第3章

さまざまな演算の実装

さまざまな数式で示されたディジタル信号処理のアルゴリズムは，最終的に商品化によって初めて実際に生かされます．途中のアルゴリズムの検証では，MATLABなどのシミュレーション・ツールを使い，数式のまま設計が可能です．しかし，最後にそのアルゴリズムを，DSPを使ったりFPGAを使って，具体的にハードウェアに実装する必要があります．DSPやFPGAなどは，数式をそのまま組み込むことはできません．

DSPの場合は基本的に，高速の掛け算と足し算器しか備わっていません．またFPGAでは，最近は掛け算器が入ったものもありますが，一般的には汎用ゲートから作りあげなければなりません．ここでディジタル信号処理の実装に慣れていない人は，シミュレーションの結果を前に途方にくれることになるかもしれません．

まず頭を切り替えないといけないのは，数式で示されたシミュレーションと違って扱う数が十進数ではなく，二進数であることです．DSPのソフトウェアではコンパイラやアセンブラが，十進で記述しても自動的に変換してくれるので，少しは楽かもしれません．FPGAなどのハードウェアの場合は，完全に1と0のロジック回路で組みますから，つねに二進数を意識する必要があります．

それから実装で問題になるのが，リアルタイム処理による処理時間の制限です．シミュレーションではリアルタイム処理ではありませんから，式の中に出てくるたとえば三角関数を理想的なアルゴリズムでいくら時間をかけて計算しても問題ありません．ところが組み込みディジタル信号処理では，入力される信号が**図3-1**のように一定時間でサンプリングされた信号です．計算に時間がかかっているうちに，次の新しいデータが入力され，処理が全然間に合わなくなることがおきます．

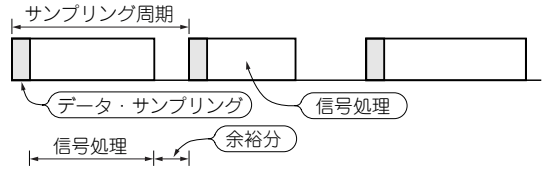

図3-1　信号処理の時間（リアルタイム処理）

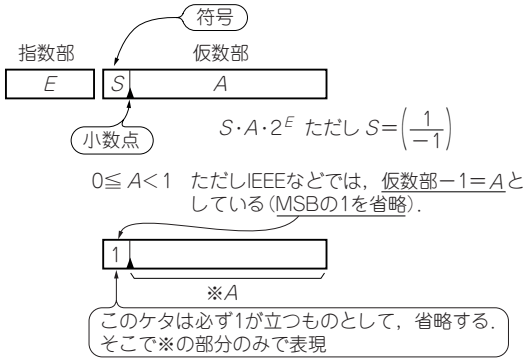

図3-2
浮動小数点

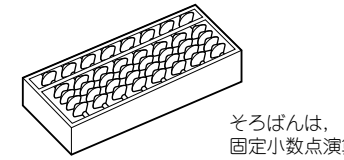

図3-3　固定小数点演算？

そのため，計算の対象となる関数の近似とか，すでに計算してある値を参照するなどして，計算速度を速くする工夫が必要です．とくにハードウェアで実装する場合は，掛け算器などの演算回路は回路面積を食うので，大胆な発想での近似の考え方や，別のアルゴリズムを用いたアプローチなども必要になってきます．

3-1　浮動小数点演算

皆さんはすでにご存じだと思いますが，電卓やコンピュータでの数値計算では一般的に浮動小数点を使います．すなわち**図3-2**のように，小数点の位置が一定の仮数と，それと掛け合わせる指数部です．たとえば十進数でいえば，

1.23456E3　→　1.23456が仮数　　E3 = 10^3が指数
　　→　1234.56と等しい

となります．浮動小数点を使えば，大きな数と小さな数を同じ桁数の精度で扱うことが可能で，とくにディジタル信号処理で広いダイナミック・レンジをもつ信号を精度良く計算するのに向いています．

世の中，浮動小数点の計算があたりまえだと考えられるかもしれませんが，そうでもありません．これと反対に固定小数点の計算方法があります．一番身近なのは**図3-3**のようなそろばんです．そろばんでは計算するときに桁取りが一定です．言い換えれば，小数点をそろばんのどこかに頭で設定して計算します．計算の精度はそろばんの幅で決まり，浮動小数点のような大きなダイナミック・レンジの数値を扱うことはできません．**図3-4**では異なる二つの数を8ビットの固定小数点で表現した場合です．このように狭いダイナミック・レンジのため，入力の数値によっては精度の良い計算をするのが難しくなります．

先ほどの浮動小数点の計算では十進数で示しましたが，これを実際にディジタル信号処理で使う二進数で表すとどうなるでしょうか？

1.23456E3　=　約　158024/128
　　二進数　→　$100110100101001000 \times 2^{-7}$
　　$= 1.00110100101001000 \times 2^{10}$

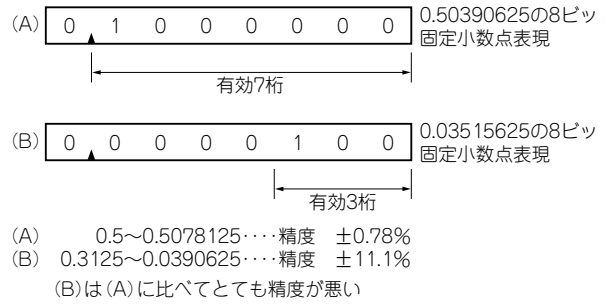

図3-4　固定小数点演算の精度

二進数だと，慣れていないためか数の大きさが直感としてわかりませんね．ここで「約」としたところに問題があります．十進数で無理数でないぴったりの数でも，浮動小数点に変換するときに無理数になる場合や，規定の幅の桁数でぴったり表現できない場合があります．すなわち，十進数を二進数の浮動小数点に変えるだけで誤差が生まれます．

そのため，コンピュータの計算では，とくにお金を扱う銀行などでは，「約」とか誤差を嫌い，二進数に変換しないで十進数のまま計算する場合もあります．すなわち十進数の1桁は0～9の10の状態があるため，十進数の1桁に4ビットを割り当てれば扱うことが可能です．しかし，ディジタル信号処理では入ってくる計算対象の信号はA-D変換器など誤差やノイズを含んだ信号が主です．充分な桁があれば，変換の際の誤差はほとんど問題ないと考えられます．それよりも十進数のまま扱うことによる，処理の複雑さや回路規模の拡大のほうが問題です．汎用CPUには十進数を計算するのに役立つ命令が準備されている場合もありますが，DSPには十進数を計算する命令やハードウェアが内蔵されているのを見たことがありません．

DSPは，**図3-12**のような積和演算を高速に実行できるハードウェアを内蔵しているのが特徴の一つです．もちろん二進数ですが，浮動小数点と固定小数点のDSPの2種類に分類されます．浮動小数点DSPは一般的に高価ですが，計算アルゴリズムを考えるのがめんどうでないために開発期間を短縮できる可能性があります．また，誤差の検証をあまり細かく行う必要がないメリットは大きいものです．経験が浅い人でも，すぐに使いこなせるようになるでしょう．

今日，DSPや汎用CPUに搭載されている浮動小数点演算ユニットは，**図3-5**のようなIEEEが定めた規格を採用しているのが多くあります．この図は単精度ですが，倍精度も規定されています．通常のディジタル信号処理であれば，単精度で充分なところがほとんどでしょう．IEEE規格の特徴は，「けち」表現を使っているところです．仮数Aの範囲が$1 \leqq A < 2$のところを，MSBの1が必ず1になることからそれを省略し，$0 \leqq (A-1) < 1$で表現しています．

3-2　固定小数点演算

やはり実際の商品化のときには，コストや消費電力が問題になります．その意味で本当は浮動小