計測自動制御学会東北支部 第194回研究集会 (2001.6.1)
資料番号 194-1

高速冗長2進加減算器の除算器への応用

Application to Divider for High-Speed Redundant Binary Adder-Subtractor

○熊谷友祐*, 五日市喜洋*, 日野杉充希*, 恒川佳隆*

○Yusuke Kumagai*, Yoshihiro Itsukaichi*, Mitsuki Hinosugi*, Yoshitaka Tsunekawa*

*岩手大学

*Iwate University

キーワード： 冗長2進表現 (redundant binary representation), 高速 (high-speed), 加減算器 (adder-subtractor), VLSI評価 (VLSI evaluation), 除算器 (Divider)

連絡先： 〒020-8551 盛岡市上田4-3-5 岩手大学 工学部
熊谷友祐, Tel.: (019)621-6468, Fax.: (019)621-6468, E-mail: t2300006@iwate-u.ac.jp

1. はじめに

算術演算の中で加算・減算は最も基本となるものであり，乗算や除算をはじめあらゆる算術演算がこれらの繰り返しで行われる．そのため，加算器・減算器の処理能力は，実現するディジタルシステムの性能に大きく関係する．

現在，マルチメディアのようなリアルタイム処理を必要とする分野において，より高速性が要求されている．さらに，近年では高速化ばかりでなく，情報セキュリティなどの種々の分野において，高精度演算の重要性も高まっている．高速性および高精度性が同時に要求される問題に対応する数体系の1つとしてSD(Signed Digit)表現がある[1]．本稿では，このSD表現の一種である冗長2進数を用いた演算について検討を行う．

われわれはこれまでに，冗長2進数に基づく一検討として，従来の加減算器よりも極めて高速な加減算器を提案してきた[2]．これは，符号変換器を必要とせず，しかもわれわれがすでに提案してきた1桁2ビット/3ビット混合表現を用いた高速冗長2進加算器および減算器を組み合わせることによって，従来の加減算器と比較して約1.7倍の高速化を可能とした[3]．

本稿では，本加減算器の応用として，除算器にこれを適用した場合についての検討を行う．これまで，除算器の構成法としてはいくつか提案されてきているが，本報告ではまず，冗長2進表現を用いた従来の除算用ハードウェアアルゴリズムに基づく高速化手法について考察する[6]．次に，このアルゴリズムは被除数，除数共に正の場合に限定した手法であるため，新たに被除数，除数共に符号を考慮した場合についての除算アルゴリズムを提案し，このアルゴリズムに基づく構成法について考察する．最後に，本除算器の構成法に対してVLSI設計システムPARTHENONを用いてVLSI設計および評価を行う．その結果，従来から提案されている高速冗長2進除算器に対し，約2倍の高速化が可能となることを明らかにする．

2. 符号変換器を必要としない高速冗長2進加減算器

本章では，最初に従来の符号変換器を用いた加減算器の構成法について述べ，これまでの問題点を明らかにする[6]．次に，われわれがすでに提案してきた符号変換器を必要としない高速冗長2進

加減算器の構成法についてその概要を述べる[2].

2.1 従来の減算法とその手法を用いた加減算器の構成

現在，ディジタルシステムにおける数値表現は，2の補数形式が広く用いられている．この数体系では減算の操作を加算に置き換えて演算できるため，減算器を用いる必要がないことが大きな特長となっている．このことから，減算器は加算器を用いて行うことが一般的となっていた．

また，本研究で扱う数体系の冗長2進数においても2の補数表現の場合と同様に，減算の方法は減数の符号を変換してそれを加算するという手法が一般的であった．したがって，減数入力の前に符号変換器を取り付ける構成となるため，その分加算器よりも遅延時間が大きくなる．なお，この従来の加減算器の遅延時間は，われわれの検討結果より加算器の遅延時間と比較して約1.3倍も大きくなることが分かっている[2].

また，従来の冗長2進数の表現方法は，1桁を2ビットで表現することが一般的であった．これは，1桁の要素である$\{\bar{1},0,1\}$の3つを表現するのに十分だからである．ここで$\bar{1}$は-1を表す．しかし，われわれのこれまでの検討結果から，1桁を2ビットで表現することはどの方法を用いたとしてもある1つの要素を検出するのに必ず2ビット参照しなければならない．したがって，従来の表現法ではハードウェア量および処理時間が増加する[3].

2.2 本加減算器の2つの高速化手法

従来の加減算器の問題に対し，われわれはこれまでに高速な冗長2進加減算器を提案してきた[2]．本節では，本加減算器を高速化する2つの手法について述べる．

＜新たな冗長2進数に基づく減算法＞

われわれは，本加減算器の高速化の最初のアプローチとして，まずこれまで検討が十分行われてこなかった減算器について考察し，その減算方法を提案した．

その方法は，減算方法を加算と同様に2つのステップで考える．最初のステップ1では，各桁で中間桁借りと中間差が同時に$\bar{1}$あるいは1にならないように，1つ下位の桁の入力も調べて中間桁借りと中間差を決める．ステップ2では，各桁で下位桁から出力された中間桁借りと中間差を加算して差を求める．ステップ2に関しては加算器と同じ処理で行うことができる．このことを利用し，次節で述べる高速加減算器を効率よく構成することができる．

＜1桁2ビット/3ビット混合表現の適用＞

前節でも述べたように，従来では1桁を2ビットで表現することが一般的であったため，その表現法を用いた冗長2進加減算器はハードウェア量および処理時間が増加してしまう．その問題を解決するために，われわれは入出力を3ビット，そして，中間桁上げおよび中間和に対しては2ビットを参照する表現法を提案してきた[3]．この表現法を用いることによって，従来の2ビット表現に対しハードウェア量を抑えつつも高速な加算器が実現できる．

そこで，本稿で述べる減算器においてもこの表現法を適用する．この表現法を適用すると，本減算器とすでに提案してきた高速加算器がよく似た構成となる．この特徴は次節で述べる高速加減算器を構成する上で極めて有効となる．

2.3 高速加減算器の構成法

本節では，最後の高速化のアプローチとして，すでに提案してきた本加算器と全く同等の処理時間を持つ加減算器の構成法について述べる[2]．その構成法とは，高速化を図るためにわれわれがすでに提案した高速加算器と前節の高速減算器を並列に組み合わせる方法である．しかし，単に組み合わせただけでは，高速加算器よりも処理時間が大きくなり，また，ハードウェア量も大きくなってしまう．そこで，次のように効率化を図る．

最初に，ハードウェア量の増加の問題に関しては，Hセルの共有化を行うことによりハードウェア量の効率化を図る．そして，加算器および減算器を組み合わせる際に，前節で述べた加算器と減算器の構成が似ているという特徴を用いる．すなわち，共有できるゲートを利用することである．

次に，処理時間の減少を図る上で，加算器および減算器の出力を制御信号によって選択するセレクタを用いた構成をベースに考える．

そこで，まず本加減算器の高速化手法の1つとして，容易に演算が決定できるよう本加減算器ではセレクタの制御信号CSを2ビット設ける．次に，制御信号を付加したことによる処理時間の増加の問題を解決するために，中間桁上げC_iに対しては制御信号による多段化を行わないよう式変形を行う．これは，加算時および減算時における中間桁上げの出力が制御信号を待たずに並列に実行させるためである．そして，中間和S_iに対しては，制御信号を2ビットで表現した特長を利用し，制御信号と下位桁の条件を組み合わせた項を新たな条

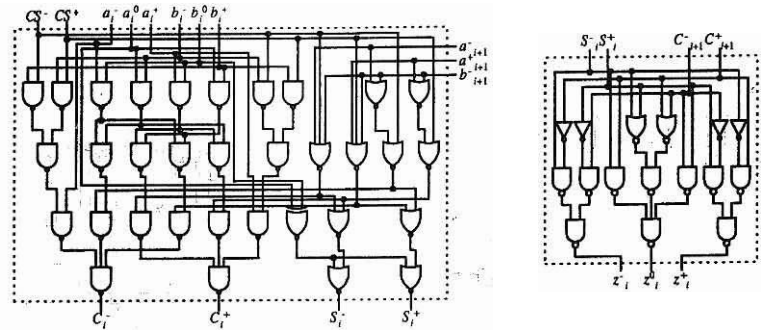


Fig. 1 Structure of high-speed redundant binary adder-subtractor.

件式とする．そして，この条件式を用いることによって式変形を行う[2]．

以上の検討結果より，本加減算器の処理時間はすでに提案してきた高速加算器と全く同等の処理時間で演算を行うことができる．したがって，従来の加減算器よりも極めて高速となる．なお，検討結果から得られた本加減算器の構成を図1に示す．

3. 従来の除算アルゴリズムを用いた冗長2進除算器の構成

これまでに冗長2進表現を用いた除算器の構成が提案されている[6]．これは，冗長2進表現の特長をいかした除算用アルゴリズムを用いて高速化を図っている．本章では，最初に従来から提案されている除算アルゴリズムについて述べる．次に，そのアルゴリズムに基づいて構成した除算器について述べる．これまでに提案された従来の除算器では，被除数が冗長2進表現，除数が純真2進表現を用いて構成されていた[6]．しかし，本稿における従来型の除算器では，被除数，除数共に1桁2ビットの冗長2進表現に拡張した構成法について考える．

3.1 従来の除算アルゴリズム

本節では，これまでに提案されている冗長2進表現の特長を最大限にいかした高速除算用アルゴリズムについて簡単に述べる[6]．これを示す前に，まず変数を以下のように定義する．

X： 入力の演算数（被除数, $\frac{1}{2} \leqq X < 1$）
Y： 入力の演算数（除数, $\frac{1}{2} \leqq Y < 1$）
Q： 出力の商（$\frac{1}{2} \leqq Q < 1$）
q_i： 商の小数点i桁目
$R^{(i)}$： i回目に得られる部分剰余
r_j： $R^{(i-1)}$のj桁目
N： 桁数

高速除算用アルゴリズムは以下の通りである．

begin
　＜ステップ1＞
　$R^{(0)} \leftarrow X$ （冗長2進数に変換）
　$D \leftarrow Y$
　＜ステップ2＞
　$q_0 := 1$
　$R^{(1)} := R^{(0)} - D$ （冗長2進数体系で計算）
　＜ステップ3＞
　for $i := 1$ to N do
　　begin

$$q_i := \begin{cases} \bar{1} & \text{if } [r_0.r_1r_2]_{SD2} < 0 \\ 0 & \text{if } [r_0.r_1r_2]_{SD2} = 0 \\ 1 & \text{if } [r_0.r_1r_2]_{SD2} > 0 \end{cases}$$

　　$R^{(i+1)} := 2R^{(i)} - q_iD$
　　（冗長2進数体系で計算）
　　end
　＜ステップ4＞
　$Q := [q_0.q_1q_2 \cdots q_N]$
　end

このアルゴリズムを見てわかるように，部分剰

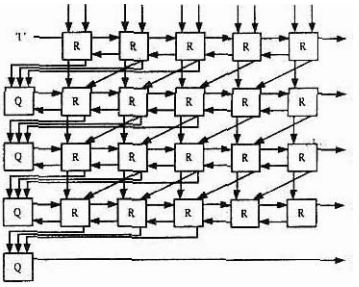


Fig. 2 Structure of divider using redundant binary representation.

余の算出の繰り返しによって商の各桁を逐次的に決定していく．また，このアルゴリズムでは部分剰余の上位3桁を参照するだけで商の各桁を容易に決定できることから，このアルゴリズムを用いた除算器は高速演算が期待できる．ただし，次行の部分剰余の加算/減算を決定するためには，部分剰余の上位3桁から次行の制御信号である商桁の符号を検出する必要がある．

3.2 従来の加減算器を用いた除算器の構成

前節で述べた除算用ハードウェアアルゴリズムに基づいて構成した除算器の構成は図2のようになる．なお，この図は演算対象となる被除数X，除数Yおよび出力となる商Qが共に5桁の場合である．この除算器では，RセルとQセルの2つのタイプのセルで構成される．

Rセルは，部分剰余の各桁の生成を行うセルである(図3(a))．このセルでは，冗長2進加算器のIセル，IIセルおよび減数の符号変換を行う符号変換器を用いる(図3(a)の細線に囲まれた部分)．なお，このセルで用いている冗長2進加算器は一般的な加算規則を用いた従来の1桁2ビット表現による冗長2進加算器である[2]．したがって，このセルの演算時間は従来の加算器と符号変換器のそれぞれの処理時間を足したものとなるため，処理時間は大きな値となる．

Qセルは，商の各桁を求めるセルである(図3(b))．このセルでは，前段の部分剰余の上位3桁から商の1桁を求めている．

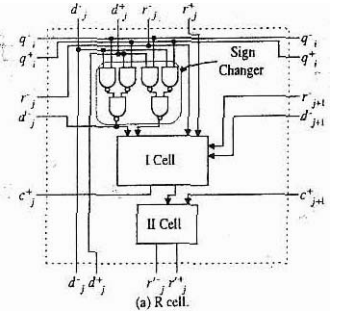


Fig. 3 Structure of each cell used in conventional divider.

4. 本加減算器を適用した高速冗長2進除算器

本章では，従来の除算アルゴリズムに本加減算器を適用した場合の高速除算器の構成法について述べる．本加減算器を適用するうえにおいて，図2に示す基本アルゴリズムは変わらないが，その構成法自体は大きく異なる．それは，従来の除算器と本除算器では冗長2進の表現法および加減算器の構成法が全く異なるためである．

最初に，部分剰余の各桁の演算を行うRセルの構成法について述べる．本報告で用いる除算アルゴリズムは，部分剰余の生成において加算および減算のほかにどちらの演算も行わずにシフトのみ行うという操作が必要とされる．それに対し，本加減算器では加算または減算のどちらか一方の演算を必ず行う構成法を用いている．そのため，制御信号によって演算結果である被加数または被減数のどちらかを選択する場合，本加減算器をそのまま適用すると図4に示すようなセレクタが必要となる．しかし，そのセレクタの部分が従来の加

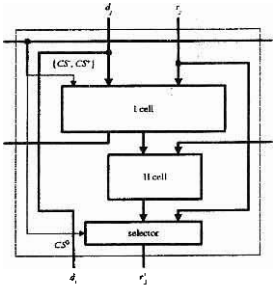


Fig. 4 Structure of R cell using selector.

減算器において減数を加数に変換する符号変換器と遅延時間が変わらないため，図4に示す構成法では本加減算器の高速性をいかせない．そこで，本加減算器を除算器向けとして高速化を図るために，セレクタの機能をIIセルに含ませることで本加減算器の並列性をより高めた新たな除算器向け冗長2進加減算器を提案する．

まず，加減算器を除算器向けとして高速化を図るためには，制御信号の生成と加減算を同時に行えるようにする必要がある．しかし，すでに提案してきた本加減算器では，制御信号の生成と加減算を同時に行うことができない．それは，提案した加減算器の高速化の手法において，加減算を行うときにはすでに制御信号が生成されていることを前提として構成されているためである[2]．

そこで，Iセルに関しては，制御信号の入力を除去し，加算結果および減算結果をそのまま出力するように改良する．すなわち，Iセルの演算では加算結果または減算結果を選択しないように構成する．これにより，制御信号の生成を待たずに加減算を並列に行うことが可能となる．その構成法では，Iセルにおいて加算器および減算器をそのまま適用し，さらに共有化できるゲートを共有化しているため，図5(a)に示すような構成になる[2][3]．

次に，IIセルに関しては，Iセルで得られた加算結果および減算結果が入力され，それを制御信号で選択する構成法を用いる．この構成法により，セレクタの機能を含ませたIIセルの構成を図5(b)に示す．図5(b)の構成法を用いることにより，加算/減算と制御信号の生成を同時に行うことが可能である．

以上の検討結果により，本提案の加減算器を本除算器に適用した場合，1桁2ビット/3ビット混合表現を用いた高速加算器と全く同等の処理時間で演算することが可能となる．なお，本提案の加減

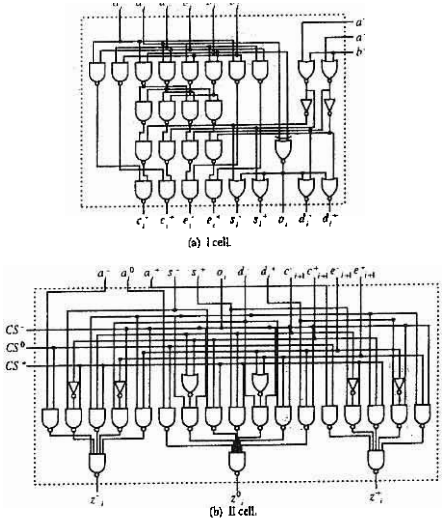


Fig. 5 Structure of proposed redundant binary adder-subtractor.

算器を用いた場合のRセルの構成を図6(a)に示す．

次に，Qセルの構成法について述べる(図6(b))．このセルでは，部分剰余の上位3桁の値から商の1桁を求める．このとき，本構成法では1桁3ビット表現を用いていることから，各桁が0であることを容易に検出できるため，従来の構成法と比較して高速に演算を行うことが可能となる．

5. 新たなアルゴリズムを用いた除算器

従来の除算アルゴリズムでは，被除数，除数共に正の場合についてのみ扱ったものであった[6]．そこで，本章では，最初に被除数，除数共に符号を考慮した新たな除算アルゴリズムについて提案する．次に，そのアルゴリズムに基づいて構成した除算器について述べる．最後に，本除算器に本提案の高速加減算器を適用した場合の構成法について述べる．

5.1 新たな除算アルゴリズム

本節では，前章で述べた従来の除算アルゴリズムにおいて，被除数，除数共に負の場合でも扱え

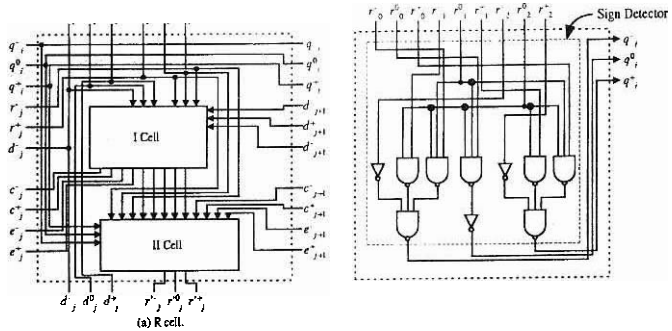


Fig. 6 Structure of each cell used in proposed divider.

るように拡張した除算アルゴリズムについて述べる．ここで，除数Yが正のとき，$\frac{1}{2} \leqq Y < 1$のようにYをあらかじめ正規化しておいて，

$2R^{(i)} \geq \frac{1}{2}$のとき，$q_i = 1$
$-\frac{1}{2} < 2R^{(i)} < \frac{1}{2}$のとき，$q_i = 0$
$2R^{(i)} \leqq \frac{1}{2}$のとき，$q_i = \bar{1}$

として，

$$R^{(i+1)} = 2R^{(i)} - q_i Y \qquad (1)$$

を求める．また，除数Yが負のとき，$-1 \leqq Y < -\frac{1}{2}$のように$Y$を正規化して，

$2R^{(i)} \geq \frac{1}{2}$のとき，$q_i = \bar{1}$
$-\frac{1}{2} < 2R^{(i)} < \frac{1}{2}$のとき，$q_i = 0$
$2R^{(i)} \leqq \frac{1}{2}$のとき，$q_i = 1$

として，(1)式を求めればよい。このことから，本提案の除算アルゴリズムは次のようになる．

X： 入力の演算数（被除数，$\frac{1}{2} \leqq |X| < 1$）
Y： 入力の演算数（除数，$\frac{1}{2} \leqq |Y| < 1$）
Q： 出力の商（$\frac{1}{2} \leqq Q < 1$）
q_i： 商の小数点i桁目
$R^{(i)}$： i回目に得られる部分剰余
r_j： $R^{(i-1)}$のj桁目
$R_s^{(j)}$： $R^{(i-1)}$における上位3ビットの符号
N： 桁数

begin
＜ステップ1＞
$R^{(0)} \leftarrow X$ （冗長2進数に変換）
$D \leftarrow Y$
＜ステップ2＞

$$R_s^{(0)} := \begin{cases} \bar{1} & \text{if } [r_0.r_1r_2]_{SD2} < 0 \\ 0 & \text{if } [r_0.r_1r_2]_{SD2} = 0 \\ 1 & \text{if } [r_0.r_1r_2]_{SD2} > 0 \end{cases}$$

if $[d_0.d_1d_2]_{SD2} < 0$
1 if $[d_0.d_1d_2]_{SD2} > 0$
$\bar{1}$ if $R_s^{(0)}$とd_sが異符号
0 if $R_s^{(0)} := 0$
1 if $R_s^{(0)}$とd_sが同符号

$R^{(1)} := R^{(0)} - q_0 D$ （冗長2進数体系で計算）
＜ステップ3＞
for $i := 1$ to N do
begin

$$R_s^{(i)} := \begin{cases} \bar{1} & \text{if } [r_0.r_1r_2]_{SD2} < 0 \\ 0 & \text{if } [r_0.r_1r_2]_{SD2} = 0 \\ 1 & \text{if } [r_0.r_1r_2]_{SD2} > 0 \end{cases}$$

$$q_i := \begin{cases} \bar{1} & \text{if } R_s^{(i)}\text{と}d_s\text{が異符号} \\ 0 & \text{if } R_s^{(i)} := 0 \\ 1 & \text{if } R_s^{(i)}\text{と}d_s\text{が同符号} \end{cases}$$

$R^{(i+1)} := 2R^{(i)} - q_i D$
（冗長2進数体系で計算）
end
＜ステップ4＞
$Q := [q_0.q_1q_2 \cdots q_N]$
end

このアルゴリズムでは，部分剰余の上位3桁だけではなく，除数の上位3桁についても判別し，その2つの結果から商の各桁を逐次的に決定していく．

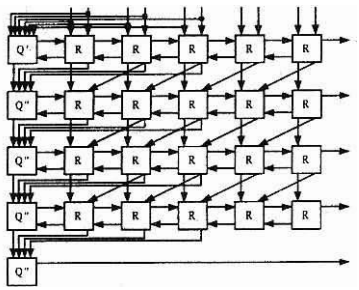


Fig. 7 Structure of divider using redundant binary representation.

5.2 従来の冗長2進加減算器を用いた除算器の構成

前節で述べた本提案の除算アルゴリズムに対して従来の1桁2ビット表現を用いた除算器の構成は図7のようになる．なお，この図は演算対象となる被除数X，除数Yおよび出力となる商Qが共に5桁の場合である．この除算器では，RセルとQ'セルおよびQ"セルの3つのタイプのセルで構成される．

Rセルは，部分剰余の各桁の生成を行うセルであり，前章の除算器におけるRセルと同じ構成である(図8(a))．

Q'セルは，1段目において被除数と除数の上位3桁から最初の商桁q_0を求めるセルである(図8(b))．ここで，上位3桁から符号を判別する回路は，図3(b)の破線に囲まれた部分に示される1桁2ビット表現用の符号検出器で構成される．また，2つの符号から商桁を決定する回路は，図3(a)の細線に囲まれた部分に示される符号変換器で構成される．

Q"セルは，Q'セルと機能は同じであるが，除数の上位3桁は2段目以降求める必要がないため，除数の上位3桁を求める回路を省略したものである(図8(c))．

5.3 本提案の加減算器を適用した除算器の構成法

本節では，本提案の高速冗長2進加減算器を適用した除算器の具体的な構成法について述べる．本冗長2進加減算器を適用するうえで図7に示す全体の構成法は変わらないものとする．

Rセルは，部分剰余の各桁の生成を行うセルであり，前章で提案した高速加減算器を用いた除算器におけるRセルと同じ構成である(図9(a))．このセルでは，前章で述べた本提案の除算器向け高速冗長2進加減算器を用いているため，従来の加減算器を用いた構成と比較して高速化が期待できる．

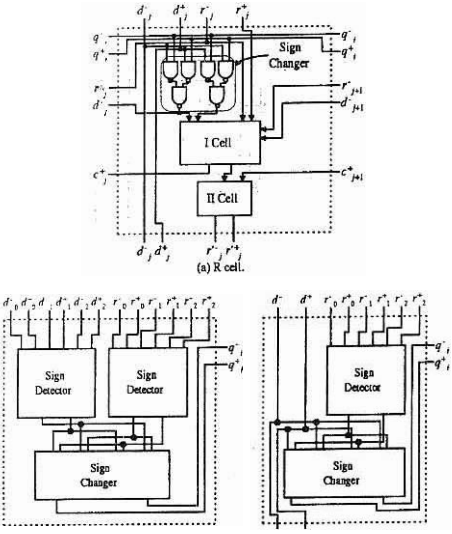
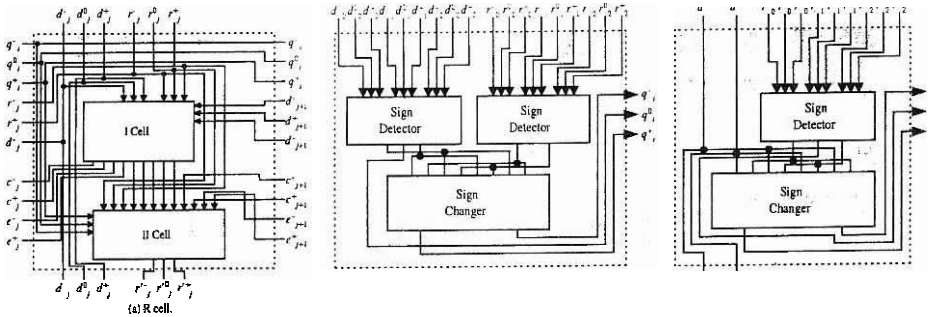


Fig. 8 Structure of each cell used in conventional divider.

Q'セルは，1段目において被除数と除数の上位3桁から最初の商桁q_0を求めるセルである(図9(b))．ここで，上位3桁から符号を判別する回路は，図6(b)の破線に囲まれた部分に示される1桁3ビット表現用の符号検出器で構成される．本除算器では，部分剰余のほかに除数の上位3桁を検出しなければならないため，符号検出器を2つ必要とする．また，部分剰余と除数の符号から商桁を決定する回路は，図3(a)の細線に囲まれた部分に示される符号変換器で構成される．ここで，本提案の除算アルゴリズムにより部分剰余の上位3桁が0の場合に限り商桁が0であり，かつ1桁を3ビット表現を用いていることから，商桁が1および$\bar{1}$の場合は部分剰余と除数の1および$\bar{1}$を表すビットを検出して商桁を生成すればよいことになる．その結果，本除算器で用いられる商桁を生成する符号変換器は，1桁2ビット表現用と同じ構成となる．そのため，制御信号である商桁を生成する回路が従来の加減算器を用いた除算器の構成と比較して簡単な回路で構成できるため，商の各桁を生成する処理時間が



(a) R cell.

減少できる．

Q"セルは，除数の上位3桁は2段目以降求める必要がないため，Q'セルの構成において除数の上位3桁を求める回路を削減したものである(図8(c))．

6. 性能評価

本章では，前章までに示した除算器の構成に対し，性能評価を行う．

6.1 論理的評価

本節では，論理的評価として単位ゲート遅延に基づく遅延時間を算出し，それを比較する．なお，単位ゲート遅延とは，NAND，NOR，NOTゲートを1Δとして換算したものである．

＜従来の除算アルゴリズムを用いた除算器の構成について＞

従来の除算器で用いられた冗長2進加減算器は冗長2進加算器に符号変換器を取り付けた構成となっている．それぞれの遅延時間は符号変換器が2Δ，加算器のIセルが6Δ，そして，IIセルが3Δである．したがって，従来の加減算器の遅延時間は11Δである．そして，その加減算器を用いた除算器の1行の遅延時間は，制御信号の生成が3Δであることから，14Δとなる．

それに対し，本提案の高速冗長2進加減算器は前章で述べたように制御信号の生成時間を遅延時間から削除することができる．したがって，本提案の高速冗長2進加減算器を適用した除算器の1行の遅延時間は7Δとなり，図10に示すような演算の流れとなる．よって，本提案の高速冗長2進加減算器を用いた除算器は，従来の除算器と比較して2倍の高速化が得られることがわかる．

＜本提案の除算アルゴリズムを用いた除算器の構成について＞

従来の加減算器を用いた除算器では，従来の除算アルゴリズムを用いた除算器と同じ加減算器を用いているため，加減算器の分の遅延時間は11Δとなる．また，部分剰余と除数の上位3桁の符号を検出する回路は従来の除算アルゴリズムを用いた除算器の制御信号を生成する回路と同じであるため3Δとなり，部分剰余と除数の上位3桁の符号から商の各桁を決定する回路が2Δであるため，16Δとなる．

それに対し，本提案の高速冗長2進加減算器を用いた除算器では，従来の除算アルゴリズムを用いた除算器と同じ加減算器であるため，7Δとなる．また，部分剰余と除数の上位3桁の符号を検出する回路は2Δであり，部分剰余と除数の上位3桁の符号から商の各桁を決定する回路は従来の除算アルゴリズムを用いた除算器の制御信号を生成する回路と同じであるため2Δである．そのため，本除算器における制御信号を生成する回路の遅延時間はこれらを足し合わせて4Δとなる．これは，加減算器のIセルと同じ遅延であるため，本提案の除算アルゴリズムを用いた構成法においても制御信号の生成時間を遅延時間から削除することができる．よって，本提案の高速冗長2進加減算器を用いた除算器は，7Δとなることから，従来の除算器と比較して約2.3倍の高速化が得られ，さらに，従来の除算アルゴリズムに基づいた構成法と同じ処理時間で演算できることがわかる．

Fig. 10 Comparison of delay time for conventional and proposed divider.

Table 1 VLSI evaluation of each divider for conventional algorithm.

(a) Power. (mW/MHz)

Digit width	Conventional divider	Proposed divider
8	1.53	2.98
16	5.92	12.37
32	22.28	53.17
64	98.99	206.87
128	392.93	715.14

(b) Area. (mm^2)

Digit width	Conventional divider	Proposed divider
8	0.30	0.50
16	1.15	2.04
32	4.34	8.69
64	19.10	33.86
128	75.80	116.86

(c) Gates. (gates)

Digit width	Conventional divider	Proposed divider
8	2487	4480
16	9525	18351
32	35758	78559
64	157726	306237
128	626051	1061722

(d) Delay time. (ns)

Digit width	Conventional divider	Proposed divider	Ratio of speed
8	186	86	2.16
16	377	167	2.26
32	748	327	2.29
64	1506	628	2.40
128	3013	1210	2.49

6.2 VLSI評価

本節では，前章で述べた除算器をVLSI設計システムPARTHENONによってVLSI設計および評価を行った[7]．評価対象を消費電力，面積，ゲート数，最大遅延時間とし，桁数を8桁から128桁まで変えて評価した．その結果について，表1では従来の除算アルゴリズムを用いた除算器の構成における評価結果を示し，表2では本提案の除算アルゴリズムを用いた除算器の構成における評価結果に示す．なお，実部品として用いたセル・ライブラリの設計ルールは0.6μmCMOSスタンダードセル(VLSIテクノロジ社)であり，電源電圧は5.0[V]とした．また，表1(c)および表2(c)のゲート数とは2入力NAND換算値であり，表1(d)の速度比とは従来の除算アルゴリズムに基づいた本提案の加減算器を用いた除算器に対する従来の加減算器を用いた除算器の演算速度の比率を表す．さらに，表2(d)の速度比とは本提案の除算アルゴリズムに基づいた本提案の加減算器を用いた除算器に対する従来の加減算器を用いた除算器の演算速度の比率を表す．

この評価結果をみてわかるように，従来の加減算器を用いた除算器と比較して本提案の加減算器を用いた除算器は2倍以上の高速化が可能となっている．

7. むすび

本稿では，まず，われわれがこれまで提案してきた1桁2ビット/3ビット混合表現を用いた加減算器を除算器向けに改良を行った．次に，その高速冗長2進加減算器を用いた応用として，冗長2進数に基づく除算器にこれを適用した場合の構成法を明らかにした．ここでは高速性の観点から高木らが提案した高速除算用ハードウェアアルゴリズムと被除数，除数の符号を考慮した除算アルゴリズムに基づいてそれぞれ考察を行った．そして，それらの構成について論理的評価として単位ゲート遅延に基づく遅延時間を算出した．さらに，それぞれのアルゴリズムを用いた除算器の構成法に対してVLSI設計システムPARTHENONを用いてVLSI設計および評価を行った．その結果，本提案の加減算器を適用した高速除算器は従来の加減算器を適用した除算器に対し，論理的評価およびPARTHENON

Table 2 VLSI evaluation of each divider for proposed algorithm.

(a) Power. (mW/MHz)

Digit width	Conventional divider	Proposed divider
8	1.63	3.33
16	6.08	13.04
32	23.40	54.59
64	101.24	211.23
128	401.85	838.43

(b) Area. (mm^2)

Digit width	Conventional divider	Proposed divider
8	0.32	0.55
16	1.19	2.14
32	4.57	8.91
64	19.57	34.50
128	77.69	136.94

(c) Gates. (gates)

Digit width	Conventional divider	Proposed divider
8	2625	4991
16	9761	19320
32	37541	80638
64	161222	312188
128	639927	1239146

(d) Delay time. (ns)

Digit width	Conventional divider	Proposed divider	Ratio of speed
8	222	106	2.09
16	438	193	2.27
32	882	374	2.36
64	1715	784	2.19
128	3430	1568	2.19

の評価からも約2倍の高速化が可能となることを明らかにした.

参考文献

1) A. Avizienis: Signed-Digit Number Representations for Fast Parallel Arithmetic, IEEE Trans. Elec. Comput., **EC-10**-9, 389/400 (1961).

2) 齊藤正人, 日野杉充希, 恒川佳隆, 三浦守: 1桁2ビット/3ビット混合表現を用いた高速冗長2進加減算器の構成法, 信学技報, CAS99-41, **99**-105, 83/90 (1999).

3) 恒川佳隆, 日野杉充希, 齊藤正人, 虻川勝已, 三浦守: 1桁2ビット/3ビット混合型高性能冗長2進加算器とその乗算器への応用, 電気学会論文誌, **119-C**-5, 644/653 (1999).

4) 日野杉充希, 恒川佳隆, 三浦守: 除算器/開平器向け高速冗長2進加減算器とその開平器への応用, 情報処理学会東北支部研究会, 資料番号00-01-10 (2000).

5) 日野杉充希, 恒川佳隆, 三浦守: 1桁2ビット/3ビット混合表現を用いた高速冗長2進加減算器の開平器への応用, 信学技報, CAS2000-53, 51/58 (2000).

6) 高木直史, 矢島脩三: 冗長2進表現を利用したVLSI向き高速除算器, 電子学会論文誌, **J67-D**-4, 450/457 (1984).

7) NTTデータ通信株式会社: PARTHENON User's Manual (1990).