続

実設計に応用できる演算回路スキルを身につけよう

デバイスの記事

外村元伸

第5回 除算・開平・逆数・逆開平数とその演算回路設計(2)ディジット選択規則を中心に

除算・開平・逆数・逆開平数の演算器は，部分剰余を求める漸化式を利用することによって，共通に設計できます．そして，商，開平，逆数，逆開平ディジットは，これら共通のディジット選択規則を導くことによって求められます．ところが，ディジット選択規則を実際に導出することは，特に部分剰余演算にキャリ・セーブ・アダーを使う場合には結構難しく，やっかいです．そのためディジット選択規則にまつわることで，あまり知られていないことがあります．今回は，このあたりのことについて詳しく解説します．（筆者）

ディジット選択規則の導出

● 伝統的に使われている漸化式のふしぎ

1ビットずつディジットを求めていく基数2の除算において，第iステップ目の漸化式は，

$$R_{i+1} \leftarrow 2\left(R_i - q_i \cdot X\right),\ R_0 = Y \quad \cdots\cdots(1)$$

であることを前回（2007年5月号，pp141-146の連載第4回）説明しましたが，実は多くの本や論文では，この式は使われていません．その代わりに，

$$R_{i+1} \leftarrow 2R_i - q_{i+1} \cdot X,\ \ R_0 = Y \quad \cdots\cdots(2)$$

が伝統的に使われています．これは，なぜなのでしょうか．開平では，部分開平値 $Q_i = \sum_{k=0}^{i} q_k \cdot 2^{-k}$ を採用していますが，筆者は除算では，部分商 $Q_i = \sum_{k=0}^{i-1} q_k \cdot 2^{-k}$ を採用しています．総和の最大数が，iか$i-1$の違いだけですが，このちょっとしたテクニックによって，除算・開平・逆数・逆開平数の第iステップ目の漸化式が式(3)のように，同じような形式になるからです．

$$\begin{cases} \text{除算：}\ R_{i+1} \leftarrow 2\left(R_i - q_i \cdot X\right),\ R_0 = Y \\ \text{開平：}\ R_{i+1} \leftarrow 2\left(R_i - q_{i+1} \cdot \left(Q_i + q_{i+1} \cdot 2^{-(i+2)}\right)\right), \\ \qquad Q_{i+1} = Q_i + q_{i+1} \cdot 2^{-(i+1)},\ R_0 = X,\ Q_0 = 0 \\ \text{逆数：}\ R_{i+1} \leftarrow 2\left(R_i - q_i \cdot X\right),\ R_0 = 1 \\ \text{逆開平数：} R_{i+1} \leftarrow 2\left(R_i - q_{i+1} \cdot X\left(Q_i + q_{i+1} \cdot 2^{-(i+2)}\right)\right) \\ \qquad Q_{i+1} = Q_i + q_{i+1} \cdot 2^{-(i+1)},\ R_0 = 1 - X,\ Q_0 = 1 \end{cases} \cdots(3)$$

では，開平と同じように$Q_i = \sum_{k=0}^{i} q_k \cdot 2^{-k}$を除算でも採用してみましょう．その場合の漸化式を導くと，

$$\begin{aligned} R_{i+1} &= 2^{i+1}\left(Y - Q_{i+1}X\right) \\ &= 2^{i+1}\left(Y - Q_iX - q_{i+1} \cdot 2^{-(i+1)}X\right) \\ &= 2R_i - q_{i+1}X \end{aligned} \quad \cdots\cdots(4)$$

となり，確かに式(2)が導かれます．この違いを意味的に考えてみます．式(1)は，現在の部分剰余R_iを用いて，商ディジットq_iを決定します．それにもとづいて部分剰余を演算し，2倍して次の部分剰余R_{i+1}とします．それに対して式(2)は，現在の部分剰余R_iを2倍してから，つまり，$2R_i$を用いて，商ディジットq_{i+1}を決定します．商ディジットq_{i+1}にもとづいて部分剰余を演算し，そのまま次の部分剰余R_{i+1}とします．ここでは，q_iのiの値は相対的に一つずれていますが，大した意味をもちません．決定的な違いは，R_iと$2R_i$で，商ディジットの選択のときに効いてきま

KeyWord 除算，開平，逆数，逆開平数，部分剰余，ディジット選択規則，漸化式，キャリ・セーブ・アダー，符号付きディジット，ディジット・アダー，SSDアダー，Simplified Signed-digit Adder

表1　ディジット選択規則

q_i	r_{-1}	r_0	r_1	(s_2, c_2)
＋1	0	1	1	＊
＋1	0	1	0	＊
＋1	0	0	1	＊
＋1	0	0	0	Not 0
0	0	0	0	0
0	1	1	1	＊
－1	1	1	0	＊
－1	1	0	1	＊
未定義	1	0	0	＊

表2　キャリ・セーブ・アダーの演算規則とその特徴をとらえる表現

＊={1, 2} ＋={0, 1}	加　数			
	11	10	01	00
被加数 11	1＋	0＊	2＋	1＊
被加数 10	0＊	0＋	1＊	1＋
被加数 01	2＋	1＊	1＋	0＊
被加数 00	1＊	1＋	0＊	0＋

(**a**) キャリ・セーブ・アダーの初期状態からの演算結果

＊={1, 2} ＋={0, 1}	加　数			
	11	10	01	00
被加数 22	2＊	2＋	1＊	1＋
被加数 21	2＋	1＊	1＋	0＊
被加数 20	1＊	1＋	0＊	0＋
被加数 12	1＊	1＋	2＊	2＋
被加数 11	1＋	0＊	2＋	1＊
被加数 10	0＊	0＋	1＊	1＋
被加数 02	2＊	2＋	1＊	1＋
被加数 01	2＋	1＊	1＋	0＊
被加数 00	1＊	1＋	0＊	0＋

(**b**) キャリ・セーブ・アダーの完全な演算結果

す．例えば式(2)の場合，$Y > X$のときは，$R_0 = Y = [0.11]$ b，$X = [0.10]$ bの例では，$2R_0 = (r_{-1}, r_0, r_1) = (0, 1, 1) = [01.1]$ bなので，**表1**のディジット選択規則から，$q_1 = 1$です．

$$R_1 = [01.0]\text{b} \leftarrow 2Y - q_1 \cdot X = [01.1]\text{b} - [0.10]\text{b} = [01.0]\text{b}$$
$$2R_1 = (r_{-1}, r_0, r_1) = (1, 0, 0) = [10.0]\text{b} \quad \cdots(5)$$

から，ディジット選択規則の定義域の範囲を超えてしまい($R_1 > 0$なのに，$2R_1 < 0$となる)，部分剰余演算を続行できません．このような状態を避けるには，$Y > X$のときは，Yの値を1ビット右にシフトして調節します($Y/2 < X$)．しかし，もし，部分剰余R_iの演算の途中で，$R_i = [01.1]$ bとなっても同じようなことが生じてしまいます．後で詳しく説明しますが，このような状態にならないように，式(2)を採用しながらも松原氏ら[3][4]はディジット選択規則(**表1**)を提案しました．

では，式(1)を採用したらどうなるのでしょうか．$Y > X$のときに，Yの値を1ビット右にシフトする調節は不要です．はじめから1ビット左シフト($2R_i$)という余分なことはしていないからです．筆者は，偶然にも，除算・開平において数少ない式(1)を採用したGosling[1][2]らの論文を参照し，検討していたので，後に式(2)の奇妙なことに気づくようになりました．式(2)を採用している本や論文の中では，$X > Y$の場合だけを想定して説明しているので，式(2)にあるこのような「ちょっとしたこと」が見過ごされています．

● キャリ・セーブ・アダーの演算結果をとらえる表現

表1のディジット選択規則の正しさや，もっと簡単な規則がないかを厳密に導いたり，証明するのは，実際にはなかなか難しい作業になります．そこで，部分剰余演算に用いるキャリ・セーブ・アダーの演算規則を記述したり，その動作を解析するために，**表2**に示す表現方式を考案しました．

2けたのキャリ・セーブ加算について考えます．キャリ・セーブ形式の各けたは，現在けたのサムsと下位けたからのキャリ入力cについて，$s + c = (s, c)$で数表現します．$2 = (1, 1)$，$1 = (0, 1) = (1, 0)$，$0 = (0, 0)$です．加数部を行，被加数部を列で表し，演算結果を行と列が交差する部分で与えます．上位側けたの演算結果は，下位側けたからのキャリ入力により確定します．一方，下位側けたは，下位側けたへのキャリ入力があるときとないときがあるので，演算結果を記号＊={1, 2}または，記号＋={0, 1}で表現します．キャリ・セーブ・アダーの初期入力状態では，{0, 1}しかありませんので，**表2**(**a**)に記述しました．それから，**表2**(**b**)に記述したように，部分剰余演算では，ディジット選択値によって決まる加数部への各けた入力値は{0, 1}で，部分剰余の被加数部の各けた入力値だけ{0, 1, 2}のキャリ・セーブ形式で扱うことに気をつけてください．

それでは，これからキャリ・セーブ・アダーを用いる場合のディジット選択規則を導出していきます．形式ばった説明をしていくと分かりにくいので，まず，既に与えている**表1**のディジット選択規則について，演算長6ビットの場合の例1(**表3**)を用いて，正しく動作するのかを検証しながら説明していきます．

● 6ビット長の除算でディジット選択規則を検証してみる

表3の見方は，列に現在の部分剰余値R_iを，行に除数Xの$-q_i$倍である加数を与えて，行と列の交差する部分に左1ビット・シフトして，捨て去る最上位けたと続く上位3けた分のキャリ伝播が表現できるように結果を与えています．