computer Design (ICCD)*, IEEE Computer Society Press, pp. 220–223 (1987).

15) PARK, S.-B. and NANYA, T.: Synthesis of Asynchronous Circuits from Signal Transition Graph Specifications, *Trans. of IEICE on Information and Systems*, Vol. E80-D, No. 3, pp. 326–335 (1997).