3E－4

十進数による直接的な数値演算

佐々木 真，大須賀 勝美，黒川 一夫

東京理科大学

１．はじめに

現在、電子計算機における数値演算は一般に二進数によって実行されている。しかし、人間が扱う数値はほとんどの場合十進数であり、その入出力、あるいは有効数字の桁数の管理においては十進数が便利である。

本研究では十進数のままでの直接的な演算の実行を考えており、演算方法としても数値的な演算でなく、演算テーブルの参照による方法を提案する。取り扱う演算としては、浮動小数点表示による四則演算の実行について検討を行った。

２．演算方法

2.1 数値の取り扱い

十進数による数値の扱い方として、十進１桁の値を４ビットで表現するＢＣＤ表現を用いる。浮動小数点表現としてここでは図１のようなものを考える。

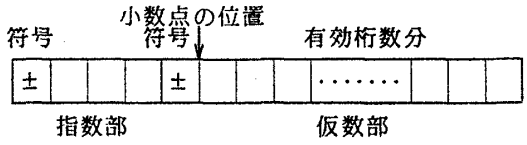

図１　数値データの書式

図１の各桁は４ビットで表現され、データの形式として固定長と可変長の２つを考える。固定長の有効桁数はモードにより７，11，15桁に設定可能とし、可変長データの場合は、数値データの前に有効桁数を記述し、仮数部はの有効桁数だけ数字の並べる。符号の部分では演算結果のフラグも表し、４ビットのそれぞれで、正負のマイナス・フラグ、数値零を特別に扱うためのゼロ・フラグ、演算のオーバー・フロー、アンダー・フローを表すキャリー・フラグとボロー・フラグの４つのフラグを用意する。

2.2 演算装置

演算装置として図２のようなものを考える。入力としては４ビットの引数が２つ、演算種類の指定が２ビット、桁上げ及び桁下げのフラグが１ビット、引き数の入れ換え制御に１ビットの合計１２ビット。出力として演算結果が８ビットで、加減算の場合は下位４ビットが結果で上位４ビットの内１本を桁上げ、桁下げ用のフラグに用いる。乗算の場合は、２つの引き数の乗算結果の上位と下位の桁がそれぞれ４ビットで出力される。除算の場合商が上位４ビットに剰余が下位４ビットに現れるように、テーブルを作成しておく。

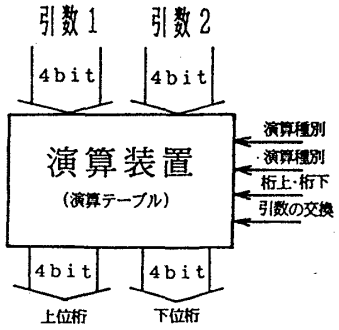

図２　演算装置

2.3 演算装置の接続

前述の演算装置１つを用いて、１桁演算を繰り返して有効桁数分の演算を行うこともできるが、ここでは有効桁数分だけ演算装置を並列接続させ、１回の動作で演算を実行することを考える。しかし、桁上がりや桁下がりが生じるので、演算終了時に全桁のフラグがなくなるまでフラグを伝搬させる必要がある。

2.4 演算方法

(1) 加算

加算の場合図３のように演算装置を接続し、下の桁から順番にテーブルを参照し、各桁の値を決定していく。次の桁を演算する場合は前回の桁上がりフラグにより参照テーブルを切り換え、桁上げ処理を行う。

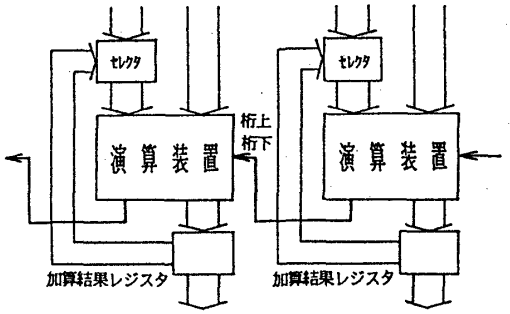

図３　加算演算の動作

(2) 減算

減算の場合も接続と動作は加算と同じであるが、２つの引き数の大小判別を事前に行い、大きいものから小さいものを減ずる。このとき引き数の入れ換えをするのではなくテーブルを２つ用意しておく。

Direct Arithmetic Method by Decimal Digit

Makoto SASAKI, Katsumi OSUGA, Kazuo KUROKAWA

Science University of Tokyo

(3) 乗算

乗算演算の場合は乗算テーブルの参照と加算テーブルの参照を、図４、図５のように組み合わせて、乗算と加算を繰り返すことにより演算を実行させる。

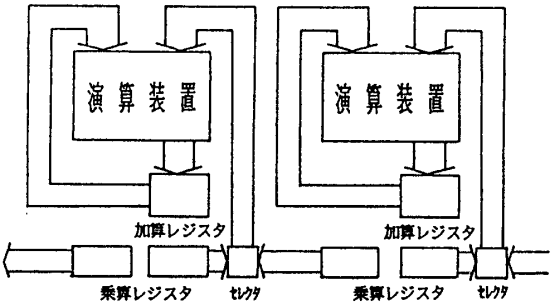

図４　乗算演算のテーブル参照動作

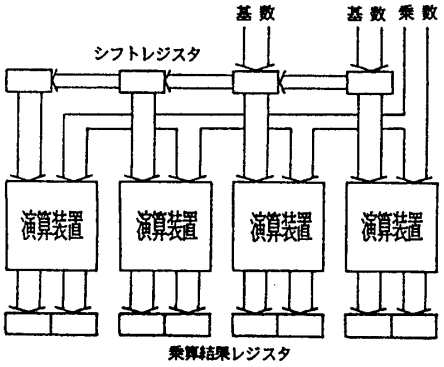

図５　乗算演算の加算動作

(4) 除算

除算の場合、特に帰除法という手法を取り入れた。この方法は商と剰余をテーブルとして用意し、それと乗算と減算を組み合わせて結果を求めるもので、一般に用いられている除算方式とは異なる。

2.5 パイプライン処理

連続的に加算または減算を行う場合、各演算ごとにフラグ伝搬処理を行う必要はなく、前回のフラグ結果により次に参照するテーブルを切り換える。そして、最後にだけフラグの伝搬処理を行うことにより高速化が図れ、連続的な演算中は１動作で演算が実行されることになる。特に乗算の場合、演算結果の上位の桁と下位の桁では数値に特徴があり、上位の桁のテーブル参照を行った後の加算では桁上がりが起こりにくいので、乗算結果を加算する順番を変えることによりフラグの伝搬が早く終了し、より高速化が図れる。

４．各演算の評価

実際にハードウェアは作成されていないので、コンピュータ上でのシミュレーションにより動作の確認と評価を行う。

並列接続した場合に、演算にかかる動作数をシミュレーションにより調べる。乱数により数値を発生させ演算を行いテーブルの参照された回数を数えて動作数とし、１万回の演算での平均値を求めた。

(1) 並列接続におけるフラグの伝搬

並列接続による加算演算における、有効桁数に対する、フラグの伝搬処理に必要な動作の平均回数を図６に示す。減算の場合もまったく同じ値になる。

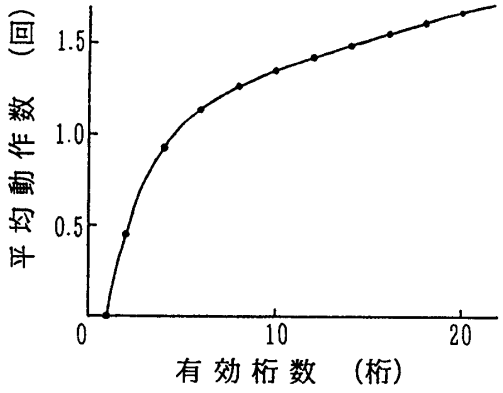

図６　加算におけるフラグ伝搬

(2) パイプライン処理による効率化

乗算において乗算テーブルの参照と加算を行う順番を変えた場合のフラグの平均伝搬回数を図５に示す。Ｈ、Ｌは加算の順序を表し、Ｈは上位桁、Ｌは下位桁の加算を表す。

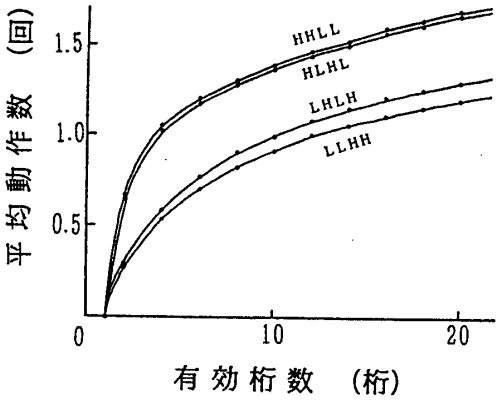

図７　乗算におけるフラグ伝搬

５．まとめ

演算テーブルを１６×１６の大きさで用意することにより、１６進数の演算も全く同様に実行可能なので、モード切り換えなどで演算を選択することもできる。また、テーブルサイズが大きくできれば、７ビットで十進数２桁、１０ビットで十進数３桁のように桁数の拡張を行って、効率化を図ることもできる。

除算についても帰除法を用いたことにより、並列的に演算装置を用いて除算した場合に、従来の除算方式よりも効率的に演算できるものと考えられる。

参考文献

Kai Hwang : “Computer Arithmetic”, 1979, John Wiley & Sons, Inc.
（堀越 彌 訳：“コンピュータの高速演算方式”, 1980, 近代科学社.）