ASCII文字フォントをハードウェアで画面上に描画して文字を表示する

第4章 FPGA内蔵ブロックRAMを使ったテキストVRAM表示コントローラの設計事

Windowsなどの GUIでは，文字をグラフィックス画面に描画している．そのため，CPU性能が低いと画面表示が遅くなってしまう．CPU性能が低い場合でも高速に文字を表示する方法として，文字コードを書き込んだだけで文字を表示できるテキストVRAM方式がある．ここでは，FPGAのブロックRAMを使ったテキストVRAM表示コントローラの設計事例について解説する．（編集部）

熊谷 あき

ここでは，CQ出版社から発売されている「組み込みシステム開発評価キット」を使い，FPGA内蔵ブロックRAM機能を生かしたテキストVRAM表示コントローラの設計について解説します．

1 テキストVRAMとは何か

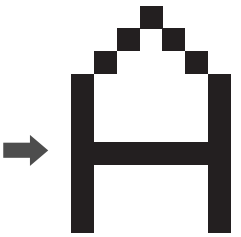
「画面に文字を表示する」方法としては，大きく分けて次の二つが一般的です．

● GUIによる文字表示方式

GUI（Graphical User Interface）システムでは，文字をドット単位でグラフィックス画面上に描画することで，画面に文字を表示します．この方式はWindowsではもちろんのこと，最近では，GUIを持っている組み込み機器でも一般的に使われています．

フレーム・バッファ・アドレス　+0 +1 +2 +3 +4 +5 +6 +7 ....

0000h　00 00 00 FF 00 00 00 00 ....
0400h　00 00 FF 00 FF 00 00 00 ....
0800h　00 FF 00 00 00 FF 00 00 ....
0C00h　FF 00 00 00 00 00 FF 00 ....
1000h　FF 00 00 00 00 00 FF 00 ....
1400h　FF FF FF FF FF FF FF 00 ....
1800h　FF 00 00 00 00 00 FF 00 ....
1C00h　FF 00 00 00 00 00 FF 00 ....



このフレーム・バッファの場合，1文字分を描画するために64バイトのデータが必要

（a）グラフィックスVRAMではドット単位で文字を描画

文字描画のためのアクセラレーション機能がない画面表示コントローラの場合，グラフィックス画面上に文字を描画するには，CPUで画面上に点を打っていくしかありません．そのため，CPU性能が低い場合は，文字描画の性能も上がりません．その代わり，画面上の文字の表示位置や表示サイズ，文字フォントなど，プログラム次第でどのようにでも設定可能なので，自由度の高い文字表示を行うことができます．従って，グラフィックス表示性能のある画面表示コントローラを搭載したシステムでは，昨今ではほとんどがこの方式を採用しています．

この方式を，ここではグラフィックスVRAM方式と呼ぶことにします．

● CPU性能が低くても文字を表示したい

CPU性能が低くても画面に文字を表示する方法として，大昔から使われている方法があります．CPUが表示したいテキスト座標位置のメモリに文字コードを書き込みます．すると，そのテキスト座標に相当する画面表示位置に，書き込まれた文字コードに対応する文字をハードウェアがCPUの力を借りずに表示するシステムです．この方式を，一般にテキストVRAM方式と呼びます．

この方法を用いて画面に文字を表示させるために，CPU

テキストVRAMアドレス　+0 +1 +2 +3 +4 ....

0000h　41 42 43 44 45 ....　→　ABCDE....
0050h　61 62 63 64 65 ....　→　abcde....

テキストVRAM方式は文字コードを書き込むだけ　1文字1バイト

（b）テキストVRAM方式では文字コードを書き込むだけ

図1　グラフィックスVRAM方式とテキストVRAM方式の違い

KEYWORD ── グラフィックスVRAM，テキストVRAM，文字コード，属性コード，文字フォント，ブロックRAM

は文字コードを書き込むだけで済みます．けれども，表示される文字の大きさやフォントを変更することはできない，もしくはそのコントローラが持つ機能の範囲内でのみ可能という，自由度が非常に制限された表示しか行うことができません．

その昔，8 ビット・パソコンの全盛期で CPU 性能が低かった時代は，この方式が一般的でした．PC-9801 シリーズは，16 ビット・パソコンでもテキスト VRAM 方式を採用していました．

グラフィックス VRAM 方式とテキスト VRAM 方式の違いを**図 1** に，両方式の比較を**表 1** に示します．

2 テキスト VRAM の仕様検討

● フォントのデザイン

まず，どのような仕様のテキスト VRAM コントローラを実現するかを考えます．とりあえずの目標として，最も基本的な ASCII 文字を表示できるコントローラを実現してみましょう．

文字を表示するには，表示する文字一つ一つに対してその文字の形を示すフォント・データが必要です．一般に ASCII 文字を表示する場合，横 8 ドット×縦 8 ドットあれば表示可能です．今回は漢字表示については考慮していませんが，後述するように将来的な拡張性として漢字表示の可能性は考えておきたいところです．そこで，横は 8 ドットですが縦は 16 ドットで ASCII 文字を表現してみることにします（**図 2**）．

なお，今回は新規にフォントをデザインせず，以前本誌で掲載された参考文献（1）のフォントを流用させていただきました．このフォントの基本デザインは，文字の上と下，さらに右側にドットを打たないスペースを確保しています．これによって，文字と文字の隙間，および行と行の隙間を確保できるため，画面に表示された文字が読み取りやすくなります．

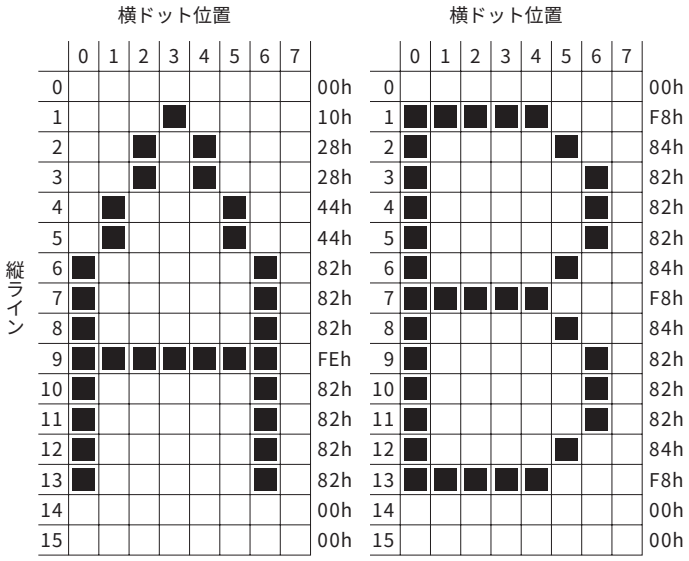


図 2　横 8 ドット×縦 16 ドットの ASCII フォント例

このフォントをデータとして保持するには，表示ラインごとに文字を横方向に輪切りにして，ドットがあるかないかの情報を 1 ライン当たり 1 バイトで保持します．画面表示の走査線は横方向なので，データを横方向に持つのが効率的です．1 文字当たり 16 バイトで，ASCII 文字が 128 文字なら 2K バイトになります．ここでは便宜上，ASCII 文

表 1　グラフィックス VRAM 方式とテキスト VRAM 方式の比較

	グラフィックス VRAM	テキスト VRAM
CPU 描画負荷	重い	軽い
フォント自由度	高い	低い
表示用メモリ容量	大容量	小容量
ハードウェア回路規模	比較的大規模（フレーム・バッファ機能のみなら小規模）	比較的小規模

上位 4 ビット

下位4ビット	0	1	2	3	4	5	6	7
0				0	@	P	`	p
1			!	1	A	Q	a	q
2			”	2	B	R	b	r
3			#	3	C	S	c	s
4			$	4	D	T	d	t
5			%	5	E	U	e	u
6			&	6	F	V	f	v
7			’	7	G	W	g	w
8			(	8	H	X	h	x
9			)	9	I	Y	i	y
A			*	:	J	Z	j	z
B			+	;	K	[	k	{
C			,	<	L	¥	l	|
D			−	=	M	]	m	}
E			.	>	N	^	n	~
F			/	?	O	_	o	

図 3
ASCII 文字一覧