プロローグ

さまざまな知識が必要とされるグラフィックスの知識を身に付けよう

組み込み機器でもグラフィックス表示!

編集部

1. 組み込み機器に必要とされるグラフィックス表示とは

● 機器内部の状態を示すインジケータの役割がある

CPUを組み込み，制御を行う「組み込み機器」．組み込み機器もコンピュータである以上，入力→演算→出力を行う演算装置です．入力部分はスイッチやセンサ，演算部分はCPU，そして本特集で取り扱うグラフィックス表示は出力部分になります．

昔の出力装置といえば，LEDや7セグメントLED，ブザーなどの単純なものでした．それが現在ではフルカラー表示のディスプレイにウィンドウ・システム[注1]，3D表示などのリッチなグラフィックス表示が使われています．なぜそのようなグラフィックス表示が必要になったのでしょうか．

まず考えられるのは表示する情報量の増大です．LEDであれば点灯/消灯の2値，7セグメントLEDでは0～9までの数値しか表せません．これでは「動作中か否か」，「3けたの時速」などのような大まかな状態しか人間に知らせられません．

次に考えられるのは「使いやすさ」です．たとえば，機器内部でエラーが起きたとき，LEDでは「エラーが起きたか否か」，7セグメントLEDでは「3けたのエラー・コード」を表示するのが関の山でしょう．エラーが起きたとき，人間に対して迅速にエラー内容を伝えるには，わかりやすいエラー・メッセージを表示すべきです．そのために単純なLEDや7セグメントLEDだけではなく，ディスプレイを用いたグラフィックス表示が使われるようになりました．

このように，組み込み機器におけるグラフィックス表示は，機器内部の状態を示すインジケータの役割があります．

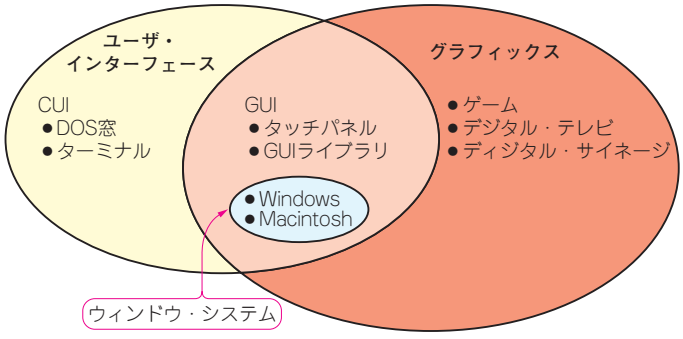

図1　グラフィックスと各機能の分類

● ユーザ・インターフェースとして

エラー表示から一歩進んで，ユーザ・インターフェースとしてグラフィックス表示を使うこともあります．いわゆるGUI（Graphical User Interface）です．GUIを使用している組み込みの機器として，コピー機や駅の券売機，「スーパーのレジ」に代表されるPOS端末が挙げられます．

ユーザ・インターフェースとは「コンピュータと対話するためのインターフェース」です．対話というのも抽象的ないい方なのですが，要するにコンピュータへ動作を指示したり，結果を見られるようにしたりするものです．

以前はコンピュータに対する指示は文字を入力し（たとえば，キーボードで「dir」と入力する），その結果が文字で表示されていました．このようなインターフェースをキャラクタ・ユーザ・インターフェース（CUI）といいます．俗にいう「DOS窓」がこれに該当します．

最近ではご存知のようにマウスでアイコンを指定し，結果がウィンドウなどで知らされます．これがGUIです．GUIを実現するためにはグラフィックス表示が必要です．この分類を**図1**に示します．

パソコンではMacintoshで有名になりWindows 95で普及したGUIですが，これと同じような使い勝手が組み込み機器にも求められるようになっています．物理スイッチがたくさん並んで操作法がわからない機器よりも，その場で必要なソフトウェア・ボタンが表示される方がわかりやすいでしょう．

● 表示それ自体が目的の機器もある

さらにグラフィックス表示それ自体が目的の機器もあります．一番わかりやすいのはテレビでしょう．アナログ・テレビにはCPUを内蔵していない製品もありますが，それも立派な組み込み機器です．もちろん昨今のデジタル・テレビはCPUと高速な演算チップ（DSPなど）の塊といえる高度な組み込み機器です．

さらに大きな市場となっているのはゲーム機です．ユーザを楽しませるため，派手で奇麗なグラフィックスの表示に各社がしのぎを削っています．

最近の流行としては，ディジタル・サイネージがあります．電子看板と訳されますが，店頭などに設置し，店名やその日のお買い得情報などを表示します．これらは機器内部に表示情報

注1：画面にウィンドウを表示するためのソフトウェアの総称．Windowsなど．WindowsはOSとウィンドウ・システムが一体となって提供されているが，UNIXでは後付けでX Window Systemというウィンドウ・システムが提供されている．

KEYWORD ── インジケータ，ユーザ・インターフェース，ビットマップ・ディスプレイ，ビデオ・コントローラ，グラフィックス・チップ，OpenGL

を保持していてそれを表示したり，ネットワーク経由で表示内容を設定できたりします．

● グラフィックス表示が必要になった

このように，組み込み機器にはグラフィックス表示が必要とされています．組み込み機器を作るためには，データの入力や演算，機器への出力だけでなく，人間に対して結果を出力するためのグラフィックス表示が必要なのです．

2. グラフィックス表示機器のハードウェア構成

● グラフィックス以前

ここまで説明してきたことを，実際のハードウェアに対応させて考えてみましょう．

グラフィックスと呼ばれる以前の表示デバイスとしては，LEDや7セグメント表示器などが使われていました．これらはON/OFFの2値で表現できるため，CPUの汎用I/Oポート（GPIO：General Perpose I/O port）にLEDをつなげるだけで実現できます．ソフトウェア的にはCPUのGPIO端子を出力に設定し，そこを‘0’か‘1’に設定するだけです〔**図2**(**a**)〕．

7セグメントLEDも同様です．7セグメントLEDはLEDが七つ並んだだけの構造なので，GPIOを7本使えば接続できます．構造が手軽なことから，現在でも使われています．

● キャラクタ・ディスプレイの時代

次に登場したのが文字を表示する機器です．よくあるのが「数字とアルファベット，カタカナ」だけの表示です．メモリ容量が少なかった時代に使われました．たとえば1文字を1バイトで表現し，文字数が1画面40×25文字の場合，1,000バイト＝約1Kバイトで画面全体を表現できます．

最近では次に述べるビットマップ・ディスプレイに置き換わってしまい，ロスト・テクノロジとなっています．

● ビットマップ・ディスプレイの初期

メモリ価格の低下により可能になったのがビットマップ・ディスプレイです．画面をいわゆる「ドット」で表現する方式で，現在の主流です．**図2**(**b**)にビデオ・コントローラで実現するビットマップ・ディスプレイの例を示します．

この方法はCPUの外に画面を保存するためのメモリとしてビデオRAM（VRAM）を接続します．たとえば，1ドットに対してRGB（赤緑青）各8ビット（1バイト）として，画面のドット数が640ドット×480ドットの場合，3バイト×640×480＝921,600＝約1Mバイトになります．1Kバイト程度で実現できたキャラクタ・ディスプレイとは，文字通りけた違いの大容量が必要です．今では1Mバイトのメモリは1円程度の計算になりますが，80年代後期あたりでは3万円近くしたものです．

ビットマップ・ディスプレイの実現には，CPUにVRAMとビデオ・コントローラ（CRTコントローラともいう）を接続し

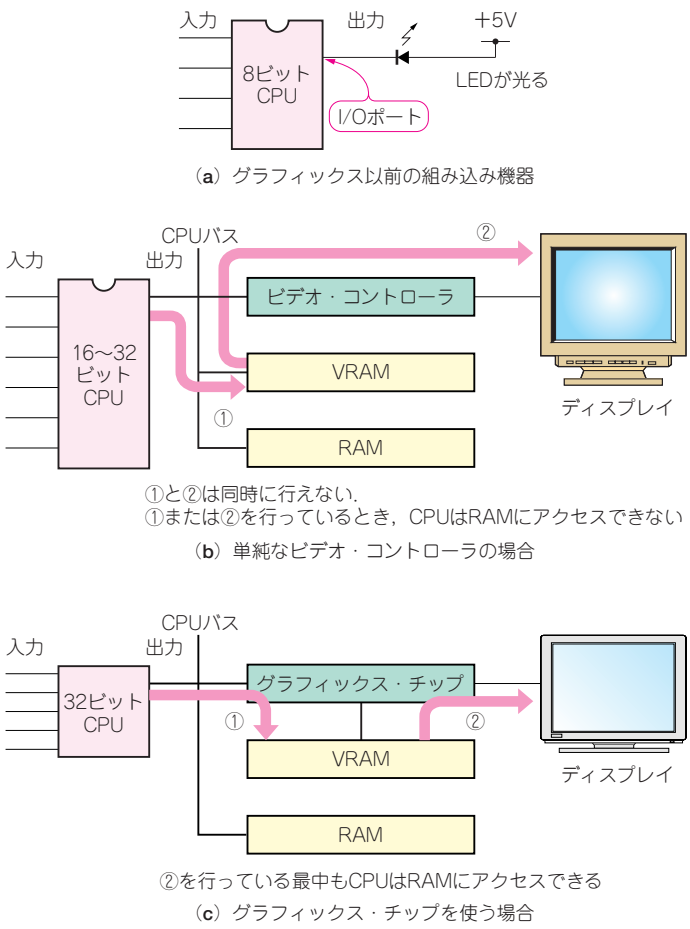

図2　グラフィックスを実現するハードウェアの構成

ます．これはVRAMから画像データを読み込み，ディスプレイへ送信するものです．CPUによる描画では，

①CPUからVRAMへ画像データを書き込む
②ビデオ・コントローラがVRAMからデータを読み出し，ディスプレイに画像データを送り出す

という2種類の作業が必要になります．

ここで問題になるのが「バスの飽和」です．バスというのは信号線をまとめたものです．この場合，CPUに直結しているバスであることから「CPUバス」と呼ばれます．CPUもビデオ・コントローラも，VRAMへのアクセスはバスを経由して行います．問題なのは「CPUとビデオ・コントローラのどちらか片方しかVRAMにアクセスできない」ということです．つまりCPUがVRAMへ書き込んでいるときは，ビデオ・コントローラはVRAMのデータを読み込むことができないし，逆にビデオ・コントローラがVRAMのデータを読み込んでいるときはCPUはVRAMへ書き込めません[注2]．そのため，片方の作業を行うときは，もう片方が待たされます．CPUバスはVRAMへのアクセスだけでなく，（普通の）メモリへのアクセスにも使われます．たとえばCPUがプログラムの実行を行うための命令の読み込

みや，変数の値の読み書きもメモリ・アクセスです．これらがビデオ・コントローラ実行中は待たされることになります．

このようにバスが「埋まって」しまい，自由に使えないことによる性能の低下を「バスの飽和」と呼びます．

さらに，CPUによるVRAM書き込みも時間がかかります．たとえば，640ドット×480ドット×3バイトのVRAMを塗りつぶすには，640 × 480 × 3 = 921,600 = 100万回のメモリ・アクセスが必要です．グラフィックスを扱うというのはこのくらい「重い」処理なのです．

● 現在のグラフィックス機器

重い処理であるグラフィックスをCPUで扱うにはもっと速いCPUを使えば解決します．しかし速いCPUを使うということは，消費電力が大きくなるということです．バッテリ動作させる組み込み機器の場合，連続使用時間の短縮に直結しますし，発熱もします．

バスの飽和とCPUによるグラフィックス描画が重いという問題を解決するために，現在ではビデオ・コントローラをさらに高機能にした専用のグラフィックス・チップが搭載されています[注3]．

まずバスの飽和の解決です．**図2(c)**を見てください．決定的に違うのが，VRAMがグラフィックス・チップの先につながっていることです．CPUからVRAMへ書き込む作業はこれまで通りCPUバスを通りますが，VRAMから読み出してディスプレイに表示するときはCPUバスを通りません．これにより，ディスプレイ表示中でもCPUは(VRAM以外の)RAMへアクセスできます．

CPUによるグラフィックス描画が重いという問題は，グラフィックス・チップの高機能化で解決します．CPUからグラフィックス・チップに対し，

```
FILL(0,0,639,479)
```

という命令を送ったときに，グラフィックス・チップがこれを理解してVRAMを塗りつぶすようにします．そうすればグラフィックス・チップが全力でVRAMを塗りつぶしている間，CPUはほかの仕事ができます．CPUは数命令実行するだけで済み，バスも数クロック分使うだけです．

さらに単純な塗りつぶしだけでなく，線の描画や画像のコピー，3D画像の表示など，高度な機能を持ったグラフィックス・チップが登場しています．カー・ナビゲーション・システムなど，高度な3Dグラフィックス表示が必要な分野ではグラフィックス・チップは必須です．最近の組み込み向けマイコンでは，マイコンの中にCPUとグラフィックス・チップの双方を内蔵した製品も出てきています．

3. グラフィックス表示機器のソフトウェア構成

● 単純なメモリ・アクセスからOpenGLまで

ソフトウェア面から見てグラフィックスを描画するということは，結局はVRAMに値を書き込むことに帰結します．C言語でいえばVRAMの存在するアドレスに`*vram++`で書き込めばいいわけで，特別なテクニックは必要ありません．

しかし前述したとおりこれでは遅いのです．グラフィックス・チップの専用コマンドを使って高速化するためには，グラフィックス・チップ専用コマンドを発行するソフトウェアが必要です．これらは一般的には，グラフィックス・チップのメーカからライブラリの形で提供されるため，自分で書く必要はありません．

さらにその上にグラフィックス共通API(Application Program Interface)であるOpenGLを搭載するのが昨今のトレンドです．OpenGLは，異なったCPUやグラフィックス・チップでも，同じソース・コードで画面描画できるように統一されたAPIです．OpenGLを使って書かれていれば，(基本的には[注4])Windowsパソコンであっても，組み込み機器であっても同じプログラムでグラフィックス描画ができます．この場合，OpenGLを理解するためのライブラリが必要になります．先ほどのグラフィックス・チップをからめて考えた場合，OpenGLのAPIである`glDrawArrays()`などを理解して，グラフィックス・チップの専用コマンドを発行しています．

● GUIを実現するには専用のソフトウェアが必要

GUIもソフトウェアで実現します．単純なグラフィックス描画機能では，単に「点を打つ」，「画面を塗りつぶす」ということしかできません．GUIを実現するためには「ボタンを表示する」のような機能が必要です．これは「ボタンを表示する」というAPIを用意し，これが呼ばれたら点を打つことによってボタンを表示するというライブラリが必要になります．GUIを実現するためにはこのようなAPIを作り込む必要があります．

＊　　＊　　＊

グラフィックス機能の基本的な知識を習得するには第1章と第2章を，グラフィックスのためのハードウェア知識は第3～5章を，OpenGLについては第6章，グラフィックスを使ったGUIの構築は第7章で解説しています．

注2：CPUとビデオ・コントローラが同時にアクセスできる「デュアル・ポートRAM」という特殊なメモリも存在する．ただし普通のメモリより高価．なお，FPGAであるSpartanシリーズの内蔵ブロック・メモリはデュアル・ポートRAMであるため，この用途に便利(ただし容量があまり大きくない)．

注3：ビデオ・コントローラとグラフィックス・チップという用語に明確な定義はないようだ．ここでは慣例に従って，ビデオ・コントローラ＝VRAMの内容をディスプレイへ転送する単純なIC，グラフィックス・チップ＝ビデオ・コントローラの機能に加えて描画高速化機能を追加したもの，としている．また，安価なグラフィックス・チップではVRAMを独立して用意せず，VRAM以外の(普通の)RAMをVRAMとして使っている製品もあるし，逆にビデオ・コントローラでもVRAMがビデオ・コントローラの先につながっている製品もある．

注4：ちょっと違う．詳細は第6章のEGLの解説を参照のこと．