システムＬＳＩ設計技術　108－14
(2003. 1. 29)

グレイコードによる上位ビットからの任意精度演算

米本　明弘†　久門　尚史†　後藤　雅典†　奥村　浩士†

† 京都大学大学院 工学研究科 電気工学専攻
〒606-8501 京都市左京区吉田本町
E-mail: †{yone,hisakado,kohshi}@kuee.kyoto-u.ac.jp

あらまし　グレイコードによる上位ビットからの四則演算アルゴリズムを提案する．これまでグレイコードはさまざまな応用で用いられてきたが，グレイコード上の算術演算アルゴリズムについては加減算について報告されているだけである．グレイコードの特徴であるその位相構造は，上位ビットからの演算に利用できる．通常，上位ビットからの演算には冗長数が用いられるが，グレイコードを用いると一意的な演算結果を得られる．
キーワード　グレイコード，上位ビットからの演算，ビットシリアル，任意精度演算，冗長数

On-line Arbitrary Precision Arithmetic Using Gray Code

Akihiro YONEMOTO†, Takashi HISAKADO†, Masanori GOTO†, and Kohshi OKUMURA†

† Department of Electrical Engineering, Kyoto University
Yoshida-honmachi, Sakyo-ku, Kyoto, 606-8501 Japan
E-mail: †{yone,hisakado,kohshi}@kuee.kyoto-u.ac.jp

Abstract　This paper presents on-line arithmetics using Gray codes. Although the Gray code has been used in many applications, arithmetics on it has not been reported. The Gray code has topological structure suitable for on-line algorithms. We propose bit serial arithmetics from the most significant bits utilizing the topological property. Although the signed digit number representation, which is usually used for on-line algorithms, has redundancy in its representation of numbers, the Gray code representation realizes the on-line arithmetics without redundancy.
Key words　Gray codes, on-line arithmetic, bit serial, arbitrary precision arithmetic, redundant numbers

1. はじめに

グレイコードは自然数のコード化の１つであり，フランク・グレイがシャフトエンコーダに用いたことからそのように呼ばれている [1]．グレイコードの大きな特徴は，それが表わす数が１増減したときに，コード中の１ビットしか変化しないことである．この位相構造を利用して，これまで A/D コンバータ [2] やディジタル回路設計 [2], [3]，ケーリーグラフ [4]，画像圧縮 [5]，遺伝アルゴリズムなど，さまざまに応用されてきた [6], [7].

また最近，立木 [8] は実関数の計算可能性を定義するために実数のグレイコード展開を用いている．グレイコード展開とは {0,1} からなる無限列のことで，その中に不定を意味する $\perp$ が高々1 つのみ許される．グレイコード展開を用いると，その位相構造によって上位ビットからの演算が可能となる．文献 [8] ではその例として加減算アルゴリズムが示されているが，それ以外の演算の具体的アルゴリズムについては報告されていない．

上位ビットから演算するという方法は，1977 年に Ercegovac と Trivedi によって提案された [9], [10]．両氏による方法では入出力に冗長数表現が用いられている [11], [12]．冗長数を用いることで上位ビットからの演算が可能になり，また，内部の演算においてキャリーの伝搬を抑えることができる．上位ビットからの演算アルゴリズムでは，入力が上位ビットから順に与えられると，それに応じて出力がやはり上位ビットから順に求まっていく．この操作を任意に継続できるという意味で，上位ビットからの演算により任意精度演算が行える．

本報告ではグレイコードを用いた上位ビットからの四則演算アルゴリズムを提案する．冗長数表現とは異なり，グレイコードによる上位ビットからの演算では，入出力のコードが一意に定まるという特徴を持つ．

2. グレイコード

表１に２進数とグレイコードの対応を示す．特にグレイコードにおいて，隣接する各行で変化するのは１ビットのみである．

上位ビットからの演算は，下位ビットを知ることなく演算を行うという点から区間演算 [13] と等価である．そこで，次のような区間 X を考える．

$$X = [\underline{X}, \overline{X}] \quad (1)$$

$$= [(x-1)2^{-n}, (x+1)2^{-n}] \quad (x, n \text{ は整数}) \quad (2)$$

ここで x は区間の中点 $x2^{-n}$ を表わす整数である．また，n のことをスケール指数と呼ぶことにする．区間 X の幅は 2^{-n+1} であり，これは区間数の精度を表わしている．図 1 に，区間 $[0,1]$ における区間の 2 進数とグレイコードによる表現を示す．図中，黒帯は対応するビットが 1 であることを示し，白帯はそれが 0 であることを示す．下図における点線は視覚的な補助線で特別な意味はない．2 進数の場合，1/4 や 1/2 などの $m2^{-k}$ $(m, k$: 整数) となる点において複数のビットが同時に変化する．一方，グレイコードの場合は複数のビットが同時に変化する点は存在しない．

表 2 に，$n = 3$ における区間 X の 2 進表現とグレイコード表現を示す．表中の $\perp$ はそのビットが不定であることを示す．2 進表現では上述の複数のビットが同時に変化することに対応し，複数のビットが不定になり，X を過大評価してしまう場合がある．一方，グレイコード表現では 1 つのビットが不定になるだけで，X を過不足なく表現できる．この性質を利用すると，グレイコードを用いることにより上位ビットからの演算が可能となる．

表 1　2 進数とグレイコードによる表現

数	2 進数	グレイコード
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

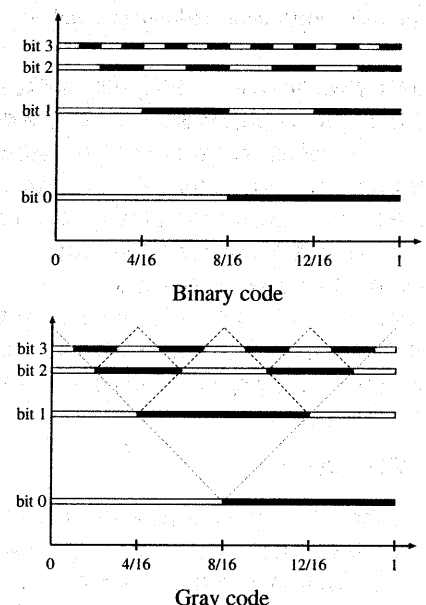

図 1　2 進数とグレイコードによる区間の表現

次に，区間 X のグレイコード表現 $x_{\mathrm{G}n_x}$ を定義する．スケール指数 n_x を固定すると，区間 $[0,1]$ に含まれる区間 X のグレイコード表現は

$$x_{\mathrm{G}n_x} = 0.x_0x_1\cdots x_{n_x-1}x_{n_x} \longleftrightarrow X \quad (3)$$

となる．ここで，$x_0, \cdots, x_{n_x-1}$ のどれかちょうど 1 つが不定ビット $\perp$ であり，それを $\perp_1$ と呼ぶことにする．また，x_{n_x} は常に $\perp$ であり，これを $\perp_2$ と呼ぶことにする．矢印 $\longleftrightarrow$ はグレイコードと区間の対応を示す．こうして，グレイコード $x_{\mathrm{G}n_x}$ は集合 $\{0,1\}$ および $\perp_1$ と $\perp_2$ で表わされる．また，区間 X として $[0,1]$ より広いものを考える場合は，(3) を

$$x_{\mathrm{G}n_x} = \cdots x_{(-2)}x_{(-1)}.x_0x_1\cdots x_{n_x-1}x_{n_x} \quad (4)$$

のように拡張するものとする．

グレイコード $x_{\mathrm{G}n_x}$ の各ビット x_i は次のように求められる．

（1）　奇数 j に対して $X \subset [j-\frac{1}{2}, j+\frac{1}{2}]2^{-i}$ ならば $x_i = 1$

（2）　偶数 j に対して $X \subset [j-\frac{1}{2}, j+\frac{1}{2}]2^{-i}$ ならば $x_i = 0$

（3）　$x \times 2^{-n_x} = (j-\frac{1}{2}) \times 2^{-i}$ ならば $x_i = \perp_1$

特に $x_k = \perp_1$ のときは，図 1 の下図より $x_{k+1} = 1$, $x_{k+2}, \ldots, x_{n_x-1} = 0$ である．

また，符号付きグレイコードとして，区間 $[-1,1]$ に含まれるグレイコードを次のように定義する．

表 2　区間 X の 2 進数とグレイコードによる表現 ($n = 3$. $\perp$ は不定ビットを表わす)

x	2 進数	グレイコード
1	0. 0 0 $\perp$	0. 0 0 $\perp$
2	0. 0 $\perp$ $\perp$	0. 0 $\perp$ 1
3	0. 0 1 $\perp$	0. 0 1 $\perp$
4	0. $\perp$ $\perp$ $\perp$	0. $\perp$ 1 0
5	0. 1 0 $\perp$	0. 1 1 $\perp$
6	0. 1 $\perp$ $\perp$	0. 1 $\perp$ 1
7	0. 1 1 $\perp$	0. 1 0 $\perp$

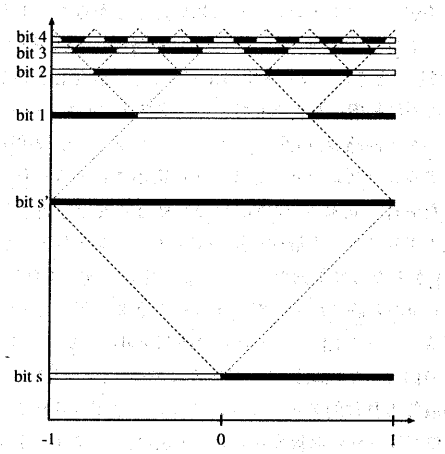

図 2　符号付きグレイコード

$$x_{\mathrm{G}n_x} = x_{\mathrm{GS}} x_{\mathrm{GS}'} . x_0 x_1 \cdots x_{n_x-1} x_{n_x} \longleftrightarrow X \quad (5)$$

ここで，x_{GS} は以下のように定める．

（1） $X \subset [0,1]$ ならば $x_{\mathrm{GS}} = 1$

（2） $X \subset [-1,0]$ ならば $x_{\mathrm{GS}} = 0$

（3） $0 \in X$ ならば $x_{\mathrm{GS}} = \perp_1$

$x_{\mathrm{GS}'}$ は常に $x_{\mathrm{GS}'} = 1$ である．図2に符号付きグレイコードを用いた区間の表現を示す．グレイコードの対称性から，正負の数の表現にはこのように絶対値表現が適切である．

3. グレイコードによる四則演算

3.1 グレイコードによる算術演算

グレイコードによる算術演算を，そのコードに対応する区間表現を用いて定義する．2つの入力グレイコードを $a_{\mathrm{G}}, b_{\mathrm{G}}$，出力グレイコードを c_{G} とし，これらに対応する区間をそれぞれ A, B, C とする．

$$a_{\mathrm{G}} \longleftrightarrow A, \quad b_{\mathrm{G}} \longleftrightarrow B, \quad c_{\mathrm{G}} \longleftrightarrow C \quad (6)$$

また，$\circ$ を2入力の区間演算子としたとき，この演算子を次のようにグレイコードに拡張する．

$$c_{\mathrm{G}} = a_{\mathrm{G}} \circ b_{\mathrm{G}} \quad (7)$$

ただし，ここで c_{G} は，対応する区間 C が

$$C \supset A \circ B \quad (8)$$

を満たすようなスケール指数 n_c が最大のグレイコードである．

3.2 グレイコードによる算術演算アルゴリズム

本節ではグレイコードによる上位ビットからのビットシリアルな演算アルゴリズムを導出する．つまり，入力グレイコードの精度向上に対する出力グレイコードの精度向上を与えるアルゴリズムを導く．グレイコードの精度向上とは，$\perp_1$ または $\perp_2$ の値が定まることを意味する．

まず，グレイコード x_{G} の $\perp_1$ あるいは $\perp_2$ が定まったときの，対応する区間 X の縮小について考える．そのために，符号無しグレイコード x_{G} のパリティ P_x を次式で定義する．

$$P_x = \bigoplus_{j=-\infty}^{i_{\mathrm{b}}-1} x_j \quad (9)$$

ただし，i_b は $\perp_1$ の位置を，$\oplus$ は排他的論理和をそれぞれ表わす．このパリティ P_x を用いて，グレイコード x_{G} の精度向上を以下の3通りに分類する．

（1） $\perp_1$ が P_x に定まる

（2） $\perp_1$ が $\overline{P_x}$ に定まる

（3） $\perp_2$ が1あるいは0に定まる

ただし，$\overline{P_x}$ は P_x の論理否定である．これに対応して，区間 $X = [x-1, x+1]2^{-n}$ は

（1） 左半区間 $[x-1, x]2^{-n}$

（2） 右半区間 $[x, x+1]2^{-n}$

（3） 中心の区間 $[x-1/2, x+1/2]2^{-n}$

にそれぞれ縮小する．すなわち，パリティ P_x は x_{G} の $\perp_1$ が0か1の値に定まることによって，対応する区間 X が左半区間か右半区間のどちらに縮小するかを与える．

まず(1)，(2)の場合の例として，次のようなグレイコード $x_{\mathrm{G}2}$ を考える．

$$x_{\mathrm{G}2} = 0.1\perp_1\perp_2 \longleftrightarrow X = [2,4]2^{-2} = [1/2, 1] \quad (10)$$

この場合，$i_{\mathrm{b}} = 1, P_x = 1$ である．ここで $x_{\mathrm{G}2}$ の $\perp_1$ が $1 = P_x$ に定まり，$x_{\mathrm{G}3} = 0.11\perp_1\perp_2$ となったとすると，対応する区間 X はその左半区間 $[0, 3/4]$ に縮小する．また，$\perp_1$ が $0 = \overline{P_x}$ に定まり，$x_{\mathrm{G}3} = 0.10\perp_1\perp_2$ となったとすると，区間 X はその右半区間 $[3/4, 1]$ に縮小する．より一般に，$\perp_1$ が P_x に定まった場合は対応する区間 X がその左半区間に縮小し，$\perp_1$ が $\overline{P_x}$ に定まった場合は X がその右半区間に縮小する．

一方，(3)の場合，すなわち $x_{\mathrm{G}n_x}$ の $\perp_2$ が1あるいは0に定まった場合は，対応する区間 X は，幅が半分でその中心の区間に縮小する．このとき，$\perp_2$ が1か0のどちらの値に定まるかは，$\perp_2$ が $\perp_1$ の直後にあるかどうかに依存する．例えば，$x_{\mathrm{G}n_x} = \cdots\perp_1\perp_2$ の $\perp_2$ が3回連続して定められたとき，$x_{\mathrm{G}n_x+3}$ と対応する区間 X はそれぞれ $x_{\mathrm{G}n_x+3} = \cdots\perp_1 100\perp_2$，$X = [x - 1/8, x + 1/8]2^{-n_x}$ となる．

表3に，これらグレイコード x_{G} の精度向上と対応する区間 X の縮小の関係をまとめる．ただし，記号 := は代入を表わす．

次に，一般の2入力グレイコード演算子 $\circ$ に対する上位ビットからの演算アルゴリズムを示すために，$\underline{d}, \overline{d}$ を次のように定義する (図3を参照)．

$$\underline{d} = 2^{n_c}(\underline{A \circ B} - \underline{C}) = 2^{n_c}\underline{A \circ B} - (c-1) \quad (11)$$

$$\overline{d} = 2^{n_c}(\overline{C} - \overline{A \circ B}) = (c+1) - 2^{n_c}\overline{A \circ B} \quad (12)$$

これらを用いると，式(8)の条件は，

$$\underline{d} \geq 0, \quad \overline{d} \geq 0 \quad (13)$$

と書ける．また，スケール指数 n_c が(8)を満たす最大のものであるという条件は，

$$\underline{d} < 1 \text{ かつ } \overline{d} < 1 \text{ かつ } (\underline{d} < 1/2 \text{ または } \overline{d} < 1/2) \quad (14)$$

表3 グレイコードの精度向上 $x_{\mathrm{G}n_x} \rightarrow x_{\mathrm{G}n_x+1}$ と対応する区間の縮小 $X = [x-1, x+1]2^{-n_x}$

$\perp_1, \perp_2$	x	X	縮小方向
$\perp_1 := P_x$	$x := 2(x - 1/2)$	$\overline{X} := \overline{X} - 2^{-n_x}$	左
$\perp_1 := \overline{P_x}$	$x := 2(x + 1/2)$	$\underline{X} := \underline{X} + 2^{-n_x}$	右
$\perp_2 := 1^*, 0$	$x := 2x$	$\underline{X} := \underline{X} + 2^{-n_x-1}$ $\overline{X} := \overline{X} - 2^{-n_x-1}$	中心

(* $\perp_1$ の直後に $\perp_2$ があるとき)

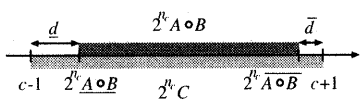

図3 $\underline{d}, \overline{d}$ の定義

と書ける．以下にグレイコードによる上位ビットからの演算アルゴリズムを与える．

グレイコードによる上位ビットからの演算アルゴリズム

S0. (14) を満たすように $a_{\mathrm{G}n_a}, b_{\mathrm{G}n_b}, c_{\mathrm{G}n_c}, \underline{d}, \overline{d}$ を初期化する．

S1. 入力 $a_{\mathrm{G}n_a}$ か $b_{\mathrm{G}n_b}$ の精度が 1 ビット向上，すなわちスケール指数 n_a あるいは n_b が 1 増加したら，$\underline{d}$ と $\overline{d}$ を更新する．

S2. 表 4 の条件のどれか 1 つが満たされる限り，出力 $c_{\mathrm{G}n_c}$ および $\underline{d}, \overline{d}$ を更新し，$n_c := n_c + 1$ とする．そして **S1.** へ．

表 4 における 3 つの条件は，図 4 に示すように区間 C がどのように縮小したかに対応する．

グレイコードによる種々の演算アルゴリズムは，その演算に対して **S1.** で必要となる $\underline{d}, \overline{d}$ の更新規則を与えることで最終的に決定される．ただし，その更新規則で更新された $\underline{d}, \overline{d}$ は (13) および (14) を満たす必要がある．

3.3 加 減 算

区間演算における加算は次式で与えられる [13]．

$$A + B = [\underline{A} + \underline{B}, \overline{A} + \overline{B}] \tag{15}$$

$$= [(a-1)2^{-n_a} + (b-1)2^{-n_b},\ (a+1)2^{-n_a} + (b+1)2^{-n_b}] \tag{16}$$

ここで a, b は整数である．すると (11), (12) より

$$\underline{d} = 2^{n_c}(\underline{A} + \underline{B}) - (c-1) \tag{17}$$

$$\overline{d} = (c+1) - 2^{n_c}(\overline{A} + \overline{B}) \tag{18}$$

である．ただし c は整数．加算は入力に関して対称な演算なので，$a_{\mathrm{G}n_a}$ の精度向上について考えればよい．表 3 の X 行と (17), (18) より，加算に関する $\underline{d}, \overline{d}$ の更新規則は表 5 のようになる．例えば，入力 $a_{\mathrm{G}n_a}$ が $\perp_1 := \overline{P_a}$ と更新されたとする．更新後の値に $'$ をつけ，

$$n_a' = n_a + 1, \tag{19}$$

$$a' = 2(a + 1/2), \tag{20}$$

などのように表わすと，(17) より

$$\begin{aligned}\underline{d}' &= 2^{n_c}\{(a'-1)2^{-n_a'} + (b-1)2^{-n_b}\} - (c-1)\\ &= 2^{n_c}[\{2(a+1/2)-1\}2^{-n_a-1} + (b-1)2^{-n_b}] - (c-1)\\ &= 2^{n_c}[\{(a-1)+1\}2^{-n_a} + (b-1)2^{-n_b}] - (c-1)\\ &= \underline{d} + 2^{n_c-n_a}\end{aligned} \tag{21}$$

を得る．表 5 の $\perp_1 := \overline{P_a}$ 行における $\underline{d}$ の更新規則はこれにもとづく．同様にして他の更新規則も得られる．

減算については，正負の数の表現に絶対値表現を用いているため，b_{G} の符号ビット b_{GS} をその論理否定に置き換えて加算を行えばよい．

初期条件として，例えば絶対値が 1 以下の 2 つのグレイコードの加減算を行うときには，以下のようにとればよい．

$$a_{\mathrm{G}0} = (\perp_1 1).\perp_2 \longleftrightarrow A = [-1, 1]2^0 = [-1, 1] \tag{22}$$

$$b_{\mathrm{G}0} = (\perp_1 1).\perp_2 \longleftrightarrow B = [-1, 1]2^0 = [-1, 1] \tag{23}$$

$$c_{\mathrm{G}(-1)} = (\perp_1 1)\perp_2. \longleftrightarrow C = [-1, 1]2^1 = [-2, 2] \tag{24}$$

$$\underline{d} = 0, \quad \overline{d} = 0 \tag{25}$$

ここで，括弧内は $a_{\mathrm{GS}}a_{\mathrm{GS}'}$ などの符号ビットを表わす．

表 5 加算における $\underline{d}, \overline{d}$ の更新規則

$a_{\mathrm{G}n_a} \to a_{\mathrm{G}n_a+1}$	$\underline{d}, \overline{d}$
$\perp_1 := P_a$	$\overline{d} := \overline{d} + 2^{n_c-n_a}$
$\perp_1 := \overline{P_a}$	$\underline{d} := \underline{d} + 2^{n_c-n_a}$
$\perp_2 := 1$ ($\perp_1$ の直後のとき), 0	$\underline{d} := \underline{d} + 2^{n_c-n_a-1}$ $\overline{d} := \overline{d} + 2^{n_c-n_a-1}$

表 4 出力の更新規則 $c_{\mathrm{G}n_c} \to c_{\mathrm{G}n_c+1}$

条件	c_{G} および $\underline{d}, \overline{d}$ の更新規則	
$\overline{d} \geqq 1$	$\perp_1 := P_c$	$\underline{d} := 2\underline{d}$ $\overline{d} := 2(\overline{d}-1)$
$\underline{d} \geqq 1$	$\perp_1 := \overline{P_c}$	$\underline{d} := 2(\underline{d}-1)$ $\overline{d} := 2\overline{d}$
$\underline{d} \geqq 1/2$ かつ $\overline{d} \geqq 1/2$	$\perp_2 := 1$ ($\perp_1$ の直後のとき), 0	$\underline{d} := 2(\underline{d}-1/2)$ $\overline{d} := 2(\overline{d}-1/2)$

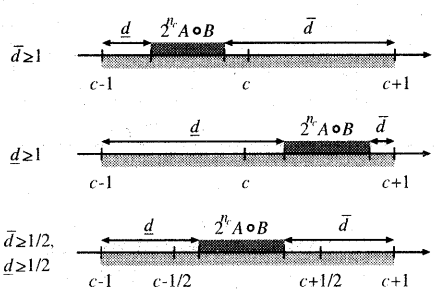

図 4 表 4 における 3 つの条件

3.4 乗　　算

まず，符号無しグレイコードの乗算を考える．区間演算における非負の乗算は次式で与えられる．

$$A \times B = [\underline{A} \times \underline{B}, \overline{A} \times \overline{B}] \tag{26}$$

$$= [(a-1)(b-1), (a+1)(b+1)]2^{-(n_a+n_b)} \tag{27}$$

ここで a, b は正整数である．すると (11), (12) より

$$\underline{d} = 2^{n_c}(\underline{A} \times \underline{B}) - (c-1) \tag{28}$$

$$\overline{d} = (c+1) - 2^{n_c}(\overline{A} \times \overline{B}) \tag{29}$$

である．ただし c は正整数．乗算は入力に関して対称な演算なので，$a_{\mathrm{G}n_a}$ の精度向上について考えればよい．表 3 の X 行と (28), (29) より，乗算に関する $\underline{d}, \overline{d}$ の更新規則は表 6 のようになる．

次に，符号付きグレイコードの乗算を考える．グレイコード x_{G} に対応する区間 X が 0 をその中に含むのは $x = 0$ のときのみで，このとき X は 0 に関して対称である．入力の少なくともどちらか一方が 0 を含む場合，出力区間も 0 を含み，0 に

関して対称になる．このことから，符号ビットと仮数ビットを別々に処理することで符号付き乗算を行える．すなわち，出力 c_G の符号ビット c_{GS} は表7に従って求め，仮数ビットに対しては上述の非負の乗算アルゴリズムを適用する．

初期条件として，例えば絶対値が1以下の2つのグレイコードの乗算を行うときには，以下のようにとればよい．

$$a_{G0} = (\perp_1 1).\perp_2 \longleftrightarrow A = [-1,1]2^0 = [-1,1] \quad (30)$$

$$b_{G0} = (\perp_1 1).\perp_2 \longleftrightarrow B = [-1,1]2^0 = [-1,1] \quad (31)$$

$$c_{G(-1)} = (\perp_1 1).\perp_2 \longleftrightarrow C = [-1,1]2^0 = [-1,1] \quad (32)$$

$$\underline{d} = 0, \quad \overline{d} = 0 \quad (33)$$

表6　乗算における $\underline{d},\overline{d}$ の更新規則

$a_{Gn_a} \to a_{Gn_a+1}$	$\underline{d},\overline{d}$
$\perp_1 := P_a$,	$\overline{d} := \overline{d} + (b+1)2^{n_c-n_a-n_b}$
$\perp_1 := \overline{P_a}$,	$\underline{d} := \underline{d} + (b-1)2^{n_c-n_a-n_b}$
$\perp_2 := 1$ ($\perp_1$ の直後のとき), 0	$\underline{d} := \underline{d} + (b-1)2^{n_c-n_a-n_b-1}$ $\overline{d} := \overline{d} + (b+1)2^{n_c-n_a-n_b-1}$

表7　乗算における出力の符号ビット c_{GS}

$a_{GS} \backslash b_{GS}$	0	$\perp_1$	1
0	1	$\perp_1$	0
$\perp_1$	$\perp_1$	$\perp_1$	$\perp_1$
1	0	$\perp_1$	1

3.5 除　　算

乗算と同様に，まず符号無しグレイコードの除算を考える．区間演算における非負の除算は次式で与えられる．

$$\frac{A}{B} = \left[\frac{\underline{A}}{\overline{B}}, \frac{\overline{A}}{\underline{B}}\right] = \left[\frac{a-1}{b+1}, \frac{a+1}{b-1}\right] 2^{n_b-n_a} \quad (34)$$

ここで a,b は正整数である．すると (11), (12) より

$$\underline{d} = 2^{n_c}\frac{\underline{A}}{\overline{B}} - (c-1) = 2^{n_c-n_a+n_b}\frac{a-1}{b+1} - (c-1) \quad (35)$$

$$\overline{d} = (c+1) - 2^{n_c}\frac{\overline{A}}{\underline{B}} = (c+1) - 2^{n_c-n_a+n_b}\frac{a+1}{b-1} \quad (36)$$

である．ただし c は正整数．除算の場合は，$(a-1)/(b+1)$ と $(a+1)/(b-1)$ の2つの整数除算を行う必要がある．しかし，表4より c_G の更新に必要な $\underline{d},\overline{d}$ の精度は $1/2$ であるから，これらを以下の方法で効率的に計算できる．まず初期状態として，

$$a_{G1} = (11).\perp_1\perp_2 \longleftrightarrow A = [0,2]2^{-1} = [0,1] \quad (37)$$

$$b_{G2} = (11).1\perp_1\perp_2 \longleftrightarrow B = [2,4]2^{-2} = [1/2,1] \quad (38)$$

$$c_{G0} = (11)\perp_1.\perp_2 \longleftrightarrow C = [0,2]2^0 = [0,2] \quad (39)$$

$$\underline{d} = 0, \quad \overline{d} = 0 \quad (40)$$

あるいは

$$a_{G1} = (11).\perp_1\perp_2 \longleftrightarrow A = [0,2]2^{-1} = [0,1] \quad (41)$$

$$b_{G2} = (11).\perp_1 1\perp_2 \longleftrightarrow B = [1,3]2^{-2} = [1/4,3/4] \quad (42)$$

$$c_{G(-1)} = (11)\perp_1\perp_2. \longleftrightarrow C = [0,2]2^1 = [0,4] \quad (43)$$

$$\underline{d} = 0, \quad \overline{d} = 0 \quad (44)$$

のどちらかをとるものとする．これらは0を含む区間を除数 B の初期値から除外したものである．このとき，

$$a_{Gn_a} \longleftrightarrow A = [a-1, a+1]2^{-n_a} \subset [0,1] \quad (45)$$

$$b_{Gn_b} \longleftrightarrow B = [b-1, b+1]2^{-n_b} \subset [1/4,1] \quad (46)$$

という包含関係が成り立つので，以下の不等式が成り立つ．

$$1 \leqq a \leqq 2^{n_a} - 1 \quad (47)$$

$$2^{n_b-2} + 1 \leqq b \leqq 2^{n_b} - 1 \quad (48)$$

ここで，n_a, n_b を固定したときの n_c の最大値を求める．すなわち，入力をそれぞれ n_a, n_b ビットずつ与えたときに，可能な最大の出力ビット数 n_c を求める．式 (8) より，

$$c_{Gn_c} \longleftrightarrow C \supset \frac{A}{B} \quad (49)$$

が成立する．右辺の区間 A/B の幅は

$$\frac{\overline{A}}{\underline{B}} - \frac{\underline{A}}{\overline{B}} = \left(\frac{a+1}{b-1} - \frac{a-1}{b+1}\right)2^{n_b-n_a} = \frac{2(a+b)}{b^2-1}2^{n_b-n_a} \quad (50)$$

で与えられ，これは $a=1, b=2^{n_b}-1$ のとき，最小値

$$\frac{\overline{A}}{\underline{B}} - \frac{\underline{A}}{\overline{B}} \geqq \frac{2^{n_b-n_a+1}}{2^{n_b}-2} \quad (51)$$

をとる．式 (49) より，区間 C の幅はこれより広いので，

$$\overline{C} - \underline{C} = 2^{-n_c+1} \geqq \frac{2^{n_b-n_a+1}}{2^{n_b}-2} > 2^{-n_a+1} \quad (52)$$

$$n_c < n_a \quad (53)$$

$$\therefore\ n_c - n_a \leqq -1. \quad (54)$$

次に商と余りとして，整数 $\underline{q},\overline{q}$ および $\underline{r},\overline{r}$ をそれぞれ

$$2^{n_b}(a-1) = \underline{q}(b+1) + \underline{r} \quad (0 \leqq \underline{r} < b+1) \quad (55)$$

$$2^{n_b}(a+1) = \overline{q}(b-1) - \overline{r} \quad (0 \leqq \overline{r} < b-1) \quad (56)$$

と定義する．すると (35) および (36) より，$\underline{d},\overline{d}$ は

$$\underline{d} = 2^{n_c-n_a}\left(\underline{q} + \frac{\underline{r}}{b+1}\right) - (c-1) \quad (57)$$

$$\overline{d} = (c+1) - 2^{n_c-n_a}\left(\overline{q} - \frac{\overline{r}}{b-1}\right) \quad (58)$$

で与えられる．ここで (54) と，(55), (56) における $\underline{r},\overline{r}$ に対する条件から，

$$0 \leqq 2^{n_c-n_a}\frac{\underline{r}}{b+1} < \frac{1}{2}, \quad 0 \leqq 2^{n_c-n_a}\frac{\overline{r}}{b-1} < \frac{1}{2} \quad (59)$$

が成り立つ．したがって，表4の3条件の判定に際して，(57), (58) における $\underline{r},\overline{r}$ の項を無視してよい．

そこで以下では，$\underline{d},\overline{d}$ の更新規則の代わりに，$\underline{q},\overline{q},\underline{r},\overline{r}$ の更新規則を与える．これらは2段階で更新される．まず入力 a_G あるいは b_G の精度が向上したら，それぞれ表8あるいは表9に従って $\underline{q}^*,\overline{q}^*,\underline{r}^*,\overline{r}^*$ を計算する．続いて，$\underline{q},\overline{q},\underline{r},\overline{r}$ を

$$\underline{q} := \underline{q}^* + Q(\underline{r}^*, b+1) \quad (60)$$

$$\underline{r} := R(\underline{r}^*, b+1) \quad (61)$$

$$\overline{q} := \overline{q}^* - Q(\overline{r}^*, b-1) \quad (62)$$

$$\overline{r} := R(\overline{r}^*, b-1) \quad (63)$$

によって更新する．ただし，$b_{\mathrm{G}n_b}$ が更新された場合には，b として更新後の値

$$b_{\mathrm{G}n_b+1} \longleftrightarrow B = [b-1, b+1]2^{-n_b-1} \quad (64)$$

を用いる．また，$Q(n,m), R(n,m)$ はそれぞれ

$$Q(n,m) = \left\lfloor \frac{n}{m} \right\rfloor \quad (65)$$

$$R(n,m) = n - Q(n,m)m \quad (66)$$

で定義される商と余りを与える関数である．

上述の更新規則を具体例によって説明する．入力 $a_{\mathrm{G}n_a}$ が $\perp_1 := P_a$ と更新されたとする．更新後の値に $'$ をつけて

$$n_a' = n_a + 1 \quad (67)$$

$$a' = 2(a - 1/2) \quad (68)$$

のように表わすことにすると，(55) より

$$2^{n_b}(a'-1) = 2^{n_b}(2(a-1/2)-1) = 2\cdot 2^{n_b}(a-1)$$
$$= 2\underline{q}(b+1) + 2\underline{r} \quad (69)$$

を得る．表 8 の $\perp_1 := P_a$ 行における $\underline{q}^*, \underline{r}^*$ の定義はこれにもとづく．この後，(55) の余りに関する条件を満たすように (60), (61) によって $\underline{q}, \underline{r}$ を更新し，$\underline{q}', \underline{r}'$ とすると，

$$2^{n_b}(a'-1) = \underline{q}'(b+1) + \underline{r}' \quad (0 \leqq \underline{r}' < b+1) \quad (70)$$

となる．これは (55) と同じ形をしている．同様にして，表 8, 9 の他の定義も得られる．

次に，符号付きグレイコードの除算を考える．被除数 a_{G} に対応する区間 A が 0 を含み，0 に関して対称のときは，出力区間 C も 0 に関して対称となる．ただし，除数 b_{G} が 0 を含む $b_{\mathrm{GS}} = \perp_1$ の場合は何も出力できない．このことより，出力 c_{G} の符号ビット c_{GS} は表 10 に従って求め，仮数ビットに対しては上述の非負の除算アルゴリズムを適用する．

4. まとめ

グレイコードを用いた上位ビットからのビットシリアルな演算アルゴリズムとして，加減算・乗算・除算のアルゴリズムを提案した．上位ビットからの演算では，入力が上位ビットから順に与えられると，それに応じて出力も上位ビットから順に得られる．これを任意に繰り返すことで任意精度演算を行える．特に，本アルゴリズムで出力されるグレイコードは，与えられた入力に対して達成可能な最大精度のものである．

謝辞:この研究の一部は 21 世紀 COE プログラム「電気電子基盤技術の研究拠点形成」の補助を受けた．

表 8 $\underline{q}^*, \overline{q}^*, \underline{r}^*, \overline{r}^*$ の定義 ($a_{\mathrm{G}n_a}$ が更新された場合)

$a_{\mathrm{G}n_a} \to a_{\mathrm{G}n_a+1}$	$\underline{q}, \underline{r}$	$\overline{q}, \overline{r}$
$\perp_1 := P_a$	$\underline{q}^* := 2\underline{q}$	$\overline{q}^* := 2\overline{q}$
	$\underline{r}^* := 2\underline{r}$	$\overline{r}^* := 2\overline{r} + 2^{n_b+1}$
$\perp_1 := \overline{P_a}$	$\underline{q}^* := 2\underline{q}$	$\overline{q}^* := 2\overline{q}$
	$\underline{r}^* := 2\underline{r} + 2^{n_b+1}$	$\overline{r}^* := 2\overline{r}$
$\perp_2 := 1$ ($\perp_1$ の直後のとき), 0	$\underline{q}^* := 2\underline{q}$	$\overline{q}^* := 2\overline{q}$
	$\underline{r}^* := 2\underline{r} + 2^{n_b}$	$\overline{r}^* := 2\overline{r} + 2^{n_b}$

表 9 $\underline{q}^*, \overline{q}^*, \underline{r}^*, \overline{r}^*$ の定義 ($b_{\mathrm{G}n_b}$ が更新された場合)

$b_{\mathrm{G}n_b} \to b_{\mathrm{G}n_b+1}$	$\underline{q}, \underline{r}$	$\overline{q}, \overline{r}$
$\perp_1 := P_b$	$\underline{q}^* := \underline{q}$	$\overline{q}^* := \overline{q}$
	$\underline{r}^* := 2\underline{r} + 2\underline{q}$	$\overline{r}^* := 2\overline{r}$
$\perp_1 := \overline{P_b}$	$\underline{q}^* := \underline{q}$	$\overline{q}^* := \overline{q}$
	$\underline{r}^* := 2\underline{r}$	$\overline{r}^* := 2\overline{r} + 2\overline{q}$
$\perp_2 := 1$ ($\perp_1$ の直後のとき), 0	$\underline{q}^* := \underline{q}$	$\overline{q}^* := \overline{q}$
	$\underline{r}^* := 2\underline{r} + \underline{q}$	$\overline{r}^* := 2\overline{r} + \overline{q}$

表 10 除算における出力の符号ビット c_{GS}

$a_{\mathrm{GS}} \backslash b_{\mathrm{GS}}$	0	$\perp_1$	1
0	1	–	0
$\perp_1$	$\perp_1$	–	$\perp_1$
1	0	–	1

文　献

[1] F. Gray, "Pulse code communication," U. S. Patent 2 632 058, March 17 1953.

[2] P.Horowitz and W.Hill, *The Art of Electronics*, Cambridge University Press, second edition, 1989.

[3] A.K.Dewdney, *The New Turing Omnibus*, Computer Science Press, 1993.

[4] E. Gilbert, "Gray codes and paths on the n-cube," *Bell System Technical Journal*, vol. 37, pp. 815–826, 1958.

[5] D.J.Amalraj, N.Sundararajan, and G.Dhar, "A data structure based on Gray code encoding for graphics and image processing," *SPIE Applications of Digital Image Processing XIII*, vol. 1394, 1990.

[6] F. G. Heath, "Origins of the binary code," *Scientific American*, vol. 227, no. 2, pp. 76–83, 1972.

[7] W.H.Press et al., *Numerical Recipes in C*, Cambridge University Press, second edition, 1992.

[8] H.Tsuiki, "Real number computation through Gray code embedding," *Theoretical Computer Science*, vol. 284/2, pp. 467–485, 2002.

[9] KS.Trivedi and M.D. Ercegovac, "On-line algorithms for division and multiplication," *IEEE Transactions on Computers*, vol. C-26, no. 7, pp. 681–687, 1977.

[10] M.D.Ercegovac, "A general hardware-oriented method for evaluation of functions and computations in a digital computer," *IEEE Transactions on Computers*, vol. C-26, no. 7, pp. 667–680, 1977.

[11] M.D.Ercegovac, "On-line arithmetic: An overview," *SPIE Real Time Signal Processing VII*, vol. 495, pp. 86–93, 1984.

[12] M.D.Ercegovac and T.Lang, "On-line arithmetic: a design methodology and applications in digital signal processing," *VLSI Signal Processing III*, pp. 252–263, 1988.

[13] G.Alefeld and J.Herzberger, *Introduction to Interval Computations*, Academic Press, New York, 1983.