FM 音源の基礎と実装

執筆：JA1TYE

1. はじめに

みなさんこんにちは，JA1TYE です．昨年の部誌にはカードサイズのシングルボードコンピュータ"Raspberry Pi"について書きましたが, 今回も懲りずに組み込み系な話題です. 今回は FM 音源とそのマイコンへの実装について書いていきたいと思います.

2. FM 音源とは

簡単な「FM音源」の説明としては「ある信号の周波数を別の信号で変化させる(周波数変調をする)ことによって様々な音色を生成する方式」というような説明が一般的かと思います．確かに，「FM音源」の"FM"は"Frequency Modulation(周波数変調)"の略ですから, おおよそこの説明は正しいといえます.

しかし，後で述べるように，一般的に「FM音源」と呼ばれている音源は周波数変調ではなく，位相変調(PM: Phase Modulation)によって複雑な音色を生成しています．しかし，本稿ではわかりやすさを優先し，一般的な呼称である「FM音源」を以下で用いることとします.

さて，よく分からない技術的な解説は後に回すとして, まずはFM音源が登場した背景を紐解いていきましょう.

3. シンセサイザとその音源方式

1960年代にロバート・モーグが開発した初期のミュージック・シンセサイザ(以下では単に「シンセサイザ」とします)は，アナログ電子回路によって鋸歯状波（のこぎり波）等の波形を発生させ，この波形の振幅や周波数特性を変えることによって様々な音色を作り出していました．おおまかに，このようなシンセサイザの構成要素は，制御電圧に応じた周波数をの信号を生成するVCO(Voltage Controlled Oscillator)，信号の周波数特性(高音を落とす，強調するといったような特性)を制御電圧に応じて変化させるVCF(Voltage Controlled Filter)，出力される信号の振幅(音量)を制御電圧に応じて変化させるVCA(Voltage Controlled Amplifier)と，VCFやVCAのための(一般的には，鍵盤を押してからの経過時間により変化する)制御信号を生成するEG(Envelope Generator)に分けることができます．これらのブロックは一般的には図1に示すように接続されます.

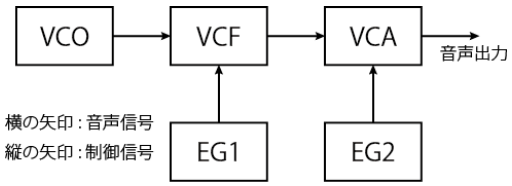


図1　初期のシンセサイザの基本構成

このような初期のミュージック・シンセサイザが採用していた音色生成方式では，基本的にはVCOによって生成された波形が持つ成分をVCFによって削っていくことによって，音色の主な部分を決めていました. そのため, このような音色生成方式を「減算合成」と呼びます.

減算合成方式のシンセサイザは，1970年代までの主流でした．しかし，アナログ電子回路によるシンセサイザは部品点数の多さ（とそれに伴う価格上昇と重量・体積増加）や，一部の回路の温度特性の悪さ(特にVCOでは温度が変化すると音程が外れてしまうという問題を生じた)といった問題点を抱えていました．これらの問題のうち，部品点数の多さについては中心部品のIC化，温度特性の悪さ

についてはVCOの部分的なデジタル制御化や，恒温槽内蔵ICの導入といった対策が取られていきましたが，どれも完璧な解決策とは言えませんでした．

これらの問題に対して，1980年代ごろから各社は「そもそもアナログ電子回路を使わない(減らす)」という手に出るようになりました．メモリに音の波形を最初から最後まで記録し，その再生速度を変化させることで音程を変化させ，楽器として演奏できるようにするというもの(メモリの価格が安くなった現在ではこの方式が主流になっています)や，VCOをデジタル制御とするのではなく，そもそもデジタル的に波形を生成してしまい，音程が外れてしまうという問題を解決しつつ，波形生成部分の回路をシンプルにしたものが現れました．そんな中，1983年に現れたのがFM音源を採用したYAMAHAのDX7です．

DX7は専用ハードウェアによるデジタル演算によってFM音源を実装したICが搭載され，パラメータ等のセッティングもボタンと7セグメントLEDの組み合わせを中心にして行うという，ほぼすべてがデジタル化されたシンセサイザでした．このデジタル化の恩恵により，当時としては破格の約25万円を実現しながらも，16音同時発音というこちらも当時としては破格のスペックを誇っていました．

しかし，何よりもDX7がヒットした要因はその音色の幅広さでしょう．FM音源は減算合成と異なり，元々の波形よりも複雑な波形を合成することができます．そのため，従来の減算合成では難しかったベル系の音や，ピアノ系，特にエレクトリック・ピアノの音などがよりリアルに再現できました．金属的な響きの音の再現には非整数次倍音，すなわち基本となる音の周波数から見て，その整数倍の周波数ではない周波数の成分を含むことが非常に重要です．減算合成で用いられる鋸歯状波や矩形波，三角波といった波形は整数次倍音(基準の周波数の成分から見て整数倍の周波数の成分)のみを含んでいます．ここから減算を行っても当然非整数次倍音は生じません．しかし，FM音源ではそれを生成することができるのです．

4. FM 音源の構成

それでは実際にFM音源がどのような構成となっているか見ていきましょう．FM音源では，減算合成のシンセサイザとは異なり，波形を生成する機能ブロックである「オペレータ」を中心に音色を生成していきます．オペレータは図2のように1周期分の正弦波形を格納する波形メモリと，その波形のどこを読み出すかを指定するためのカウンタ，そしてカウンタの増加速度を決定するレジスタ，EGからの制御信号に応じて信号の振幅を制御するための乗算器，変調のための加算器からなります．

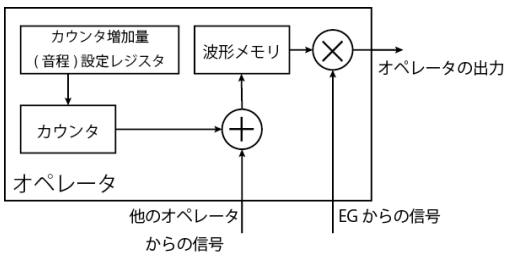
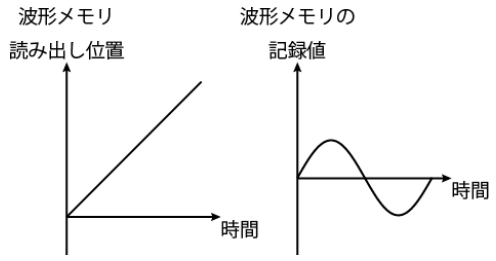


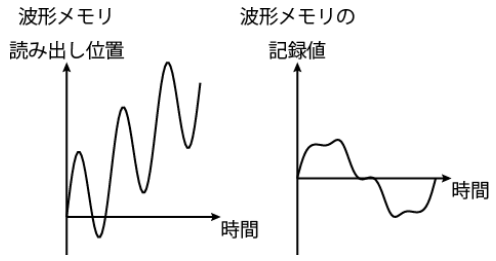
図2 オペレータの構成

さて，このオペレータですが，図2に示すカウンタ増加量設定レジスタの値をカウンタに順次加算していくだけでは，図3(a)のように波形メモリ読み出し位置は時間とともに線形に変化していくだけであり，波形メモリに保持されている波形そのものしか出力することができません．(この部分はDDS: Direct Digital Synthesis と呼ばれる技術そのものでありま

すが)これでは様々な音を生成することは不可能です．そこで，FM音源ではオペレータの波形メモリの読み出し位置を，図3(b)のようにカウンタの値に他のオペレータの出力を加えた値により決定するようにしました．



(a)　変調をかけない場合



(b)　変調をかけた場合

図3 オペレータへの変調

この時，最終的な出力として得られる波形を生成する方のオペレータを「キャリア」，キャリアの波形読み出し位置を変化させるために使われるオペレータを「モジュレータ」と呼びます．このように，キャリアの波形の読み出し位置(位相)をモジュレータの波形によって変化させるので，実際にはFM音源はFMではなくPMなのです．

さて，このようにキャリアの波形読み出し位置をモジュレータの波形によって変化させると何が起こるのかというと，キャリアよりもモジュレータの周波数が非常に低いうちはビブラートがかかったような(本来のビブラートはFMですから，あくまで「ような」です)音となります．この周波数の関係が逆転していくと，ビブラートのような震えている感じではなくなり，独特の「ギョーン」といった音が得られます．この時，キャリアとモジュレータの周波数の比を整数とならないように調整すると，非整数次倍音が発生し，金属的な音や，音程を持たない音を生成することができます．すなわち，キャリアとモジュレータの周波数比によって，音色を変えることができるのです．

この他に，各オペレータのEGによって，モジュレータがキャリアに及ぼす影響の大きさを制御したり，音量を制御することができます．これによって，打鍵からの時間に応じて変化する音色を実現することができます．

さらに実際のFM音源では，このオペレータを2個以上使用することができます．DX7は6個まで使用可能です．この6個のオペレータをどのように接続するか，という組み合わせのことを「アルゴリズム」と呼びます．6個のうち5個をモジュレータに使うのではなく，例えば打鍵時の衝撃音の再現にキャリアとモジュレータを1つずつ割り当て，その後の主な音の部分に残り4つのオペレータを配分するといったことや，オルガンのような正弦波を重ねて作るような音色では6つのオペレータをすべてキャリアとして用いるといった使い方も可能です．

ここまで解説してきた「キャリアとモジュレータの周波数比」「EGの設定(各オペレータの出力波形の振幅)」「アルゴリズム」の3つを上手く制御することで，FM音源では多種多様な音色を作り出すことができるのです．

5.　8bit マイコンへの実装

実際にこのFM音源を8bitマイコンであるAVR ATMega328Pに実装した際の話を紹介します．

まず，デジタル的に波形を生成する場合には，サンプリング周波数を決定する必要があります．人間の聴覚の上限である20kHzを極力カバーしつつ，性能との折り合いを取り32kHzのサンプリング周波数としました．マイコンを定格内最速の20MHzで動かす前提では, 1サンプルを生成するのに使えるクロックサイクル数は625クロックサイクルとなります．実装はここにいかにして処理を詰め込むかという問題になってきます. FM音源をソフトウェア的に実装する場合,

・各種パラメータの設定
・波形メモリ参照用カウンタの更新
・波形メモリの参照
・EGの値の更新
・EGの値に基づいて波形の振幅を決定
・各オペレータの出力を混合
・DACへの出力

といった処理が主になります.

前に述べた処理のうち，各種パラメータの設定, DACへの出力以外がFM音源の「音をつくる」部分になりますが，これらの部分の大半が変数に対する加算と乗算，配列の参照で実現できる処理なので, 乗算器のある8bitマイコン程度であればなんとか処理しきれるレベルの処理だと思います. (もちろん32bitマイコンのほうが余裕はありますが…)

当然ですがオペレータ数にほぼ比例して計算量も増加します．ここも作成できる音色とのトレードオフですが，楽器にもよく見られる4オペレータを詰め込むのが私の技量では限界でした．サンプリング周波数を落とせば当然6オペレータも実現できるとは思いますが，ここは高音域の音質との相談です.

また，DACについては，SPI接続のMCP4921/4922を用いたり，ATMega328PのPWM出力を利用してDACを構成したりしました. 前者は12bit, 後者は8bitの分解能となりますが，8bitマイコンではFM音源の処理の際の演算に用いるbit幅をそこまで多く取れない(基本的に8bit，必要に応じて16bit)ので，この程度の分解能で十分かと思います．音質を重視するのであれば，32bitマイコン等を使い，内部演算のbit幅を32bitで行い，オーディオ用のDACを利用するとよいでしょう.

その他注意すべき事項としては固定小数点演算を実装する事によって高速化する必要があるという点(当然除算は使用厳禁です．シフトで置き換えましょう)と，使用するマイコンのROMサイズに応じた適切な波形メモリサイズを決定するといった点が挙げられると思います． 4オペレータの場合，アルゴリズムもそこそこのバリエーションが存在しますが, これらはすべてベタ書きで実装しました．固定小数点の桁あわせ等，細かいところで各アルゴリズム間で差異があるための判断でしたが，リソース的に余裕が有る場合はC++のクラスを活用して書くなど，色々と手の打ちようがあると思います.

6. さいごに

ふんわりした実装の話しか書けませんでしたが，実装した結果はWebサイト(http://t-techlab.net/)に掲載していますので，もしよろしければご覧ください.
FM音源について, 少しでも理解を深めていただけたなら幸いです.

7. 参考文献

[1]"DX7 オペレーティングガイドブック",ヤマハ株式会社