

キャラクタコード (Character Code) は次のように決められている。

上位4ビット

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
下位4ビット	0		DE		0	@	P		p			-	タ	ミ		x	
	1	SH	D1	!	1	A	Q	a	q			。	ア	チ	ケ		円
	2	SX	D2	"	2	B	R	b	r			「	イ	ツ	メ		年
	3	EX	D3	#	3	C	S	c	s			」	ウ	テ	モ		月
	4	ET	D4	\$	4	D	T	d	t			,	エ	ト	ヤ		日
	5	EQ	NK	%	5	E	U	e	u			・	オ	ナ	ユ		時
	6	AK	SN	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		分
	7	BL	EB	'	7	G	W	g	w			ア	キ	ヌ	ラ		秒
	8	BS	CN	(8	H	X	h	x			イ	ク	ネ	リ		
	9	HT	EM)	9	I	Y	i	y			ウ	ケ	ノ	ル		
	A	LF	SB	*	:	J	Z	j	z			エ	コ	ハ	レ		
	B	HM	EC	+	;	K	[k	{			オ	サ	ヒ	ロ		
	C	CL		,	<	L	¥	l				ヤ	シ	フ	ワ		
	D	CR		-	=	M]	m	}			ユ	ス	ヘ	ン		
	E	SO		.	>	N	^	n	~			ヨ	セ	ホ	°	/	
	F	SI		/	?	O	_	o				ッ	ソ	マ	°	\	

01H ~ 1FH はコントロールキャラクタ
80H ~ 9FH, E0H ~ EBH はグラフィックキャラクタ

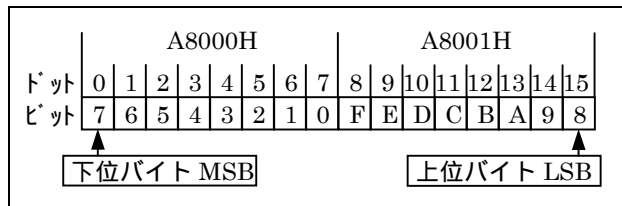
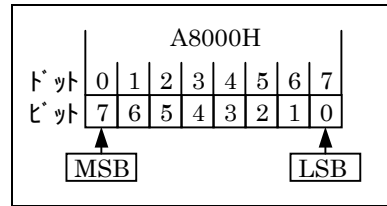
例えば、画面の左上端 (0 行 (X座標 0) , 0 桁 (Y座標 0)) に「 1 」を表示する場合には、物理アドレス「 A0000H 」に「 31H 」のデータを書き込めばよい。

次に、「グラフィックVRAM (GVRAM) 」を説明する。グラフィック画面はテキスト画面とは独立に横 640 ドット、縦 400 ドットから構成され、GVRAM の 1 ビットが CRT 上の 1 ドットに対応している。GVRAM は 4 分割されたメモリ構成であり、GVRAM0 は青 (Blue) , GVRAM1 は赤 (Red) , GVRAM2 は緑 (Green) であり、GVRAM3 はインテンシティ (通常使用しない) がある。各 GVRAM のビットに「 1 」を書き込むことにより対応した色でドット表示される。この 3 つの GVRAM への書き込みを組み合わせれば合計 8 種類の色を表示できる。各 GVRAM の構成は同じであり、その物理アドレス、すなわちセグメントが異なるだけである。例えば、GVRAM0 のメモリ構成と CRT ドット

	0				ドット				639			
	0	ドット	15	16	ドット	31	624	ドット	639			
0	A8000	A8001	A8002	A8003	A804E	A804F					
1	A8050	A8051	A8054	A8053	A809E	A809F					
2					
ドット					
398	AFC70	AFC71	AFC72	AFC73	AFCAE	AFCAF					
399	AFCB0	AFCB1	AFCB2	AFCB3	AFCFE	AFCFF					

との関係は次のとおりである。

1 アドレスの1バイトはグラフィック画面の横8ドットに対応している。例えば、X=0, Y=0 ドットは物理アドレス A8000H の1バイトのMSBに、X=7, Y=0 ドットそのLSB (Least Significant Bit) に対応する。また、ワード(16ビット)でGVRAMをアクセスする場合には下図のように逆ワードとなり、1ワード上位バイトのLSBが15ドット目に対応する。



応する。

他のGVRAMも同様なRAM構成であり、開始物理アドレスは

GVRAM1 (赤) : B0000H GVRAM2 (緑) : B8000H

となっている。また、色の組み合わせは以下のようなものである。

	黒	青	赤	紫	緑	水	黄	白
GVRAM0 (B)	0	1	0	1	0	1	0	1
GVRAM1 (R)	0	0	1	1	0	0	1	1
GVRAM2 (G)	0	0	0	0	1	1	1	1

例えば、画面左上に横8ドットで緑を表示するには、物理アドレス「B8000H」に「FFH」のデータを書き込み、また、黄色表示とする時には物理アドレス「B0000H」と「B8000H」の2ヶ所のVRAMに「FFH」のデータを書き込みめばよい。さらにこれらの表示を2ドット間隔で表示するには、書き込むデータを「11001100B」すなわち「CCH」とする。

CPUの命令を表すアセンブラ言語によるこのようなVRAMを直接アクセスする表示方法をマスターすれば、BASIC言語やC言語によるプログラムよりも高速に様々な文字や図形を描くことができる。