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

2009-SLDM-138 (21)
2009 / 1 / 30

規則性予見演算器

佐藤 仁† 中村 次男† 冬爪 成人†
笠原 宏† 田中 照夫‡

† 東京電機大学大学院・情報環境学研究科 〒270-1355 千葉県印西市武西学園台 2-1200
‡ 東京電機大学・工学部 〒101-8457 東京都千代田区神田錦町 2-2
E-mail: † {zin, nakamura, fuyu, kasahara}@itl.sie.dendai.ac.jp, ‡ tanaka@eee.dendai.ac.jp

あらまし 減算と加算に関して入力パターンから出力の規則性を予見し，桁借り伝搬を抑制する設計法の提案を行う.また，FPGA でのシミュレーションによる既存の演算方式との性能を比較し，その結果，比較的高速で，かつ，ゲート数を抑え，モジュール化に向いた構成法であることが確認できた. このような入出力パターンの規則性予見を行う回路の設計法は算術演算用だけではなく，他の回路に適用することも可能であり，提案する設計法による加減算器について報告する.

キーワード 加減算器，演算の規則性，高速演算法，モジュール化

Foreknown Regularity Arithmetic Processing Unit

Jin Sato† Tsugio Nakamura† Narito Fuyutsume†
Hiroshi Kasahara† Teruo Tanaka‡

† Graduate School of Information Environment･TokyoDenki University2-1200 muzaigakuendai inzai-si tiba-ken Japan
‡ School of Engineering･Tokyo Denki University 2-2 Nisikityou, Kanda Tiyoda-ku Tokyo-to Japan
E-mail: † {zin, nakamura, fuyu, kasahara}@itl.sie.dendai.ac.jp, ‡ tanaka@eee.dendai.ac.jp

Abstract The paper proposes a method of designing an arithmetic unit based on the regularity of the output depending on input pattern. The advantages of this method are reduced number of gates without sacrificing high speed calculation and easy modularization scheme for high precision arithmetic unit. The soundness of this method is confirmed by the implementation on FPGA. This circuit design method based on foreknown regularity between input and output pattern can be applied not only to arithmetic operation such as addition/subtraction, but also to other circuit units.

Keyword Adder/Subtractor, Regularity of the Arithmetic, High-speed Arithmetic method, Modularizing

1. まえがき

算術演算の基本となる加減算器はこれまで用途に応じていろいろな方式が考案されてきた[1]. 最も一般的な方式には桁上げ伝搬方式があるが，桁上げが各桁間を伝搬するため，伝搬遅延とゲート数(面積)が桁数に比例する[2]. 演算速度に関しては最も高速な方式の冗長2進表現を用いた手法では一桁を 2 ビットで表現するため多くのゲート数を要する[3],[4]. 先見桁上げ方式では加算と並行して桁上げ／桁借り分を処理するため比較的高速処理が可能であるが，桁数が多くなった場合の面積増および分散処理化には不向きである.

そこで，本稿では加減算の規則性を考察し，入力の規則性から演算結果の出力パターンを予見し，基本となる桁数のモジュール内では桁借りが伝搬しない構成方式を提案する[5]. 先見桁上げ方式よりは演算速度は劣るが，桁上げ伝搬方式より高速で，ゲート数の抑制と演算桁数増に対して柔軟に対応可能で，更にモジュール化が容易な設計法について，また，FPGA による試作結果と他方式との比較について報告する.

2. 減算の規則性

2.1. モジュール内キャリー伝搬の抑制

先ず加減算器の中で減算の規則性を検討する. 最も一般的な演算器としてキャリー伝搬形がある. 各桁はそれぞれ並列に演算処理を行うが，キャリーが各桁間を伝搬していく方式であり，最上位桁の回路が結果を出力するには最下位桁からのキャリー伝搬を待たなければならない. その為に，大きな遅延が生じてしまうのが欠点である.

そこでモジュール化された回路を多段接続し，一桁ずつ計算するのではなく複数桁を同時に処理できる減算器を考える. 一桁計算し，桁借りが生じた場合，上位桁に伝搬するのではなく，入力の規則性から出力結果を予知する回路であ

る．例えば，四桁の減算器では被減数 1001 と減数 0011 という入力には 1001−0011 なので 00110 を出力する．このようにして 00000000～11111111 まで全ての出力を予知することで桁借りは生じず，高速な演算を可能にすることができる，しかし，単純にこの方法で回路構成しただけでは回路の桁数が増えた場合，回路規模が増大してしまい回路設計が困難になるという問題が生じる．そこで，1 桁～4 桁における減算器の全入出力の関係を観察し，桁が増えた時の入出力増に対する規則性を考察する．ある一定の規則で入出力が増えていることが分かれば，桁が増えても回路の入出力を予知でき，真理値表を容易に作成できるので回路の複雑化による設計の困難度を軽減できる．

2.2. 2桁減算における入出力の関係

1 桁と 2 桁減算器の真理値表(表 1，表 2) を観察していくと次のような規則性を見つけることができる．

(1) 減算器の入出力は桁借りが 0 か 1 で全て反転している

表 1 と表 2 に共通なパターンの規則性を表 2 で説明する．表 2 の入出力の桁借り B_{in} が 0 の部分に A～P の記号を割り振る．次にそれとは対象に桁借り B_{in} が 1 の部分には P'～A' を割り振る．この時，中心から上下対称に 2 つずつ入出力を見ていくと，お互いに各ビット反転の関係になっている事がわかる．

例　　P ＝01111000

　　P'＝10000111

従って，表2の下半分は上半分の反転で求めることができ，B_{in} が 0 か 1 かで分かれているので B_{in} を判定基準とし，反転を行うかどうかの制御用として，入出力に EXOR 回路を介することにより，真理値表の規模を半分に縮小することができる．

表 1　1 桁減算器の真理値表

Table. 1. Truth table for one-digit subtractor

入力				出力	
下位桁からの桁借り	上位桁 被減数	下位桁 減数		上位桁への桁借り	最下位桁
B_{in}	X	Y		B_{out}	Out
0	0	0		0	0
0	0	1		1	1
0	1	0		0	1
0	1	1		0	0
1	0	0		1	1
1	0	1		1	0
1	1	0		0	0
1	1	1		1	1

(2) 最下位桁は桁借りを生じないので，ある一定の出力を繰り返している

表 3 は前述の方法で真理値表を半分にして，入力 B_{in} を省略したものである．次に，出力について考察する．それぞれ出力を 4 分割しブロックごとに区切ると，最下位桁(一番右の出力)Out0 が同じパターンで出力されている．これは，Out0 の出力に桁借りの処理が含まないために，単純に同じ出力が繰り返されている．

(3) 最下位桁以降の出力もある一定の規則で増えている

次に，表 3 の出力の 2 桁目 Out1 を考察すると，一番上のブロック「0100」を基準として，その下はそのブロックの反転「1011」となっている．その下も同じく「1011」，最後に基準の「0100」といったように，通常，反転，反転，通常といったパターンになっている．最後に最上位桁の B_{out} について，この出力の最初と最後のブロックにおいては隣の Out1 と同じである．2 番目，3 番目ブロックにおいてはそれぞれ全部 1，全部 0 で埋まっており，Out1 との出力の違いは 2，3 番目のブロックだけなので，回路は同じものを用い，一番左の入力 X1 と Y1 から，入力が“01”なら強制的に 1，“10”なら強制的に 0 にする回路を通すことで B_{out} の出力を簡単に得ることができる．これらの規則性は全て桁数が増えていっても適用できるので，この方法で 4 つのブロックに区切ることで簡単に入出力を予知することができる．

表 2　2 桁減算器の真理値表

Table. 2. Truth table for two-digit subtractor

	下位桁からの桁借り	上位桁 被減数	上位桁 減数	下位桁 被減数	下位桁 減数		上位桁への桁借り	上位桁	最下位桁
	B_{in}	X1	Y1	X0	Y0		B_{out}	Out1	Out0
A	0	0	0	0	0		0	0	0
B	0	0	0	0	1		1	1	1
C	0	0	0	1	0		0	0	1
D	0	0	0	1	1		0	0	0
E	0	0	1	0	0		1	1	0
F	0	0	1	0	1		1	0	1
G	0	0	1	1	0		1	1	1
H	0	0	1	1	1		1	1	0
I	0	1	0	0	0		0	1	0
J	0	1	0	0	1		0	0	1
K	0	1	0	1	0		0	1	1
L	0	1	0	1	1		0	1	0
M	0	1	1	0	0		0	0	0
N	0	1	1	0	1		1	1	1
O	0	1	1	1	0		0	0	1
P	0	1	1	1	1		0	0	0
P'	1	0	0	0	0		1	1	1
O'	1	0	0	0	1		1	1	0
N'	1	0	0	1	0		0	0	0
M'	1	0	0	1	1		1	1	1
L'	1	0	1	0	0		1	0	1
K'	1	0	1	0	1		1	0	0
J'	1	0	1	1	0		1	1	0
I'	1	0	1	1	1		1	0	1
H'	1	1	0	0	0		1	0	1
G'	1	1	0	0	1		0	0	0
F'	1	1	0	1	0		0	1	0
E'	1	1	0	1	1		0	0	1
D'	1	1	1	0	0		1	1	1
C'	1	1	1	0	1		1	1	0
B'	1	1	1	1	0		0	0	0
A'	1	1	1	1	1		1	1	1

2.3. 3桁減算以上の規則性

次に 1，2，3 桁と出力の桁数が増加する場合の真理値表を比較する．表 4 は出力だけに注目した真理値表を示したものである．前述したことの繰り返しになるが，まず Out0 は桁数を問わず「0110」の繰り返しで出力されている．桁数をひとつ増やした場合，横に出力がひとつ増える．全出力は縦に 4 倍に増えるので，前の表で 4 ブロック分だったものが次の表の 1 ブロック目になっている．そして引き継がれたその 1 ブロック目以降の出力は前述の規則にしたがって決定される．

2.4. 8桁減算器のモジュール化

図 1 に表 3 で示す桁借り回路(B_{out})を示す．この規則を利用し構成した 8 桁の減算回路を図 2 に，図中のモジュール部分の 7 ブロックはすべて同一で，図 1 の回路である．図 2 では 2.2.で述べた反転機構と入力 B_{in} は含まれていない．反転機構を含んだ 8 桁減算回路のモジュールを図 3 に示す．2.2 で述べたように，中心の回路は真理値表の上半分に相当するため，入力 B_{in} と EXOR による反転機構を付加してある．

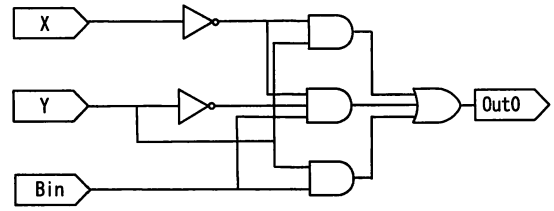


図 1　桁借り(B_{out})回路

Fig. 1. Borrow circuit (Bout)

表 3　2 桁減算器における入出力の規則

Table. 3. Rule of I/O in two-digit subtractor

上位桁		下位桁			出力					
被減数	減数	被減数	減数							
X1	Y1	X0	Y0		Bout		Out1		Out0	
0	0	0	0		0	右と同じ	0	通常	0	同じパターンになっている
0	0	0	1		1		1		1	
0	0	1	0		0		0		1	
0	0	1	1		0		0		0	
0	1	0	0		1	全部1	1	反転	0	
0	1	0	1		1		0		1	
0	1	1	0		1		1		1	
0	1	1	1		1		1		0	
1	0	0	0		0	全部0	1	反転	0	
1	0	0	1		0		0		1	
1	0	1	0		0		1		1	
1	0	1	1		0		1		0	
1	1	0	0		0	右と同じ	0	反転	0	
1	1	0	1		1		1	反転	1	
1	1	1	0		0		0	||	1	
1	1	1	1		0		0	通常	0	

4ブロック区切りの判断はここを見る

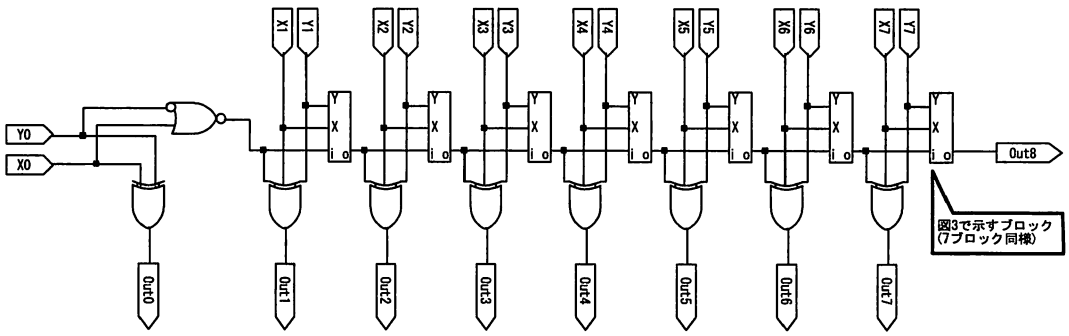
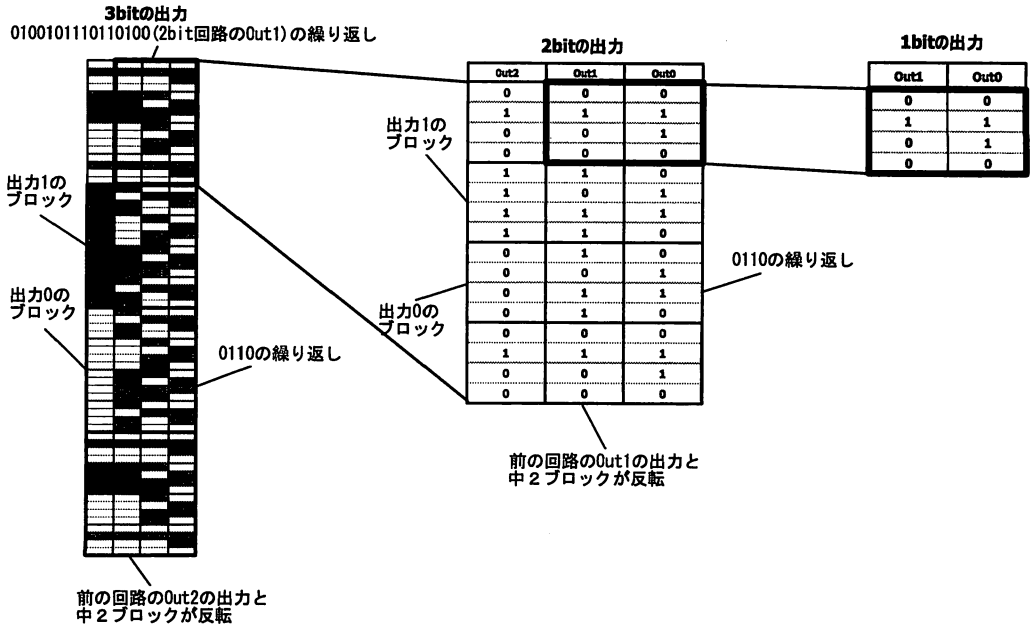


図 2　8 桁減算回路の主構成部分

Fig. 2. The main component of the 8-bit subtraction circuit

表 4　1,2,3 桁での出力の規則性

Table. 4. A regularity of the output with 1,2, and 3 digits



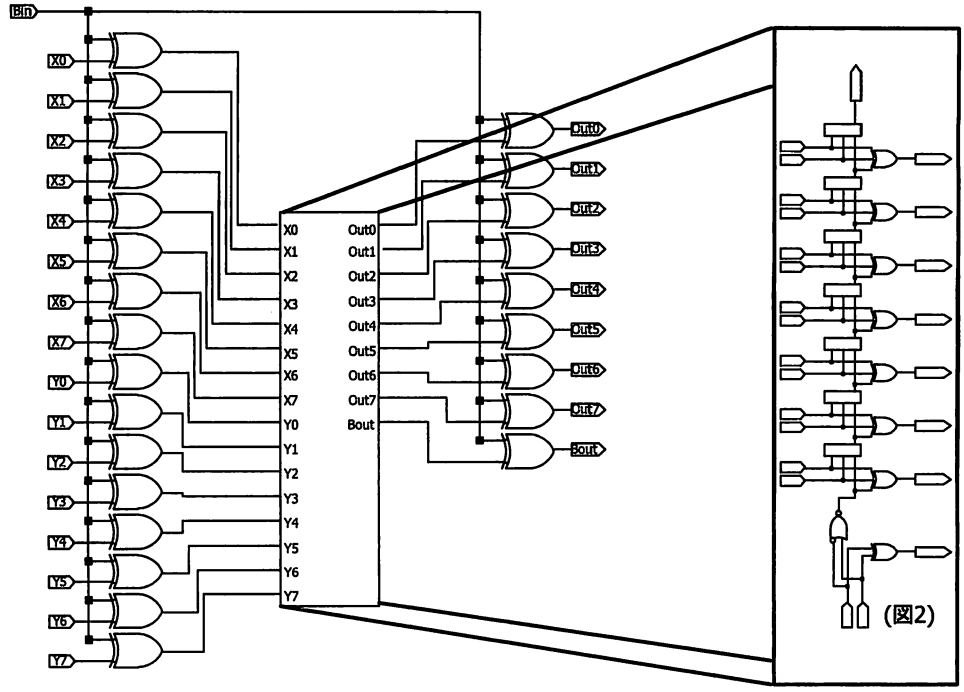


図 3　8 桁減算器のモジュール

Fig. 3. 8-bit subtractor module

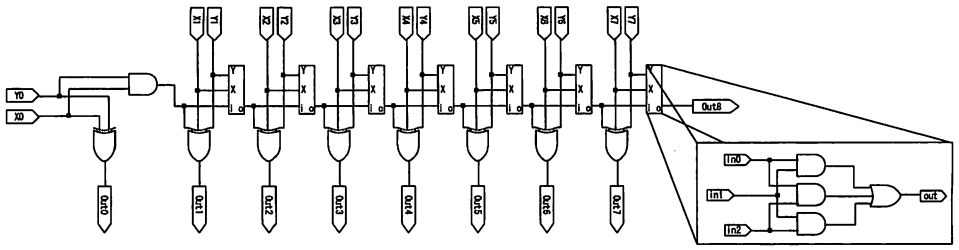


図 4　8 桁加算回路の主構成部分

Fig. 4. The main component of the 8-bit addition circuit

3. 加算器への応用

これら減算器の規則性は加算器にも共通しており,適用することにより加算器も同様に構成することができる．加算器の場合の構成図と桁上げ(Cout)回路部分を図 4 に示す．反転によって真理値表の半分を EXOR 回路によって求める事ができる規則性も共通しているため,加算器の場合も最終的な全体図は図 3 と同様な回路構成になる．

4. FPGA での試作

提案する方式は図 2 や図 4 に示したように，各桁のモジュール化により，キャリー伝搬方式のようなビット数に比例した各桁をカスケードに接続して演算器を構成することができる．試作においては 8 ビット単位のモジュール化を行い，それをカスケード接続して演算桁数を増加できることをシミュレーションと FPGA による試作によって確認した．Xilinx 社の FPGA Virtex4，xc4vlx60 をターゲットデバイスとして動作確認を行った．設計ツールに Xilinx 社 ISE Foundation 9.1i を，シミュレータには Mentor Grahpics 社 ModelSim XE III 6.2g を使用した．

5. 既存方式との比較

FPGA でのシミュレーションによる，既存の桁上げ伝搬方式と先見桁上げ方式との性能を比較した．図 5，図 6 より，実行速度では先見桁上げ方式に及ばないものの，桁上げ伝搬方式よりは早く，ゲート数においては他の 2 方式よりも少ないゲート数で構成できることを示している．しかし，FPGA での実測結果であり，NAND ゲートのセミカスタムやフルカスタムでの実装においては，桁上げ伝搬方式のゲート数が最も少なく実現できる．また，ビット数の増加に伴いゲート数の差が広がっていくので，筆者らが提案する方式は多数ビットの演算や加減算器の多段構成となる乗除算器等のゲート数抑制に有効で，かつ高速演算が期待できる．図 5 と図 6 は 4 ビットのブロック先見桁借りを演算ビットに合わせて複数個接続，たとえば 64 ビット演算では 16 ブロックで構成した結果の特性を示したものである．その他の方式も同様なブロック(モジュール化)を行っている．同じく減算器のシミュレーション結果を図 7，図 8 に示す。加算器と同様な結果が伺える．図 9 と図 10 では全ての方式がブロック方式ではなく，4，8 および 16 ビットで構成した場合の比較を示す(先見桁借り方式ではビット数増加に伴い，ゲート数が急激に増加するため，32 と 64 ビットは実測していないのでグラフは 16 ビットまでとした)．両図より，桁借り伝搬方式はモジュール化と相違はないが，先見桁借り方式と規則性予見方式ではより高速化が進む．ゲート数に関しては，モジュール化を行わない場合と比較しても 8bit の時点で提案する規則性予見方式が先見桁借り方式よりも少ないゲート数で構成でき，どの桁数でモジュール化を行っても規則性予見方式での構成の方がゲート数を大幅に低減できることを示している．

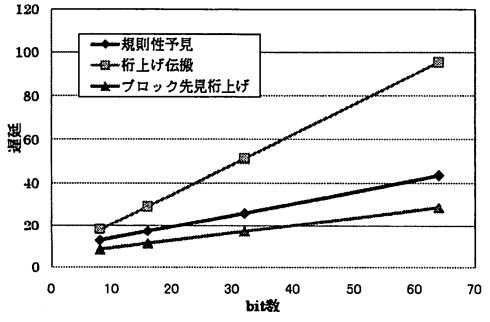


図 5　ビット数に対する遅延(加算器)

Fig. 5. Delay for the number of bit(adder)

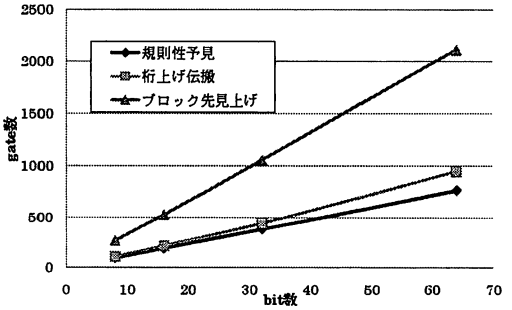


図 6　ビット数に対するゲート数(加算器)

Fig. 6. Gate count for the number of bit(adder)

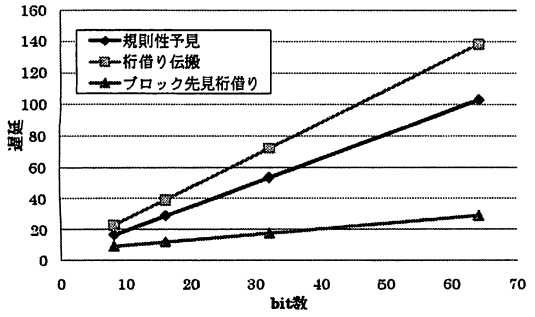


図 7　ビット数に対する遅延(減算器)

Fig. 7. Delay for the number of bit(subtractor)

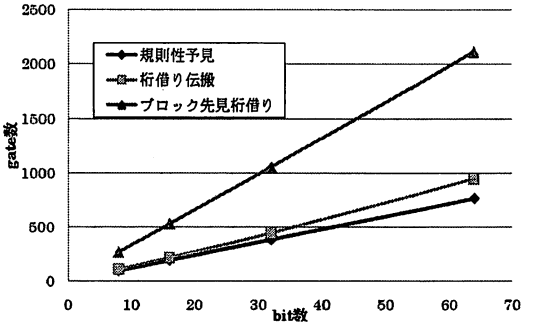


図 8　ビット数に対するゲート数(減算器)

Fig. 8. Gate count for the number of bit(subtractor)

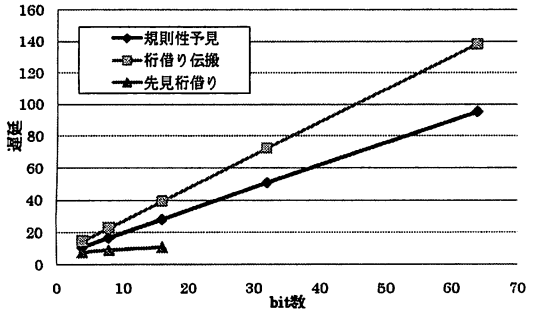


図 9　モジュール化を行わない場合の遅延

Fig. 9. Delay without modularization

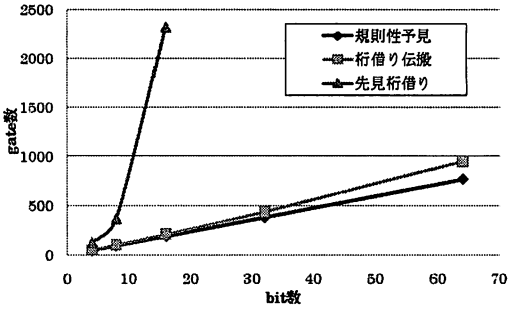


図 10　モジュール化を行わない場合のゲート数

Fig. 10. Gate count without modularization

6. まとめ

減算と加算に関して，入力パターンから出力の規則性を予見し，キャリー伝搬を抑制する設計法の提案を行った．その結果，比較的高速で，かつ，ゲート数を抑え，モジュール化に向いた構成法であることが確認できた．ゲート数の低減は消費電力の低減にも影響し，モジュール化に向いた構成法であることやゲート数を抑えた設計法であることから，高集積，多段構成に向いており，多くの演算や多数ビットの演算を行なう乗除算器への応用および暗号処理などへの応用が期待できる．

現在は冗長２進加減算器についてもゲート数削減を目的に開発を進めている．また，このような入出力パターンの規則性予見を行う回路の設計法自体も演算回路のみだけではなく，従来の回路に適用することによるゲート数の抑制やモジュール設計の容易化について研究中である．

文　献

[1] Kai Hwang : “Computer Arithmetic-PRINCIPLES, ARCHITECTUR AND DESIGN”, Jhon Wiley & Sons, Inc (1979).

[2] 中村次男:「ディジタル回路の基礎」，日本理工出版会(1992).

[3] Tsugio Nakamura, Kazuhiro Abe, Narito Fuyutsume, Hiroshi Kasahara, and Teruo Tanaka: ”Super-high Speed, Accuracy, and Modularized Residue Number System based on Redundant Binary Representation”, IEEJ Trans.EIS, Vol.125, No.6, pp.876- 886(2005).

[4] Yoshitaka Tsunekawa, Mitsuki Hinosugi, Masato Saito, Katsumi Abukawa, and Mamoru Miura : ”High-Performance Redundant Binary Adder Representing Each Digit by Hybrid 2 Bits /3 Bits and Its Application to Multiplier”, T.IEE Japan, Vol.119-C, No.5, pp.644-653(1999).
恒川 佳隆・日野杉 充希・斉藤 正人・虻川 勝巳・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」, 電学論 C, Vol.119-C,No.5,pp.644-653(1999).

[5] 佐藤 仁・中村 次男・冬爪 成人・笠原 宏・田中照夫:「規則性予見減算器」,電気学会,電子回路研究会,ECT-08-9,pp.37-42(2008.1)