社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2008－SLDM－137（18）
2008 / 11 / 18

配線遅延を考慮した回路モデル上でのハードウェアアルゴリズムの評価

長瀬 哲也† 高木 一義† 高木 直史†

† 名古屋大学大学院情報科学研究科 〒464–8603 愛知県名古屋市千種区不老町
E-mail: †{nagase,ktakagi,ntakagi}@takagi.i.is.nagoya-u.ac.jp

あらまし 集積回路設計において，計算時間や面積などの要求に応じてハードウェアアルゴリズムを設計，選択することが重要となる．従来のハードウェアアルゴリズムの評価では，配線遅延を無視し，回路の段数により計算時間を評価する回路モデルが使用されていた．しかし，集積回路の微細化に伴い，論理素子の遅延に対して配線遅延が相対的に増加しており，配線遅延を考慮した，より現実に即した回路モデルが必要であると考えられる．本稿では，配線長に依存する配線遅延を仮定した回路モデルを提案し，種々のハードウェアアルゴリズムについて，計算時間を評価する．評価により，段数の小さい回路ほど配線遅延の影響が大きくなるという結果が得られた．
キーワード 配線遅延，ハードウェアアルゴリズム，回路モデル，並列乗算器，並列加算器

Evaluation of Hardware Algorithms on a Circuit Model Considering Wire Delay

Tetsuya NAGASE†, Kazuyoshi TAKAGI†, and Naofumi TAKAGI†

† Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464–8603 Japan
E-mail: †{nagase,ktakagi,ntakagi}@takagi.i.is.nagoya-u.ac.jp

Abstract In the design of integrated circuits, it is important to design or choose algorithms according to the requirements such as the computation time and area. In the conservative logic circuit model, the computation time of hardware algorithms are evaluated by the circuit depth, and the wire delay has been ignored. However, with the recent miniaturization of the integrated circuits, the wire delay become significant and cannot be ignored relative to the delay of the logic elements. Therefore, the more realistic circuit model considering the wire delay is necessary. In this report, we propose a circuit model which assumes that the wire delay depends on its length. We evaluate computation time of several hardware algorithms. As a result, we could find that the effect of the wire delay grows in the circuit with small circuit depth.
Key words wire delay, hardware algorithm, circuit model, parallel multiplier, parallel adder

1. はじめに

集積回路の開発においては，専用回路や演算回路の設計にあたって，計算時間や面積などの要求に応じて回路の基礎となるハードウェアアルゴリズムを設計，選択する必要がある．ハードウェアアルゴリズムは，並列処理による計算の高速化や，集積回路化に適した規則正しい回路構造などを意識したハードウェア実現向きの計算手順である．ハードウェアアルゴリズムの設計，選択の際には，その性能を評価する必要がある．

ハードウェアアルゴリズムの性能評価は，回路を数学的にモデル化し，評価基準を定義した上で，その回路モデルに基づいて行う．従来，計算時間は，アルゴリズムに基づく回路の段数を評価基準として評価してきた．各論理素子における遅延を同一の定数値とし，配線遅延を無視することにより，回路の計算時間は回路の段数に比例する．しかし，実際の CMOS 論理回路において，遅延は素子の出力の負荷容量などに依存し，従って，ファンアウト及び配線の長さに依存する．回路の微細化の進展に伴い，論理素子の遅延に対して配線遅延が相対的に大きくなってきている．

このように，アルゴリズムに基づく回路の計算時間を回路の段数のみで見積もることが妥当ではなくなってきており，ファンアウトや配線による遅延を考慮する必要があると考えられる．本稿では，ハードウェアアルゴリズムのより現実に即した性能評価を行うために，配線長に依存する配線遅延を考慮した回路

モデルを提案する．その上で，種々のハードウェアアルゴリズムについて，アルゴリズムに基づく回路の配線遅延を見積もり，回路の計算時間を評価する．

本稿の構成は以下の通りである．2節では従来の回路モデル及び評価基準について述べる．3節では配線遅延を考慮した回路モデルとその評価基準について述べる．4節では並列乗算及び並列加算のハードウェアアルゴリズムを評価し，得られた結果について考察を行う．5節ではまとめを述べる．

2. 従来の回路モデルと評価基準

ハードウェアアルゴリズムを評価するための従来の回路モデルとして，組合せ回路モデルと，レイアウトまで考慮したVLSIモデルが挙げられる[1],[2]．これらのモデルの上で，係数や定数を無視したオーダーの議論により評価基準に基づいてハードウェアアルゴリズムの評価を行う．

2.1 組合せ回路モデル

組合せ回路モデルでは，回路はANDやORなどの与えられた種類の論理関数を実現する，入次数（ファンイン）に制限のある論理素子により構成される．論理素子の出次数（ファンアウト）には制限を設けない．回路はフィードバックループを持たない．回路の段数を計算時間の評価基準とし，回路の素子数をハードウェア量の評価基準とする．

2.2 VLSIモデル

VLSIモデルでは正方格子を考え，以下を仮定する[2]．

（1） 配線は，格子線に沿って，縦方向または横方向のみに引かれる．配線層はk層とする．すなわち，1つの格子点で交わる配線は最大k本である．2つの層の配線を接続するためにはコンタクトが必要である．

（2） 入出力パッド，論理素子，フリップフロップ，コンタクトなどの回路要素は1つの格子点上に位置する．1つの格子点を占める回路要素は1つである．回路要素が占める格子点に配線が行き当たっているとき，その配線はその回路要素に接続している．配線は回路要素が占めている格子点を通過できない．

（3） 配線は1方向の信号のみを伝える．

（4） 組合せ回路モデルと同様，論理素子のファンインは定数であり，ファンアウトの制限はない．

（5） 回路の領域は，回路を含む，各辺が格子線に沿って，最小の長方形である．入出力パッドは領域の外周線上にある．

(5) の長方形の領域の大きさを回路面積の評価基準とする．

VLSIモデルに従ってレイアウトした全加算器を図1に示す．図1では配線層を2層とし，水平方向の配線が一方の層を，垂直方向の配線が他方の層を使用するという制限を設けた．図中の■の素子はコンタクトを表す．

3. 配線遅延を考慮した回路モデル

従来，ハードウェアアルゴリズムの計算時間は，組合せ回路モデルでモデル化された回路の段数を評価基準として評価してきた．CMOS論理回路においては，遅延は素子の出力の負荷容量などに依存する．論理素子の出力の負荷容量は，ファンアウト及び配線の長さに依存する．さらに，集積回路の微細化により，長距離配線で使用されるようなグローバル配線ではスケーリングに対して遅延時間が増大し，演算器内部で使用されるようなローカル配線ではスケーリングに対して一定となる．このようなことから，論理素子の遅延に対して配線遅延が支配的になりつつある[3]．

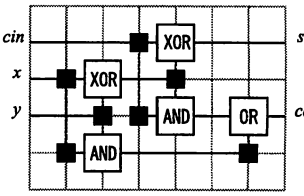


図1 VLSIモデル上の全加算器のレイアウト
Fig. 1 Layout of FA on the VLSI model.

より現実に即したアルゴリズムの計算時間の評価を行うために，論理素子のファンアウトや配線による遅延を考慮する必要がある．本稿では特に配線による遅延に着目し，配線長に依存する配線遅延モデルを提案し，その上で回路の配線遅延を見積もることでアルゴリズムの計算時間を評価する．

3.1 回路モデル

配線長に依存する配線遅延を考慮した回路モデルを提案する．VLSIモデルに従ってレイアウトを行い，その上で論理素子間の配線の遅延がその長さに依存すると仮定する．回路の入力から出力までの経路のうち，論理素子間の合計配線遅延の最大値を回路の配線遅延とする．論理素子の遅延は従来の回路モデルと同様，同一の定数であると仮定する．これらの仮定により，回路の計算時間は配線遅延に比例する．本稿では，配線遅延をハードウェアアルゴリズムの計算時間の評価基準とする．

回路全体の配線遅延D_wは次のように計算する．長さlの配線に対して，配線遅延が$d_w(l)$で与えられるとする．回路の入力から出力に至る経路の集合をPとする．経路$p \in P$における回路の段数がK_pであるとする．回路のk段目（$k = 1, 2, \cdots$）の論理素子の出力配線をk段目の配線と呼ぶこととする．このとき，k段目の論理素子における長さl_{p_k}の出力配線に対する配線遅延を求め，それらの和をとる．このように求めた配線遅延のうち，最大のものを回路の配線遅延とする（式(1)）．

$$D_w = \max_{p \in P} \left(\sum_{k=1}^{K_p} d_w(l_{p_k}) \right) \tag{1}$$

3.2 配線遅延モデル

配線遅延を評価するためには，配線をモデル化した上でその遅延を見積もる必要がある．従来の回路モデルにおいては配線遅延は配線長lに対して一定としていたが，本稿では，配線長lに依存した配線遅延を仮定する．配線の物理モデルとして，配線を容量Cのみでモデル化する方法や，抵抗Rと容量Cでモデル化する方法が知られている．これらの物理モデルについて次に述べる．これらの物理モデルから，長さlの配線に対する配線遅延のモデルを仮定する．

3.2.1 配線容量によるモデル化

配線遅延について，配線容量による影響が支配的で，配線抵

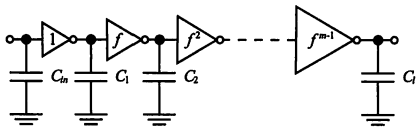


図 2　m 段のバッファの挿入
Fig. 2　m stages buffer insertion.

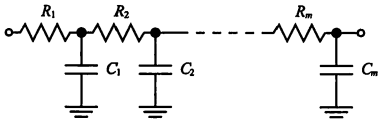


図 3　RC 配線モデル
Fig. 3　RC wire model.

抗による影響が無視できるようなモデルを考える．配線容量 C_l が配線長 l に比例するため，このモデルでは，配線遅延は配線長に比例する．

配線長に比例する配線遅延を，バッファを挿入することによって改善することができる [4]．図 2 のように，バッファを m 個挿入し，各段で容量が f 倍になるようにバッファサイズを定める．i 段目のバッファの遅延は $C_{i+1}/C_i = f \cdot t_0$ となる．t_0 は最小サイズのインバータの遅延時間である．従って，入力容量 C_{in}，配線容量 C_l に対して，$C_l/C_{in} = f^m \cdot t_0$ となる．配線遅延は $d_w(l) = m \cdot f \cdot t_0$ となる．最適な f は自然対数の底 e であり，遅延時間は $d_w(l) = e \cdot \log(C_l/C_{in}) \cdot t_0 = O(\log l)$ となる．

3.2.2　配線抵抗及び配線容量によるモデル化

配線が配線抵抗 R と配線容量 C でモデル化されるとし，その上で，配線遅延の評価を Elmore 遅延モデル [5] により行う．この RC モデルにおいて配線が図 3 にモデル化される場合，Elmore 遅延モデルに基づく配線遅延 d_w は $d_w = R_1 \cdot C_1 + (R_1 + R_2) \cdot C_2 + \cdots + (R_1 + R_2 + \cdots + R_m) \cdot C_m$ となる．$R_i = R$, $C_i = C$ と仮定すると，遅延は $d_w = R \cdot C \cdot m \cdot (m+1)/2$ となり，m^2 に比例する．単位長さ当たりの抵抗を r，単位長さ当たりの容量を c とすると，長さ l の配線では遅延 $d_w(l)$ が rcl^2 に比例する．r, c は配線長に依存しないため，配線遅延は配線長の 2 乗に比例する．

配線長の 2 乗に比例する配線遅延に対し，配線を分割してバッファをリピータとして挿入することにより配線遅延を改善することができる．長さ l の配線に対して $p = l/l_0$ 個のバッファを挿入すると (l_0 = 定数)，p 本の長さ l_0 の配線に分割される．長さ l_0 の配線の配線遅延は定数であるため，全体の配線遅延は $O(p) = O(l)$ となる．

以上で述べた配線の物理モデルでは，配線遅延が配線の長さ l に対して l や l^2 に比例し，バッファを挿入することにより $\log l$ や l となる．ハードウェアアルゴリズムの評価では，回路に適切なバッファの挿入がなされるものと考える．従って，配線遅延の評価においては，バッファ挿入後の配線遅延である $\log l$ や l といった遅延モデルを使用する．以上のことから，配線遅延が配線長 l に対して $\log l \sim l$ 程度になると考え，$\log l$, $\sqrt{l}$, l という 3 つの配線遅延モデルを評価に使用する．

4.　種々のハードウェアアルゴリズムの評価

上述の配線遅延を考慮した回路モデルに従ってレイアウトを仮定し，ハードウェアアルゴリズムに基づく回路の性能を評価する．種々の並列乗算及び並列加算のハードウェアアルゴリズムを対象とする．

2.2 で述べた，入出力パッドが回路の外周上にあるという仮定では，組み合わせ回路で並列乗算器及び並列加算器を構成する場合に，本質的に長さ $\Omega(n)$ の配線が存在する [6], [7]．従って，長さ l の配線に対する配線遅延が $\log n$, $\sqrt{n}$, n のとき，計算時間の下界は $\Omega(\log n)$, $\Omega(\sqrt{n})$, $\Omega(n)$ となる．

4.1　並列乗算のハードウェアアルゴリズムの評価

n ビット × n ビット並列乗算のハードウェアアルゴリズムを評価する．並列乗算器は乗数と被乗数の各ビットの AND を並列に計算する部分積生成部 (Partial Product Generator: PPG)，生成した部分積を桁上げ保存形で順次加え合わせる部分積累算部，桁上げ伝播加算器 (Carry Propagation Adder: CPA) から成る最終加算部により構成される．本稿では，まず部分積累算部について評価を行い，その後，部分積生成部も考える．

並列乗算のハードウェアアルゴリズムは部分積の累算の方法により種々の構成方法がある．本稿では，配列型乗算 [8]，バランス木を用いた乗算 [9]，Wallace 木を用いた乗算 [10]，4–2 加算木を用いた乗算 [11] のアルゴリズムを評価した．素子数，段数，面積を表 1 に示す．

4.1.1　配列型乗算

配列型乗算は，部分積を並列に生成し，桁上げ保存加算により順次累算する．配列型乗算器は，図 4 のような桁上げ保存加算器 (Carry Save Adder: CSA) を並べた構成になる．VLSI モデル上での配列型乗算器のレイアウトを図 5 のように仮定する．図中の●は n ビットの部分積，□は図 4 のような桁上げ保存加算器である．回路の段数は $O(n)$ である．

4.1.2　バランス木を用いた乗算

バランス木を用いた乗算は，部分積をいくつかのグループに分け，図 6(a) のようなバランス木を構成することにより，部分積の累算を並列に行う．バランス木を用いた乗算器のレイアウトを図 6(b) のように仮定する．回路の段数は $O(\sqrt{n})$ となる．図中の太線の経路は最も配線遅延が大きな経路である．

4.1.3　Wallace 木を用いた乗算

Wallace 木を用いた乗算は桁上げ保存加算を基本計算として，部分積を図 7(a) のように木状に累算する．Wallace 木を用いた乗算器のレイアウトを図 7(b) のように仮定する．回路の段数は $O(\log n)$ となる．

4.1.4　4–2 加算木を用いた乗算

4–2 加算木を用いた乗算は，部分積を図 8(a) のように 2 分木状に累算する．基本計算は，図 8 の点線で囲まれた 4–2 加算器であり，桁上げ保存加算器 2 段で構成される．レイアウトを図 8(b) のように仮定する．回路の段数は $O(\log n)$ となる．

これらの並列乗算アルゴリズムを，配線遅延を考慮した回路

表 1　従来の回路モデルによる並列乗算器の評価

Table 1　Evaluation of multipliers by a traditional circuit model.

	素子数	面積	段数
配列型	$O(n^2)$	$O(n^2)$	$O(n)$
バランス木	$O(n^2)$	$O(n^2)$	$O(\sqrt{n})$
Wallace 木	$O(n^2)$	$O(n^2 \log n)$	$O(\log n)$
4–2 加算木	$O(n^2)$	$O(n^2 \log n)$	$O(\log n)$

表 2　並列乗算器の計算時間の評価

Table 2　Evaluation of the computation time on multipliers.

	k 段目の配線長	配線遅延		
		$\log l$	$\sqrt{l}$	l
配列型	$O(1)$	$O(n)$	$O(n)$	$O(n)$
バランス木	$O(k)$ $k=even$ $O(1)$ $k=odd$	$O(\sqrt{n}\log n)$	$O(n^{3/4})$	$O(n)$
Wallace 木	$O((3/2)^k)$	$O((\log n)^2)$	$O(\sqrt{n})$	$O(n)$
4–2 加算木	$O(2^k)$	$O((\log n)^2)$	$O(\sqrt{n})$	$O(n)$
Wallace 木 +PPG	$O((3/2)^k)$	$O((\log n)^2)$	$O(\sqrt{n})$	$O(n)$
4–2 加算木 +PPG	$O(2^k)$	$O((\log n)^2)$	$O(\sqrt{n \log n})$	$O(n \log n)$

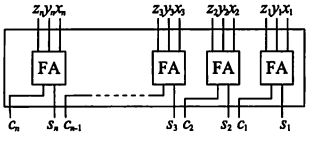


図 4　桁上げ保存加算器

Fig. 4　Carry save adder.

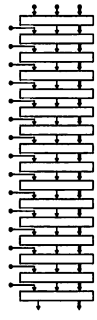
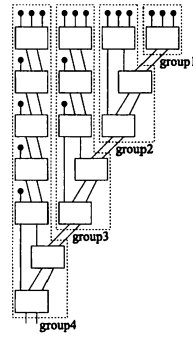
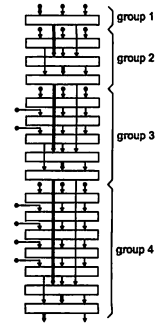


図 5　配列型乗算器のレイアウト

Fig. 5　Layout of array multiplier.



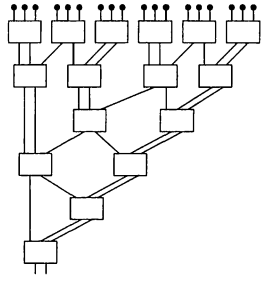
(a) バランス木



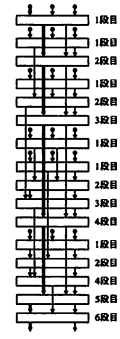
(b) レイアウト

図 6　バランス木を用いた乗算器

Fig. 6　Multiplier with a balanced tree.



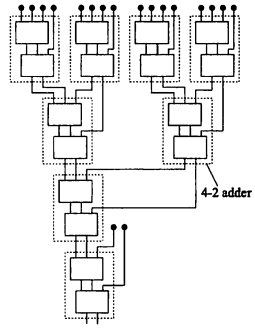
(a) Wallace 木



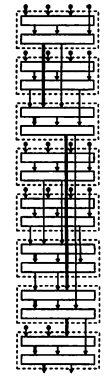
(b) レイアウト

図 7　Wallace 木を用いた乗算器

Fig. 7　Multiplier with Wallace tree.



(a) 4–2 加算木



(b) レイアウト

図 8　4–2 加算木を用いた乗算器

Fig. 8　Multiplier with 4–2 adder tree.

モデル上で評価した結果を表 2 に示す．部分積の累算を並列に行う乗算器では，レイアウト上で桁上げ保存加算器を飛び越す配線が増加し，素子間の配線が長くなる傾向がある．従って，回路の段数が小さい乗算器ほど，計算時間の増加率が大きくなっている．また，配線長に比例する配線遅延を仮定した場合では，どの乗算器においても計算時間が $O(n)$ となる．

4.1.5　部分積生成部を含めた評価

図 9 のように，乗算器の入力となる被乗数は回路の上端から，乗数は右端から入力されると仮定する．このとき，被乗数 x_i，乗数 y_i の分配ための配線において，最長配線が回路の幅に比例する長さとなる．Wallace 木，4–2 加算木を用いた乗算器では，一つの桁上げ保存加算器を飛び越す配線がビットスライス当たり最大 $O(\log n)$ 本であることから，回路の幅が $O(n \log n)$ となり，部分積生成部の配線の長さが $O(n \log n)$ となる．Wallace 木，4–2 加算木を用いた乗算器の部分積生成部を含めた回路の配線遅延を評価すると，配線遅延モデルが $\sqrt{l}$，l のときに部分積生成部の遅延時間が支配的となる（表 2）．

4.2　並列加算のハードウェアアルゴリズムの評価

n ビット並列加算のハードウェアアルゴリズムを評価する．本稿では，順次桁上げ加算，桁上げ飛び越し加算，並列プレフィクス加算のアルゴリズムを評価した．素子数，段数，面積を表 3 に示す．

4.2.1　順次桁上げ加算

順次桁上げ加算では，下位ビットから順に各桁の和と桁上げを計算する．順次桁上げ加算器は，図 10 のように全加算器を直列に接続する構成になる．回路のレイアウトは図 10 の通りに全加算器を一直線に並べる．段数は $O(n)$ である．

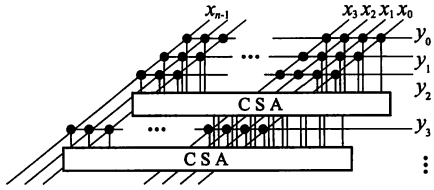


図 9 部分積生成部

Fig. 9 Partial product generator.

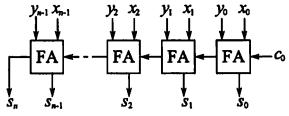
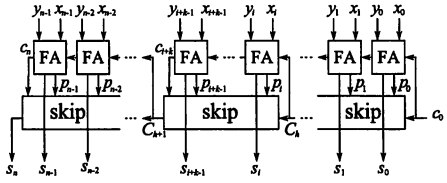
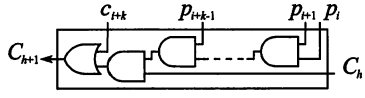


図 10 順次桁上げ加算器

Fig. 10 Layout of ripple carry adder.



(a) 桁上げ飛び越し加算器



(b) 桁上げ飛び越し回路

図 11 桁上げ飛び越し加算器

Fig. 11 Carry skip adder.

4.2.2 桁上げ飛び越し加算

桁上げ飛び越し加算は，計算をいくつかのブロックに分け，各ブロックでの桁上げ伝播条件を求め，条件が成立する場合は下位からの桁上げがそのブロックを飛び越して上位のブロックに伝播する．桁上げ飛び越し加算器を図 11(a) に示す．各ブロック内では順次桁上げ加算を行い，図 11(b) の桁上げ飛び越し回路で桁上げ伝播条件を求める．ブロックサイズは $O(\sqrt{n})$ とする．桁上げ飛び越し加算器では，各ブロック内の $O(\sqrt{n})$ 段の順次桁上げ加算，$O(\sqrt{n})$ 個の桁上げ飛び越し回路の計算を行う．回路のレイアウトとして，図 11 の通りに素子を配置したものを仮定する．段数は $O(\sqrt{n})$ となり，桁上げ飛び越し回路においてブロックサイズに比例する長さの配線が存在する．

4.2.3 並列プレフィクス加算

n ビット加算における各桁の桁上げ生成条件と桁上げ伝播条件をそれぞれ $g_i = x_i \cdot y_i$，$p_i = x_i \oplus y_i$ とすると，各桁の和と桁上げはそれぞれ $c_{i+1} = g_i \vee p_i c_i$，$s_i = p_i \oplus c_i$ となる．連続する 2 桁から成るブロックの桁上げ生成条件と桁上げ伝播条件について，$g_{i+1,i} = g_{i+1} \vee p_{i+1} g_i$，$p_{i+1,i} = p_{i+1} \oplus p_i$ が成り立つ．この計算を以下のように定義する．

$$(g_{i+1,i}, p_{i+1,i}) = (g_{i+1}, p_{i+1}) * (g_i, p_i) \qquad (2)$$

この計算により，$g_{i,0}$，$p_{i,0}$ を求め，$c_{i+1} = g_{i,0} \vee p_{i,0} c_0$ により各桁の桁上げを計算する．計算 $*$ はプレフィクス計算と呼ばれる．プレフィクス計算 $*$ は結合律が成り立つため，繰り返し計算を並列化できる．これに基づいて構成したのが並列プレフィクス加算器である．プレフィクス計算の結合順序により，並列プレフィクス加算器には種々の実現方法がある．本稿では，Ladner-Fischer [12] 加算器（図 12），Kogge-Stone [13] 加算器（図 13），Brent-Kung [14] 加算器（図 14）を評価した．図の回路では，プレフィクス加算器におけるプレフィクス計算の部分を抜き出しており，■はプレフィクス計算 $*$ に相当する．上端から (g_i, p_i) が入力され，下端から (g_i, p_0) が出力される．並列プレフィクス加算器のレイアウトとしては，図の通りに素子を配置したものを仮定する．回路の段数は全て $O(\log n)$ であるが，表 3 に示すように，構成によって素子数，面積が異なる．

表 3 従来の回路モデルによる並列加算器の評価

Table 3 Evaluation of adders by a traditional circuit model.

	素子数	面積	段数
順次桁上げ	$O(n)$	$O(n)$	$O(n)$
桁上げ選択	$O(n)$	$O(n)$	$O(\sqrt{n})$
Ladner-Fischer	$O(n \log n)$	$O(n \log n)$	$O(\log n)$
Kogge-Stone	$O(n \log n)$	$O(n^2)$	$O(\log n)$
Brent-Kung	$O(n)$	$O(n \log n)$	$O(\log n)$

表 4 並列加算器の計算時間の評価

Table 4 Evaluation of the computation time on adders.

	k 段目の配線長	配線遅延		
		$\log l$	$\sqrt{l}$	l
順次桁上げ	$O(1)$	$O(n)$	$O(n)$	$O(n)$
桁上げ選択	$O(k)$	$O(\sqrt{n} \log n)$	$O(n^{3/4})$	$O(n)$
Ladner-Fischer	$O(2^k)$	$O((\log n)^2)$	$O(\sqrt{n})$	$O(n)$
Kogge-Stone	$O(2^k)$	$O((\log n)^2)$	$O(\sqrt{n})$	$O(n)$
Brent-Kung	$O(2^k)$	$O((\log n)^2)$	$O(\sqrt{n})$	$O(n)$

これらの並列加算アルゴリズムを，配線遅延を考慮した回路モデル上で評価した結果を表 4 に示す．桁上げの計算を並列に行う加算器では，素子間の配線が長くなる傾向がある．乗算器と同様に，回路の段数が小さい乗算器ほど，計算時間の増加率が大きくなっている．また，配線長に比例する配線遅延を仮定した場合では，どの加算器においても計算時間が $O(n)$ となる．

論理素子の負荷容量を増加させる要因は配線長だけでなく，ファンアウトにもある．評価した並列プレフィクス加算器のうち，Ladner-Fischer 加算器では k 段目の素子のファンアウトの最大値は 2^k に比例する．従って，ファンアウトの影響による遅延が無視できないと考えられる．素子のファンアウトを定数に制限することを考えると，図 15 のようにファンアウトを 2 分木状に分岐することにより，f 個のファンアウトを $O(\log f)$ 段で構成できる．ファンアウト木のレイアウトを図 15 のように行うと仮定して，計算時間を評価した結果を表 5 に示す．回路の段数が $(\log l)^2$ に増加し，配線遅延モデルが $\log l$ のときに計算時間が $(\log l)^3$ に増加するという結果を得た．

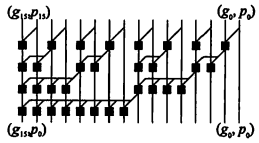


図 12 Ladner-Fischer 加算器
Fig. 12 Ladner-Fischer adder.

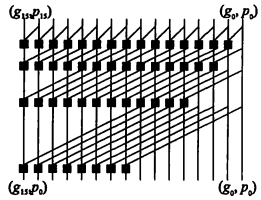


図 13 Kogge-Stone 加算器
Fig. 13 Kogge-Stone adder.

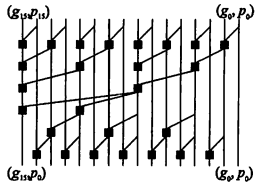


図 14 Brent-Kung 加算器
Fig. 14 Brent-Kung adder.

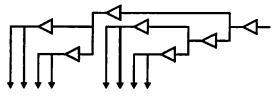


図 15 ファンアウト木
Fig. 15 Fanout tree.

表 5 ファンアウトを考慮した Ladner-Fischer 加算器の計算時間の評価
Table 5 Evaluation of the computation time on the Ladner-Fischer adder considering the fanout.

	段数	配線遅延		
		$\log l$	$\sqrt{l}$	l
Ladner-Fischer	$O((\log n)^2)$	$O((\log n)^3)$	$O(\sqrt{n})$	$O(n)$

4.3 考　　察

評価により，部分積の累算を並列に行う乗算器や，桁上げ伝播を並列に行う加算器においては，段数が小さい回路ほど，素子間の配線が長くなる傾向があり，計算時間の増加率が大きくなるという結果を得た．配線長に比例する配線遅延を仮定した場合には，乗算器，加算器ともに，計算時間が n に比例するという結果を得た．さらに，複雑な配線により回路の面積が増加する Wallace 木や 4–2 加算木を用いた乗算器では，部分積生成部に長い配線が生じ，配線遅延 $\sqrt{l}$，l モデルではその配線の遅延時間が支配的となるという結果を得た．この場合，Wallace 木や 4–2 加算木を用いた乗算器よりも，配列型乗算器の方が優れているということができる．

以上より，並列度が増し，回路の段数が小さくなるほど，配線遅延による影響が大きくなるということができる．従って，集積回路の微細化が進み，ゲート遅延に対して配線遅延がより支配的になれば，回路の段数に基づく評価では，正しい計算時間を見積もることができず，最適なアルゴリズムを選択できない可能性がある．配線遅延を考慮した回路モデルは，ハードウェアアルゴリズムのより現実に即した評価を行うための手がかりとなると考えられる．

並列プレフィクス加算器では，段数や配線遅延を考慮した評価では計算時間に違いがなかったアルゴリズムに対して，ファンアウトを考慮することによって，違いが表れることがあるという結果を得た．従って，ファンアウトを考慮することも，ハードウェアアルゴリズムのより現実に即した評価を行うための手がかりとなると考えられる．

5. ま と め

集積回路の微細化により，配線遅延が論理素子の遅延に対して相対的に大きくなってきており，回路の計算時間の評価において配線遅延を考慮する重要性から，本稿では配線長に依存する配線遅延を考慮した回路モデルを提案した．この回路モデルに基づいて，種々のハードウェアアルゴリズムを評価したところ，段数が小さい回路ほど，素子間の配線が長くなる傾向があり，計算時間に対する配線遅延の影響が大きくなるという結果を得た．従って，配線遅延を考慮した回路モデルは，ハードウェアアルゴリズムのより現実に即した評価を行うための手がかりとなると考えられる．

文　　献

[1] 高木 直史：“算術演算の VLSI アルゴリズム”，コロナ社 (2005).
[2] J. D. Ullman: “Computational Aspects of VLSI”, Computer Science Press (1983).
[3] 吉川 公麿：“多層配線技術とスケーリング”，電子情報通信学会論文誌 C, **83**, 2, pp. 105–117 (2000).
[4] C. Mead and L. Conway: “Introduction to VLSI Systems”, Addison-Wesley (1980).
[5] W. C. Elmore: “The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers”, Journal of Applied Physics, **19**, p. 55 (1947).
[6] J. Vuillemin: “Combinatorial Limit to the Computing Power of VLSI Circuits”, IEEE Trans. Comput., **C-32**, 3, pp. 294–300 (1983).
[7] B. Chazelle and L. Monier: “A model of computation for vlsi with related complexity results”, Journal of the ACM, **32**, 3, pp. 573–588 (1985).
[8] C. R. Baugh and B. A. Wooley: “A Two's Complement Parallel Array Multiplication Algorithm”, IEEE Trans. Comput., **C-22**, 12, pp. 1045–1047 (1973).
[9] S. D. Pezaris: “A 40-ns 17-Bit by 17-Bit Array Multiplier”, IEEE Trans. Comput., **C-20**, 4, pp. 442–447 (1971).
[10] C. S. Wallace: “A Suggestion for a Fast Multiplier”, IEEE Trans. Electronic Computers, **EC-13**, 1, pp. 14–17 (1964).
[11] J. Vuillemin: “A very fast multiplication algorithm for VLSI implementation”, Integration, the VLSI Journal, **1**, pp. 39–52 (1983).
[12] R. E. Ladner and M. J. Fischer: “Parallel Prefix Computation”, Journal of the ACM, **27**, 4, pp. 831–838 (1980).
[13] P. M. Kogge and H. S. Stone: “A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations”, IEEE Trans. Comput., **C-22**, 8, pp. 786–793 (1973).
[14] R. P. Brent and H. T. Kung: “A Regular Layout for Parallel Adders”, IEEE Trans. Comput., **C-31**, 3, pp. 260–264 (1982).