Chapitre 7

L'affichage en mode texte

Après avoir étudier l'implémentation du périphérique le plus utilisé, à savoir le clavier, nous allons aborder dans ce chapitre l'affichage. Les premiers affichages se faisaient uniquement en **mode texte**, puis est apparu le **mode graphique**.

Le mode texte est encore important de nos jours car sa programmation est plus simple. L'ordinateur, lorsqu'il démarre, est toujours en mode texte, à la fois parce que c'est plus simple mais aussi parce qu'il existe une norme respectée par toutes les *cartes graphiques*.

Nous allons voir une description fonctionnelle de l'aspect matériel et surtout son étude du point de vue de la programmation système.

Dans une première étape, pour la programmation système, il suffit de connaître le principe général du fonctionnement de l'affichage. Nous verrons, en effet, que seule la manipulation de la *mémoire graphique* (*video memory* en anglais) est nécessaire dans ce but.

La conception des macro-instructions concernant l'affichage nous montre un aspect nouveau du BIOS : l'indépendance par rapport au matériel utilisé. En effet, dès le premier PC, on pouvait choisir entre deux cartes graphiques. Ces cartes ne se programment pas de la même façon. Le BIOS présente des macro-instructions indépendantes de la carte utilisée mais, bien sûr, lors de la conception de ces macro-instructions, les procédures dépendront de la carte utilisée.

7.1 Principe physique d'un moniteur

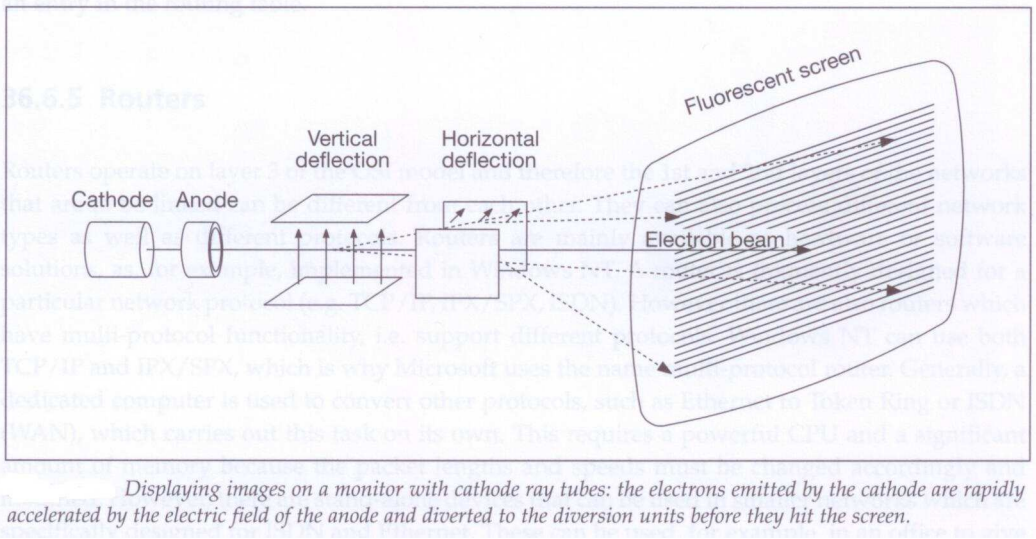
L'affichage se fait sur l'**écran** d'un **moniteur**. Celui-ci comporte deux câbles : un câble d'alimentation électrique et un câble relié à l'unité centrale, plus exactement à la **carte graphique** (*video card* en anglais).

Vous avez peut-être déjà changé la carte graphique de votre ordinateur. Elle est tout simplement insérée dans l'une des *positions d'extension* (*slot*) de la carte mère et assujettie au boîtier grâce à une vis.

7.1.1 Tube cathodique

La partie la plus imposante du moniteur, à l'époque de la conception de l'IBM-PC, est le **tube cathodique** (**CRT** pour l'anglais *Cathode Ray Tube*). Il s'agit d'un tube analogue à celui d'un poste de télévision, dont le schéma est donné ci-dessous.

De nos jours, on utilise des *écrans plats* à led (*diode électroluminiescente*) mais, pour la programmation, le modèle du CRT est conservé.



Displaying images on a monitor with cathode ray tubes: the electrons emitted by the cathode are rapidly accelerated by the electric field of the anode and diverted to the diversion units before they hit the screen.

FIGURE 7.1 – Principe du tube cathodique

Principe.- Le tube cathodique est principalement une bouteille en verre à fond plat vide d'air contenant quelques fils. Le fond, revêtu à l'intérieur de matériaux fluorescents constitue l'**écran**. Ces matériaux *fluorescents* ont la propriété de s'illuminer durant un très court instant lorsqu'ils sont frappés par un électron, d'une couleur qui dépend du matériau.

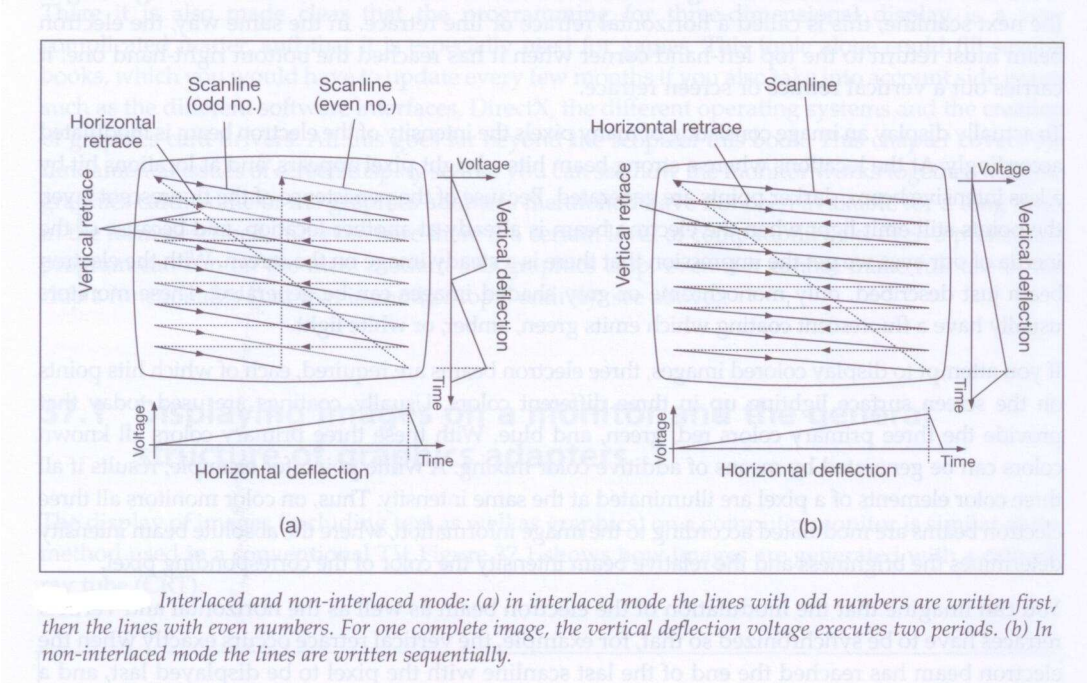
Les électrons sont engendrés par une plaque de cuivre, appelée **cathode**, qui est située à l'opposé de l'écran. Cette cathode est chauffée à blanc par une résistance électrique pour pouvoir libérer les électrons. Les électrons s'éparpillent dans toutes les directions. Une **anode**, plaque de cuivre en forme de tube présentant une différence de potentiel de signe positif par rapport à la cathode, a pour fonction de les accélérer et de les diriger vers le centre de l'écran.

Les électrons peuvent être déviés à la fois verticalement et horizontalement par deux champs électriques (créés par les **déflecteurs** vertical et horizontal) de façon à atteindre n'importe quel point de l'écran. En fait seul un nombre limité de points de l'écran nous intéressent. L'écran est

divisé en un certain nombre de **lignes** horizontales, chaque ligne comportant un certain nombre de points, appelés **points écran** (**pixel** en anglais pour *PICTure ELement*).

Balayage de l'écran.- Bien que cela serait possible, on n'envoie pas un faisceau d'électrons sur le premier pixel désiré puis ensuite sur le second et ainsi de suite. L'écran est parcouru en permanence ligne par ligne, l'intensité du faisceau d'électrons variant de façon à frapper ou non un pixel (cas du tout ou rien) ou à le frapper plus ou moins (cas d'une échelle d'intensité).

La première ligne est décrite, puis la seconde et ainsi de suite. À la fin d'une ligne, on doit aller au début de la suivante : on a alors un **rebroussement horizontal** (*horizontal retrace* en anglais), comme le montre la figure ci-dessous :



Interlaced and non-interlaced mode: (a) in interlaced mode the lines with odd numbers are written first, then the lines with even numbers. For one complete image, the vertical deflection voltage executes two periods. (b) In non-interlaced mode the lines are written sequentially.

FIGURE 7.2 – Principe de l'entrelacement

Lorsqu'on est arrivé à la dernière ligne, il faut revenir au début de la première ligne : on a alors un **rebroussement vertical** (*vertical retrace* en anglais).

Persistance.- Grâce à la persistance de la couche fluorescente, les points émettent encore de la lumière après le passage du faisceau d'électrons. Grâce à l'inertie concomitante de notre œil, nous avons donc l'impression de voir une image formée sur l'écran alors qu'un seul pixel est frappé à la fois.

Moniteur monochrome et couleur.- Nous venons de décrire le principe d'un **moniteur monochrome**. On ne parle pas de noir et blanc, contrairement aux postes de télévision, car suivant la couche fluorescente la lumière émise est verte, ambre ou blanche.

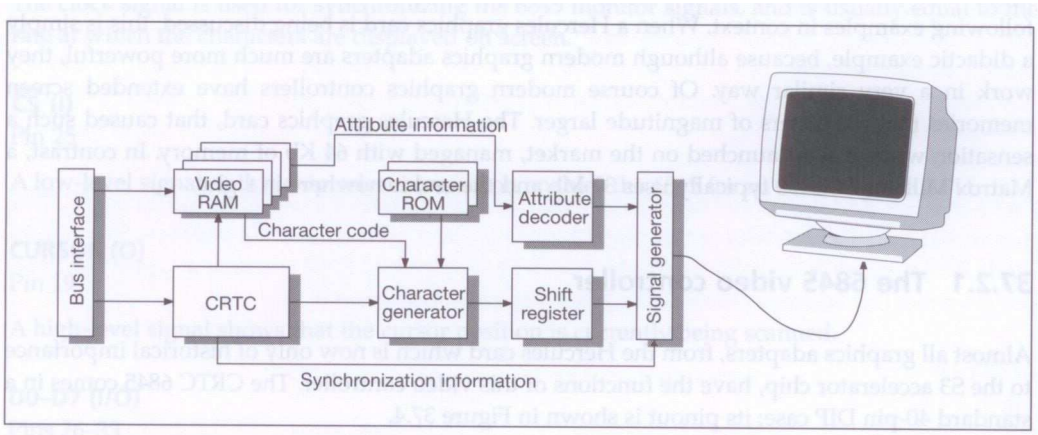
Le même principe s'applique aux **moniteurs à intensité de gris**. Comme nous l'avons vu, l'intensité du faisceau d'électrons est plus ou moins grande dans ce cas.

Dans le cas d'un **moniteur couleur**, on utilise une couche fluorescente constituée de cellules, chaque cellule étant revêtue d'un matériau fluorescent pouvant émettre une lumière rouge, verte ou bleue. On peut obtenir presque toutes les couleurs connues grâce à ces **couleurs primaires** grâce au **principe additif des couleurs**. Un point blanc, par exemple, est obtenu en illuminant les trois pixels contigus de la même intensité.

7.1.2 Principe d'une carte graphique

Les tubes cathodiques sont utilisés dans de nombreux appareils, dont le plus connu est la télévision, le premier ayant été l'oscilloscope. Un moniteur traite un signal numérique, contrairement à un poste de télévision qui traite un signal analogique. Le **contrôleur graphique** est chargé d'afficher un pixel sur l'écran du moniteur.

Une carte graphique (pour le mode texte) est constituée suivant le schéma suivant :



Block diagram of a graphics adapter.

FIGURE 7.3 – Principe d'une carte graphique

— La partie essentielle de la carte graphique est le **contrôleur de tube cathodique** (**CRTC** pour l'anglais *Cathode Ray Tube Controller*) qui supervise les fonctions de la carte et qui envoie les signaux de contrôle nécessaires.

— Le microprocesseur accède à la mémoire graphique *via* l'**interface du bus** pour écrire les informations qui définissent le texte que doit afficher le moniteur.

— Le CRTC accède sans arrêt à la **mémoire graphique** (*Video RAM*) pour lire les caractères à afficher et les transférer au **générateur de caractères** (*character generator*). Les caractères sont en général spécifiés par leur code ASCII et par un **attribut**. Cet attribut indique de quelle façon le caractère doit être affiché : par exemple normal ou inversé.

La mémoire graphique est de la mémoire RAM située sur la carte graphique mais dont les adresses sont une partie de celles accessibles par le microprocesseur.

— Il existe une mémoire ROM, appelée **ROM caractères** (*Character ROM*), qui contient pour chaque code ASCII le motif du caractère correspondant.

— Le générateur de caractère convertit le code ASCII, en utilisant le motif de la ROM caractères, en une suite de bits de pixel qu'il transmet au **registre de décalage** (*Shift Register* en anglais).

— Le **générateur de signal** (*Signal generator*) engendre les signaux nécessaires au moniteur, en utilisant le flot de bits du registre de décalage, les informations sur l'attribut et les signaux de synchronisation émis par le CRTC.

7.2 Le contrôleur graphique Motorola 6845

7.2.1 Génération des caractères en mode texte

Pour un moniteur, l'écran est modélisé conformément à ce qui apparaît à la figure 7.4. On passe de ce modèle au pilotage (analogique) du tube cathodique grâce au contrôleur graphique.

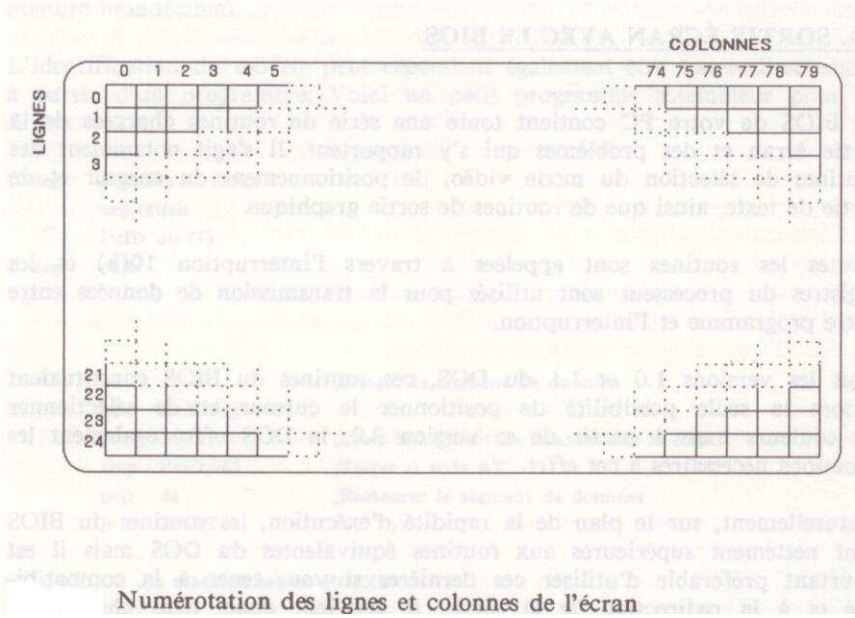


FIGURE 7.4 – Numérotation des lignes et colonnes de l'écran d'un moniteur

Le caractère supérieur gauche est celui qui est fourni immédiatement après un rebroussement vertical. Le contrôleur graphique place sur les broches d'adresse l'adresse en mémoire graphique de ce premier caractère et récupère alors le code du caractère ainsi que ses attributs. Le code du caractère est envoyé au générateur de caractère comme premier index de la ROM des caractères. Le numéro de ligne des pixels est alors 0, c'est-à-dire que le contrôleur graphique envoie également cette donnée pour accèder à la première ligne de la matrice du caractère. Les bits de la matrice de pixels sont transférés du registre de décalage au générateur de signal. Si le générateur de signal reçoit un '1' du registre de décalage, il génère un signal vidéo correspondant à la couleur de premier plan du caractère. S'il reçoit un '0', il génère un signal vidéo correspondant à la couleur de fond du caractère.

Le contrôleur graphique place alors sur les broches d'adresse l'adresse en mémoire graphique du second caractère de la ligne. De même les bits de la première ligne de la matrice de pixels sont transférés du registre de décalage au générateur de signal. On fait de même pour le troisième caractère, le quatrième caractère, et ainsi de suite jusqu'au dernier caractère de la ligne.

La première ligne de pixels est ainsi affichée sur le moniteur. Lorsque le faisceau d'électrons atteint la fin de la ligne de pixels, le contrôleur graphique active la sortie HS pour produire un rebroussement horizontal et une synchronisation horizontale. Le faisceau d'électrons va alors au début de la ligne de pixels suivante. Après chaque rebroussement horizontal, le contrôleur graphique incrémente les broches de numéro de ligne (du motif du caractère). Cette adresse de ligne donne le décalage pour la matrice de pixels permettant d'afficher les caractères.

On décrit de même les caractères un, deux, trois et ainsi de suite jusqu'au dernier caractère de la même première ligne. Mais, cette fois-ci, c'est la deuxième ligne de pixels qui est affichée.

On recommence pour afficher la troisième ligne de pixels puis la quatrième et ainsi de suite.

Après le nombre voulu de lignes de pixels, l'adresse de numéro de ligne du motif du caractère revient à la valeur 0 et le contrôleur graphique fournit une nouvelle adresse de caractère. Une deuxième ligne de caractères est ainsi affichée, représentant un certain nombre de lignes de pixels.

De même pour la troisième ligne de caractères, puis la quatrième et ainsi de suite.

À la fin de la dernière ligne de caractères, le contrôleur graphique réinitialise l'adresse de ligne de caractères, l'adresse de ligne de pixels et engendre un signal VS de façon à obtenir un rebroussement vertical et une synchronisation verticale.

Un écran a donc été complètement affiché. On recommence ainsi indéfiniment. La plupart du temps, cela revient à **rafraîchir** l'écran. Si on a changé quelque chose dans la mémoire graphique, le contenu de l'écran change également.

7.2.2 Description du brochage du MC6845

Le contrôleur graphique des premières cartes graphiques de l'IBM-PC était le Motorola 6845. Ce n'a plus été le cas par la suite mais la plupart des contrôleurs graphiques ont une compatibilité ascendante avec celui-ci. Ses fonctionnalités sont toujours utilisées dans les modes texte de démarrage de l'ordinateur.

FIGURE 7.5 – Aspect du contrôleur graphique Motorola MC6845

Le contrôleur graphique 6845 se présente sous la forme d'un circuit intégré DIP à 40 broches, comme le montre la figure 7.6 :

— Les broches 1 (`GND`, pour *GRounD*, terre) et 20 (`Vcc`) servent à l'alimentation électrique du contrôleur, en général 0 V et +5 V.

— La broche 21 (`CLK`, pour *CLocK*, horloge, en entrée seulement) reçoit le signal d'horloge utilisé pour synchroniser les signaux du moniteur. Celui-ci est en général égal au taux auquel les caractères doivent être affichés à l'écran.

— Les 14 broches 4 à 17 (`MA0-MA13`, pour *Memory Address*, en sortie seulement), spécifient l'adresse de la mémoire graphique d'un caractère dont le contrôleur va afficher une ligne.

— Les 5 broches 34 à 38 (`RA0-RA4`, pour *Row Address*, en sortie uniquement) spécifient la ligne du motif du caractère à afficher. Un caractère peut donc occuper au plus 32 lignes en hauteur.

— La broche 39 (`HS`, pour *Horizontal Synchronization*, en sortie uniquement) envoie un signal de synchronisation horizontale, ce qui produit un rebroussement horizontal.

— La broche 40 (`VS`, pour *Vertical Synchronization*, en sortie uniquement) envoie un signal de synchronisation verticale, ce qui produit un rebroussement vertical.

— Les huit broches 26 à 33 (`D0-D7`, pour *Data*), reliées au bus des données, permet une communication entre les registres internes du 6845 et le microprocesseur, et donc la programmation de celui-ci.

— La broche 25 ($\overline{\texttt{CS}}$, pour *Chip Select*, en entrée uniquement) permet de savoir si le microprocesseur désire accéder au 6845 (signal bas).

— La broche 23 (`E`, pour *Enable*, en entrée uniquement), active le bus des données par une transition haut-bas. Cela sert de pulsation d'horloge pour lire les données d'un de ses registres ou écrire sur l'un d'eux.

— La broche 22 (`R`/$\overline{\texttt{W}}$, pour *Read/Write*, en entrée uniquement) permet de savoir si le microprocesseur veut lire un registre interne du 6845 (signal haut) ou écrire sur celui-ci (signal bas).

— La broche 24 (RS, pour *Register Select*, en entrée uniquement) permet de savoir si la prochaine donnée sera un index de registre interne (signal bas), ou une donnée pour un tel registre (signal haut).

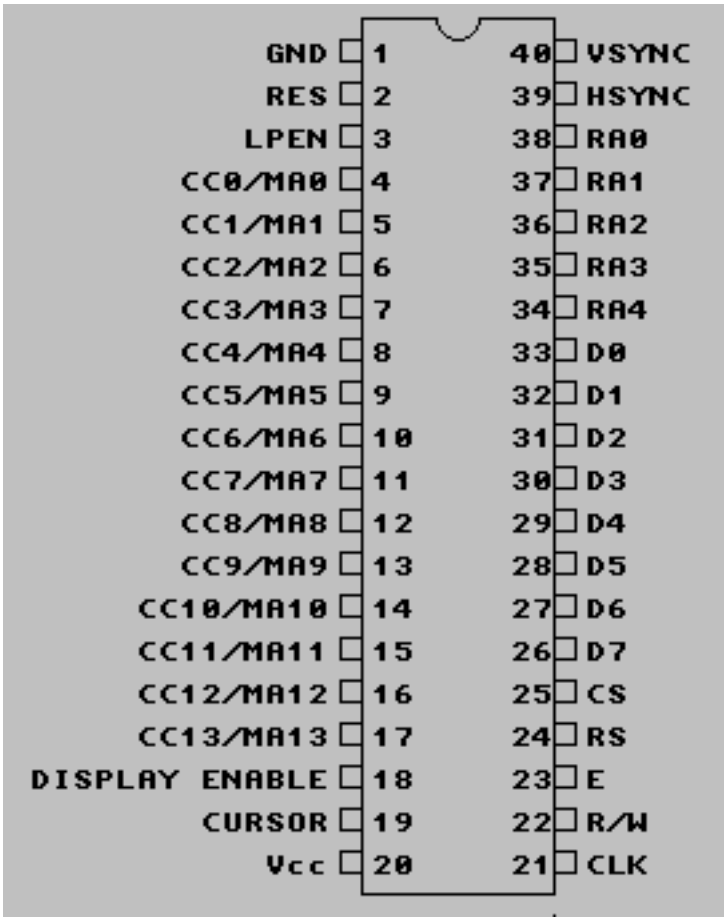


FIGURE 7.6 – Brochage du contrôleur graphique Motorola MC6845

— La broche 2 ($\overline{\text{RESET}}$, en entrée uniquement) permet de réinitialiser le 6845 (signal bas).

— La broche 19 (CURSOR, en sortie uniquement) indique, par un signal haut, qu'on se trouve actuellement à la position du curseur.

— La broche 18 (DE, pour *Display Enable*, en sortie uniquement) indique, par un signal haut, que le 6845 fournit actuellement une adresse et des signaux de contrôle destinés à l'écran.

— La broche 3 (LPSTB, pour *Light Pen StroBe*, en entrée uniquement), grâce à une impulsion du stylo lumineux, indique au 6845 de stocker l'adresse actuelle de la RAM graphique dans le registre du stylo. Le microprocesseur lit ce registre et détermine ainsi la position du stylo à l'écran.

7.2.3 Les registres internes du 6845

Description.- Le 6845 possède dix-huit registres internes, repérés par un **index** (ou numéro), de 0 à 17 :

— Le registre d'index 0 (`Horizontal Total`, en écriture uniquement) indique le nombre de caractères par ligne. Le compteur de caractères interne au 6845 commence une nouvelle ligne lorsque ce nombre est atteint.

Si on y place 97, au 97-ième caractère (HSYNC comptant également comme caractère), une nouvelle ligne commence.

— Le registre d'index 1 (`Horizontal Displayed`, en écriture uniquement) indique le nombre de caractères visibles par ligne. On ne compte pas HSYNC en particulier. La valeur de ce registre doit être inférieure à celle du registre précédent.

Si on y place 80, chaque ligne affiche 80 caractères.

— Le registre d'index 2 (`HSYNC Position`, en écriture uniquement) contrôle la position sur la ligne à laquelle le 6845 doit activer le signal HS. Il contient la position du caractère concerné.

Si on y place 82, l'activation de HS arrive avec le 83-ième caractère de la ligne.

Dans notre exemple, une ligne comprend donc 80 caractères visibles, un caractère invisible (bord noir de l'écran) et la durée de 15 caractères pour le rebroussement horizontal.

— Le registre d'index 3 (`SYNC Width`, en écriture seulement) détermine la durée de l'impulsion HS en nombre de caractères. Ce registre doit contenir une valeur comprise entre 1 et 15 caractères pour s'adapter aux divers moniteurs. La valeur 0 signifie qu'il n'y a pas de signal HS.

Dans notre exemple, on a évidemment 15.

— Le registre d'index 4 (`Vertical Total`, en écriture seulement) indique le nombre de lignes de texte, y compris le rebroussement vertical, moins une.

Si on place 25, il y a donc 26 lignes de caractères (dont une non visible).

— Le registre d'index 5 (`Vertical Total Adjust`, en écriture seulement) indique le nombre de lignes additionnelles, nécessaires pour que le signal de synchronisation verticale VS se déroule correctement.

On y place 6, par exemple, si la durée de la synchronisation verticale équivaut à l'affichage de 6 lignes visibles.

— Le registre d'index 6 (`Vertical Displayed`, en écriture seulement) indique le nombre de lignes visibles. Sa valeur est donc inférieure à celle du registre `Vertical Total`.

On a, par exemple, 25 pour afficher 25 lignes de texte.

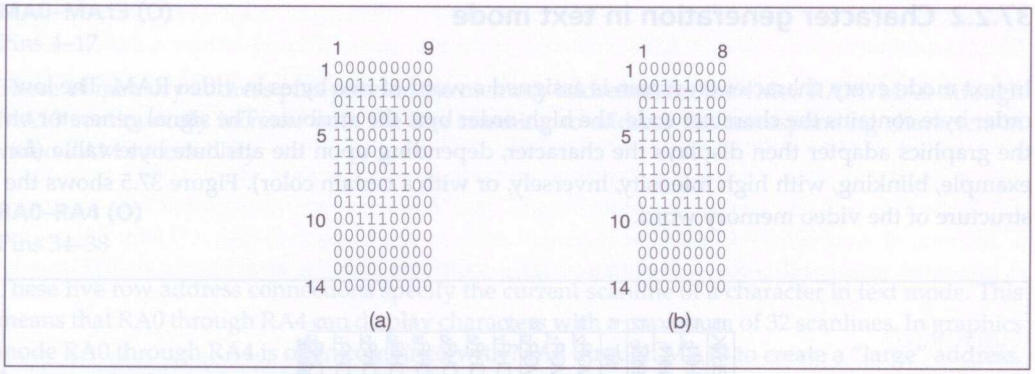
— Le registre d'index 7 (`VSYNC position`, en écriture seulement) indique la ligne à laquelle débute le rebroussement vertical, et donc le moment de l'émission d'un signal VS.

On a, par exemple, 25 dans notre exemple.

— Le registre d'index 8 (`Interlace Mode`, en écriture seulement) indique le mode d'affichage. On utilise seulement les deux bits de poids inférieurs de la façon suivante :

bit 1	bit 0	Mode
0	0	non entrelacé
1	0	non entrelacé
0	1	entrelacé
1	1	entrelacé/vidéo

La figure ci-dessous montre l'affichage du caractère 'O' (en mode non entrelacé) dans chacun des deux autres modes.



Pixel pattern O in text mode: (a) on screen, (b) in character ROM.

FIGURE 7.7 – Affichage du caractère 'O'

— Dans le **mode entrelacé** (figure b), le signal VS est retardé d'une demi-période de ligne de texte et les deux passages du faisceau d'électrons écrivent la même information. Ceci renforce la lisibilité du caractère mais le caractère n'a pas une meilleure résolution.

— Dans le **mode entrelacé/vidéo** (figure c), le signal VS est également retardé d'une demi-période de ligne de texte mais les deux passages du faisceau d'électrons écrivent des informations différentes. Lors du premier passage, il s'agit des lignes impaires ; lors du second passage des lignes paires. La lisibilité du caractère n'est pas améliorée mais le moniteur assure une meilleure résolution pour une même largeur de bande.

— Le registre d'index 9 (`Max Scanline`, en écriture seulement) indique le nombre de lignes par caractères (moins une), limité à 32.

On place, par exemple, 13 pour un mode texte à 14 lignes par caractère.

— Le registre d'index 10 (`Cursor Start`, en écriture seulement) indique la première ligne du curseur et son comportement. Les bits 0 à 5 spécifient la ligne de début et les bits 6 et 7 son comportement :

bit 6	bit 5	Comportement du curseur
0	0	non clignotant
0	1	non affiché
1	0	clignotant avec un taux 16*rafraichissement
1	1	clignotant avec un taux 32*rafraichissement

Le **taux de rafraîchissemnt** est la durée entre deux signaux VS.

On place, par exemple, 11 pour que le curseur débute à la ligne 11 et ne clignote pas.

— Le registre d'index 11 (`Cursor End`, en écriture seulement) indique la dernière ligne du curseur.

On place, par exemple, 12, ce qui donne un curseur de deux lignes.

— Les registre d'index 12 (`Start Address (High)`) et d'index 13 (`Start Address (Low)`), en écriture seulement, donnent l'adresse de `MA0-MA13` après un rebroussement vertical. Ceci permet de définir plusieurs **pages** dans la mémoire graphique.

Si on place, par exemple 0, la page active débute au décalement 0 de la mémoire graphique.

— Les registres d'index 14 et 15 (`Cursor High` et `Cursor Low`, en lecture et écriture), contenant 14 bits (8 bits pour l'inférieur et 6 bits pour le supérieur), définissent la position du curseur dans la mémoire graphique. Lors d'un changement de page, on doit changer la position du curseur. Lorsque le 6845 détecte que l'adresse mémoire en cours MA0–MA13 coïncide avec l'entrée dans ce couple de registre, il affiche le curseur à l'écran et active le signal `CURSOR`.

La position du curseur n'est pas spécifiée par un numéro de ligne et un numéro de colonne mais par un numéro d'emplacement compris entre 0 et une certaine valeur (par exemple 1999 pour 25 lignes de 80 caractères). Pour calculer ce numéro, il suffit de multiplier par le nombre de caractères par ligne (par exemple 80) le numéro de ligne et d'additionner à ce produit le numéro de colonne.

Par exemple si on a 0, le curseur est situé au début de la mémoire graphique, c'est-à-dire au coin supérieur gauche de l'écran.

— Les registres d'index 16 et 17 donnent la position du crayon lumineux (partie haute et basse de l'adresse respectivement), en lecture uniquement.

Accès aux registres internes.- Le MC6845 possède deux ports, appelés **port d'index** et **port de données**. Lorsqu'on veut écrire une valeur dans l'un des 18 registres, on écrit tout d'abord l'index du registre (0 à 17) sur le port d'index. On écrit ensuite sur le port des données la valeur à transférer dans le registre voulu. Le 6845 transfère alors cette valeur dans le registre spécifié.

7.3 La carte graphique MDA

L'IBM-PC a été livré successivement avec des cartes graphiques de plus en plus performantes. Comme toujours pour le PC, il y a compatibilité ascendante. Les premiers PC et XT étaient équipés de la carte graphique monochrome **MDA** (pour *Monochrome Display Adapter*) d'IBM, pour l'utilisation professionnelle. L'affichage est de 25 lignes de 80 colonnes monochrome, avec une matrice de 9×14. Elle correspond au **mode 7** du BIOS tel que nous l'avons vu lorsque nous avons étudier la programmation avec le BIOS.

Décimal	Hexadécimal	Caractère	Décimal	Hexadécimal	Caractère	Décimal	Hexadécimal	Caractère	Décimal	Hexadécimal	Caractère
0	00		32	20		64	40	@	96	60	`
1	01	☺	33	21	!	65	41	A	97	61	a
2	02	●	34	22	"	66	42	B	98	62	b
3	03	♥	35	23	#	67	43	C	99	63	c
4	04	♦	36	24	$	68	44	D	100	64	d
5	05	♣	37	25	%	69	45	E	101	65	e
6	06	♠	38	26	&	70	46	F	102	66	f
7	07	•	39	27	'	71	47	G	103	67	g
8	08	◘	40	28	(	72	48	H	104	68	h
9	09	○	41	29	)	73	49	I	105	69	i
10	0A	◙	42	2A	*	74	4A	J	106	6A	j
11	0B	♂	43	2B	+	75	4B	K	107	6B	k
12	0C	♀	44	2C	,	76	4C	L	108	6C	l
13	0D	♪	45	2D	-	77	4D	M	109	6D	m
14	0E	♫	46	2E	.	78	4E	N	110	6E	n
15	0F	☼	47	2F	/	79	4F	O	111	6F	o
16	10	►	48	30	0	80	50	P	112	70	p
17	11	◄	49	31	1	81	51	Q	113	71	q
18	12	↕	50	32	2	82	52	R	114	72	r
19	13	‼	51	33	3	83	53	S	115	73	s
20	14	¶	52	34	4	84	54	T	116	74	t
21	15	§	53	35	5	85	55	U	117	75	u
22	16	▬	54	36	6	86	56	V	118	76	v
23	17	↨	55	37	7	87	57	W	119	77	w
24	18	↑	56	38	8	88	58	X	120	78	x
25	19	↓	57	39	9	89	59	Y	121	79	y
26	1A	→	58	3A	:	90	5A	Z	122	7A	z
27	1B	←	59	3B	;	91	5B	[	123	7B	{
28	1C	∟	60	3C	<	92	5C	\	124	7C	|
29	1D	↔	61	3D	=	93	5D	]	125	7D	}
30	1E	▲	62	3E	>	94	5E	^	126	7E	~
31	1F	▼	63	3F	?	95	5F	_	127	7F	⌂

FIGURE 7.8 – Caractères affichés (1)

7.3.1 Caractères affichés

Jeu de caractères.- L'affichage du texte sur un PC travaille avec un jeu de caractères ASCII étendu, qui comprend les 256 caractères que montrent les figures 7.8 et 7.9. Ces caractères sont numérotés de 0 à 255.

Décimal	Hexadécimal	Caractère	Décimal	Hexadécimal	Caractère	Décimal	Hexadécimal	Caractère	Décimal	Hexadécimal	Caractère
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	▒	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	η
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

FIGURE 7.9 – Caractères affichés (2)

Attributs d'un caractère.- Un caractère nécessite un octet pour le déterminer, puisqu'il y en a 256. Un second octet, dit **octet d'attribut**, indique, pour chaque emplacement de caractère à l'écran, le mode d'affichage (souligné, clignotant, inversé, ...) du caractère à afficher. Cet octet d'attribut, défini par IBM en même temps à la fois pour la carte MDA et la carte CGA, a le format suivant :

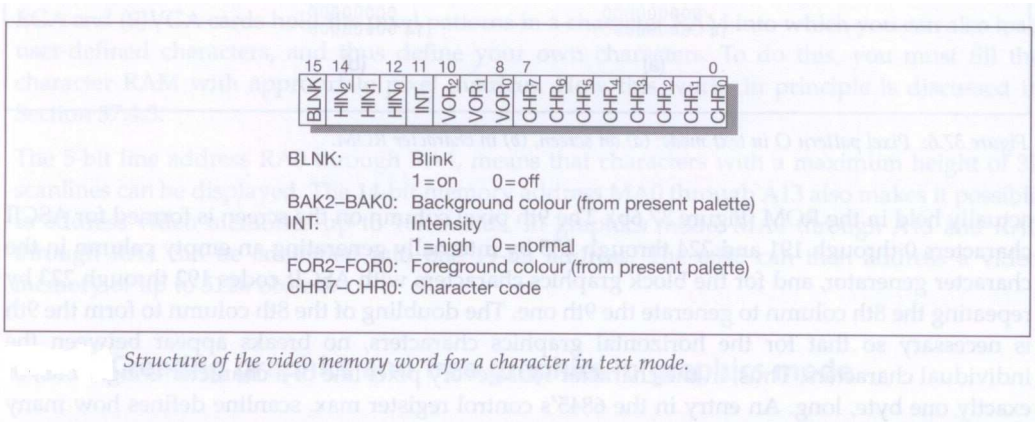


FIGURE 7.10 – Structure d'un élément de la mémoire graphique

— Les bits 0 à 7 indiquent le **code** du caractère. Ce code sert d'index dans la **table des caractères** se trouvant en ROM caractères.

— Le bit 15 (`BLNK`, pour *BLiNK*), lorsqu'il est à 1, indique que le caractère doit clignoter. Suivant la carte graphique et le constructeur, le caractère clignote à une fréquence comprise entre 1 Hz et 3 Hz.

— Les bits 12 à 14 (BAK_0, BAK_1, BAK_2, pour *BACKground*) spécifient la couleur de fond du caractère. Huit couleurs différentes sont donc possibles. La couleur dépend de la carte graphique, de la palette de couleur sélectionnée et du fait qu'il s'agisse d'un moniteur monochrome ou couleur.

— Le bit 11 (`INT` pour *INTensity*) fait apparaître un caractère plus brillant (plus intense) s'il est à 1.

— Les bits 8 à 10 (FOR_0, FOR_1, FOR_2 pour *FOReground*) spécifient la couleur de premier plan du caractère.

En ce qui concerne la carte MDA, ces bits peuvent être définis librement mais seules certaines combinaisons de bits pour les couleurs de premier plan et de fond produiront un affichage de caractères valables :

7	6	5	4	3	2	1	0	Effet
	0	0	0		0	0	0	Pas de caractère (noir sur noir, pour les mots de passe)
	0	0	0		0	0	1	Caractère souligné
	0	0	0		1	1	1	Caractère normal (blanc sur noir)
	1	1	1		0	0	0	Caractère inversé (noir sur blanc)
	1	1	1		1	1	1	Contour du caractère blanc (blanc sur blanc)

7.3.2 Structure de la mémoire graphique

Emplacement et capacité de la mémoire graphique.- Puisque chaque caractère nécessite deux octets, pour un affichage de 25 lignes de 80 caractères, on a besoin d'une mémoire de 4 000 octets.

La mémoire graphique de la carte MDA n'occupe pas les 128 KiO possibles de la VDR. Elle commence à l'adresse `B000:000h` et occupe 4 KiO (4 096 octets).

Comme seuls 4 000 octets sont nécessaires pour le stockage des caractères, les 96 octets en fin de la mémoire graphique sont inutilisés et sont donc à la disposition du programmeur.

Adresse dans la mémoire graphique.- La mémoire graphique est vue comme un tableau linéaire. Le premier mot concerne le caractère du coin supérieur gauche, c'est-à-dire de la ligne 1, colonne 1, le second mot celui de la ligne 1, colonne 2, et ainsi de suite.

L'adresse du mot mémoire pour le caractère de la ligne i, colonne j est donnée par l'équation suivante :

$$adresse = sv + 2 * ncpl * i + 2 * j$$

où sv (pour *Segment Video*) désigne l'adresse de début de la mémoire graphique (donc B000h) et $ncpal$ le nombre de caractères par ligne. Les variables i et j commencent à 0.

7.3.3 La table des caractères

Motif.- La carte MDA offre un affichage de texte de très bonne qualité, grâce à l'utilisation d'une matrice de 9×14 pour l'affichage des caractères, c'est-à-dire que chaque caractère est représenté par un bloc de pixels de 14 lignes et 9 colonnes.

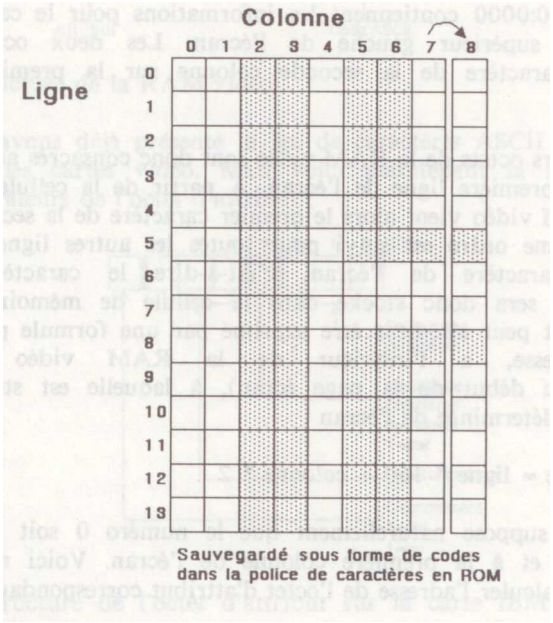


FIGURE 7.11 – Motif d'un caractère

Le format de cette matrice est d'autant plus extraordinaire qu'un générateur de caractères contenant les modèles de bits de chaque caractère ne permet pas de produire des caractères d'une largeur de plus de 8 pixels (figure 7.11). C'est pourquoi un circuit a été implanté sur cette carte qui permet de copier tout simplement le huitième pixel dans le neuvième pixel pour tous les caractères dont le code ASCII étendu est compris entre B0h et DFh, c'est-à-dire essentiellement les caractères de cadre.

Disposition.- Dans la ROM des caractères, les matrices de pixels pour les 256 caractères sont arrangées de telle façon que le code du caractère forme un premier index, indiquant le début du caractère en ROM. Le numéro de la ligne de pixels pour un caractère est un second index, ou décalage, qui détermine la ligne de pixel (un octet) du caractère.

Capacité.- La capacité mémoire nécessaire pour les motifs du jeu de caractères est donc de $256 \times 14 = 3\,584$ octets, soit d'un peu moins de 4 KiO. Pour une raison assez mystérieuse, cette ROM comporte toutefois non pas 4 KiO mais 8 KiO, de sorte que 4 KiO sont inutilisés.

7.3.4 Ports associés

La carte monochrome MDA utilise quatre ports (ou registres) d'entrées-sorties avec les fonctions suivantes :

Adresse	Fonction
3B4h	Registre d'index du 6845
3B5h	Registre des données du 6845
3B8h	Registre de contrôle
3BAh	Registre d'état

Les deux premiers ports sont utilisés pour les communications avec le contrôleur vidéo 6845 et les deux derniers ports permettent la modification de certaines caractéristiques de l'écran.

Le **registre de contrôle**, qui figure sur le port 3B8h sur l'IBM-PC, sert à contrôler différentes fonctions de la carte. On ne peut qu'y écrire mais pas le lire. Sa structure est la suivante :

7	6	5	4	3	2	1	0
X	X		X		X	X	1

Seuls les bits 0, 3 et 5 jouent un rôle :

- Le bit 0 commande la résolution de l'écran. Bien que la carte de supporte qu'une seule résolution (25 lignes sur 80 colonnes), ce bit doit être positionné à 1 lors de l'initilisation du système général, faute de quoi, l'ordinateur sombrerait dans une boucle d'attente perpétuelle.
- Le bit 3 commande la mise en place d'une image visible sur l'écran. S'il est positionné à 0, l'écran apparaît tout en noir et le curseur clignotant disparaît. S'il est à nouveau positionné à 1, l'image revient sur l'écran.
- Le bit 5 autorise ou interdit le clignotement des caractères dont le bit 7 de l'octet d'attribut est positionné à 1. S'il vaut zéro, ces caractères ne clignoteront pas mais seont affichés sur une couleur de fond particulièrement claire, c'est-à-dire que le bit 7 de l'octet d'attribut fonctionnera alors comme bit d'intensité pour le fond.

La valeur usuelle de ce registre est 29h, c'est-à-dire que les trois bits significatifs ont 1 comme valeur.

Le **registre d'état**, qui figure sur le port 3BAh pour un IBM-PC, sert à spécifier l'état de la carte. À l'inverse du registre de contrôle, ce registre peut être lu mais on ne peut pas y écrire. Sa structure est la suivante :

7	6	5	4	3	2	1	0
1	1	1	1		1	1	

Seuls les bits 0 et 3 de ce registre sont utilisés mais tous les autres bits doivent contenir la valeur 1 :

- Le bit 0 indique qu'un signal de synchronisation horizontale est actuellement en train d'être envoyé à l'écran.
- Le bit 3 contient la valeur du point écran (*pixel*) sur lequel est actuellement placé le faisceau. Un 1 signale que le point doit être visible sur l'écran, 0 que l'écran doit rester noir dans cet emplacement.

7.3.5 Initialisation du contrôleur graphique

Pour la carte MDA, le contrôleur graphique MC6845 est initialisé de la façon suivante :

Registre	Signification	Valeur
0	Nombre total de caractères	97
1	Nombre de caractères visibles	80
2	Position HS	82
3	Largeur HS	15
4	Nombre total de lignes	25
5	Ajustement vertical	6
6	Nombre de lignes visibles	25
7	Position VS	25
8	Mode d'entrelacement	2
9	Nombre de lignes d'un motif	13
10	Début curseur	11
11	Fin curseur	12
12	Début mémoire (haut)	0
13	Début mémoire (bas)	0
14	Adresse curseur (haut)	0
15	Adresse curseur (bas)	0

7.4 La carte graphique CGA

Pour l'utilisation à la maison, les premiers PC étaient livrés avec la carte **CGA** (pour *Color Graphics Adapter*) d'IBM, avec une résolution graphique de 640×200. Elle permet, comme la carte MDA, d'afficher du texte avec le jeu de caractères standard de l'IBM PC, mais elle offre en plus différents modes d'affichage graphique (ou point par point, APA pour *All Points Adressable*).

Il existe un mode texte de 25 lignes de 40 caractères de matrice 8×8 en 16 couleurs (**mode 0** du BIOS), ce qui donne une résolution assez mauvaise, et un mode texte de 25 lignes de 80 caractères en 8 couleurs (**mode 3** du BIOS). En mode graphique, deux types de résolution sont supportés : la *moyenne résolution* qui fournit une division de l'écran de 200 lignes de 320 points avec 4 couleurs ; la *haute résolution* qui permet une représentation de l'écran sur 200 lignes de 600 points en 2 couleurs chacune. La *basse résolution*, non supportée par la ROM, fournit une représentation de l'écran de 100 lignes de 160 points chacune en 16 couleurs.

7.4.1 Caractères affichés

Jeu de caractères.- Le jeu de caractères est le même que pour la carte MDA.

Attributs d'un caractère.- L'octet d'attribut (figure 7.10) présente, par rapport à la carte MDA, une structure légèrement différente :

— Les quatre bits inférieurs (INT, FOR_0, FOR_1 et FOR_2) spécifient l'une des 16 couleurs possibles pour le premier plan du caractère :

Déc.	Hexa.	Binaire	Couleur
0	0	0000	Noir
1	1	0001	Bleu
2	2	0010	Vert
3	3	0011	Bleu océan (Turquoise)
4	4	0100	Rouge
5	5	0101	Violet (Magenta)
6	6	0110	Marron
7	7	0111	Gris clair
8	8	1000	Gris foncé
9	9	1001	Bleu clair
10	A	1010	Vert clair
11	B	1011	Bleu marine clair
12	C	1100	Rouge clair
13	D	1101	Violet clair
14	E	1110	Jaune
15	F	1111	Blanc

— La signification des quatre bits supérieurs (BLNK, BAK_0, BAK_1 et BAK_2) varie suivant que le clignotement est ou non activé. Si c'est le cas, les bits 4 à 6 spécifient une des 8 premières couleurs de la palette de couleur pour le fond alors que l'état du bit 7 indique que le caractère doit clignoter ou non. Si le clignotement est désactivé, les bits 4 à 7 spécifient une des 16 couleurs possibles pour le fond.

Lorsque la carte émule une carte monochrome, le caractère d'attribut a la même signification que sur la carte MDA, si ce n'est qu'il n'est pas possible d'afficher des caractères soulignés.

7.4.2 Affichage point à point

Dans le cas d'un mode graphique, il n'y a pas d'octet d'attribut car chaque point écran est défini directement par sa couleur et non plus par l'appartenance à tel ou tel caractère d'un jeu de caractères.

Si on veut afficher un caractère en mode graphique, on doit dessiner celui-ci point à point.

En résolution de 320 points sur 200, n'importe laquelle des 16 couleurs disponibles peut être utilisée comme couleur de fond. La couleur de premier plan doit par contre être l'une des trois couleurs d'une palette prévue par la carte graphique. Deux palettes sont disponibles, la première avec les couleurs turquoise, magenta et blanc, la seconde avec les couleurs vert, rouge et jaune.

7.4.3 Structure de la mémoire graphique

Notion de page.- La capacité mémoire graphique de la plupart des cartes graphiques est beaucoup plus importante que ce qui est nécessaire pour une seule page de texte. La mémoire graphique est donc souvent divisée en **pages**, quantité de mémoire nécessaire pour une écran texte. La taille de la page dépend de la résolution. Le nombre maximum de pages dépend de la résolution et de la capacité de la mémoire graphique.

Emplacement et capacité de la mémoire graphique.- La mémoire graphique de la carte CGA commence à l'adresse `B800:000h` et occupe 16 KiO. Les mémoires graphiques des cartes MDA et CGA ne se chevauchent pas de sorte qu'elles peuvent être utilisées en parallèle sur l'ordinateur, pour sortir chacune son image sur un moniteur particulier.

La carte permet donc de sauvegarder quatre pages de texte de 80*25 caractères, commençant respectivement aux adresses `B800:0000h`, `B800:1000h`, `B800:2000h` et `B800:3000h`.

Elle permet également de sauvegarder huit pages de texte de 40*25 caractères, commençant respectivement aux adresses `B800:0000h`, `B800:0800h`, `B800:1000h`, `B800:1800h`, `B800:2000h`, `B800:2800h`, `B800:3000h` et `B800:3800h`.

Pour une page graphique, la totalité des 16 KiO est nécessaire. Il n'y a donc qu'une seule page possible.

Adresse dans la mémoire graphique.- L'adresse du mot mémoire pour le caractère de la ligne i, colonne j de la page k est donnée par l'équation suivante :

$$adresse = sv + tp * k + 2 * ncpl * i + 2 * j$$

où sv (pour *Segment Video*) désigne l'adresse de début de la mémoire graphique (donc B800h), tp désigne la taille d'une page et $ncpal$ le nombre de caractères par ligne. Les variables i, j et k commencent à 0.

7.4.4 La table des caractères

La carte CGA offre un affichage de texte avec une matrice de 8×8 pour l'affichage des caractères, ce qui entraîne une qualité d'affichage nettement inférieure à celle de la carte MDA.

7.4.5 Ports associés

La carte couleur CGA utilise huit ports d'entrées-sorties, ou registres, avec les fonctions suivantes (le numéro de port étant celui de l'IBM-PC) :

Adresse	Fonction
3D0h	Registre d'index du 6845
3D1h	Registre des données du 6845
3D8h	Registre de sélection de mode
3D9h	Sélection de la couleur
3DAh	Registre d'état
3DBh	Effacement du crayon lumineux
3DCh	Positionnement du crayon lumineux

Les deux premiers ports sont utilisés pour les communications avec le contrôleur vidéo 6845. le port 3D8h définit le mode (texte ou graphique), le port 3D9h les couleurs et la palette utilisée et le port 3DAh l'état.

— Le port 3D8h, nommé **registre de sélection de mode**, peut être comparé au registre de commande de la carte MDA. On peut écrire dans ce registre mais non le lire. Il est relié à 6 bits de positionnement gérant l'utilisation de l'écran :

Bit	Fonction	Bit = 0	Bit = 1
0	Vitesse du signal caractère	lent (40 c)	rapide (80 c)
1	Mode texte ou graphique	texte	graphique
2	Couleur ou N/B	couleur	noir/blanc
3	Affichage vidéo	pas d'affichage	normal
4	Moyenne et haute résolution	320 * 200 (MR)	640 * 200 (HR)
5	Attribut d'octet	Clignotant	Fond clair

— Le bit 1 permet de passer du mode de texte en mode graphique 320*200 points ou vice-versa.
— Le bit 2 est intéressant pour exploiter une carte CGA avec un moniteur monochrome. Si ce bit vaut 1, la production d'un signal de couleur par le 6845 n'est pas permise et seules les images en noir et blanc pourront être réalisées.
— Il est conseillé de désactiver le bit 3 chaque fois que l'on change de mode d'affichage, pour éviter que des signaux « anarchiques » ne soient envoyés au moniteur, ce qui risquerait de l'endommager.

Ce registre permet donc de sélectionner les différents modes de texte ou modes graphiques ainsi que l'affichage couleur ou monochrome :

Bit 4	Bit 2	Bit 1	Bit 0	Résultat
0	1	0	0	40*25 texte monochrome
0	0	0	0	40*25 texte couleur
0	1	0	1	80*25 texte monochrome
0	0	0	1	80*25 texte couleur
0	1	1	0	320*200 graphique monochrome
0	0	1	0	320*200 graphique couleur
1	1	1	0	640*200 graphique monochrome

A priori une quelconque combinaison des valeurs de ces bits est acceptable, toutefois, de façon matérielle l'IBM PC ne traite que certaines d'entre elles. Par exemple lorsque le bit 4 est positionné à 1 (mode haute résolution graphique), le bit 2 doit toujours être positionné à 1 (noir et blanc) et le bit 1 à 1 pour sélectionner le mode graphique.

— Le **registre de sélection de couleurs** 3D9h spécifie la couleur de la bordure en mode texte puisque, dans ce cas, le texte ne remplit pas tout l'écran. En mode graphique, il permet la sélection des palettes.

— Le **registre d'état** 3DAh fournit l'information de retour de la carte CGA vers le programme. C'est lui qui indique lorsqu'il est possible de lire ou d'écrire des données dans la mémoire tampon graphique. Il signale également si un crayon lumineux est attaché au système. Il peut être lu mais on ne peut pas y écrire. Sa structure est la suivante :

7	6	5	4	3	2	1	0
1	1	1	1				

— Le bit le plus intéressant de ce registre est le bit 0, qui vaut 1 chaque fois que le 6845 est en train d'émettre un signal de synchronisation horizontale en direction du moniteur. Il indique, pour une carte CGA, que l'on peut alors écrire dans la mémoire graphique.

En mode texte haute résolution (80 colonnes de 25 lignes), un effet de « neige » peut se produire à l'écran si l'écriture des données dans la mémoire graphique ne s'effectue pas au moment adéquat.

Ce problème provient du fait que le 6845 doit accéder en permanence à la mémoire graphique pour en lire le contenu et pouvoir ainsi construire et rafraîchir une image. Donc, lorsqu'un programme essaie de transférer des données dans la mémoire graphique, cela peut causer des difficultés si le 6845 veut accéder à celle-ci au même moment. Le résultat d'un tel conflit se traduit pas un tremblotement de l'écran. Pour éviter ce problème, il faut n'accéder à la mémoire graphique que lorsque le 6845 est au repos. Or c'est justement le cas chaque fois qu'un signal de synchronisation horizontale est envoyé à l'écran car il faut un certain temps (même si ce délai est très bref selon nos critères humains) au faisceau d'électrons pour exécuter cet ordre de synchronisation. C'est pourquoi, avant tout accès à la mémoire graphique sur une carte CGA, il convient de lire le registre d'état et de répéter cette lecture tant que le bit ne vaut pas 1. Dès que c'est le cas, on peut alors transférer un ou, au maximum, deux octets dans la mémoire graphique.

Ce délai est inutile sur la carte MDA qui est dotée d'une logique électronique particulière et de circuits de RAM particulièrement rapides.

— Le bit 1 est égal à 1 lorsque le tracé en couleur est impossible.

— Le bit 2 est égal à 0 lorsque le tracé couleur est activé et à 1 lorsqu'il est désactivé.

— Le bit 3 est égal à 1 lors du retour du faisceau de balayage vertical.

— Le **registre de sélection de couleurs** 3D9h est ouvert en écriture mais pas en lecture. Sa structure est la suivante :

7	6	5	4	3	2	1	0
1	1						

— La signification des bits de ce registre dépend du mode d'affichage. Dans les modes de texte, les 4 bits inférieurs sélectionnent la couleur du cadre de l'écran, en définissant le numéro de l'une des 16 couleurs disponibles. En mode graphique 320 points sur 200, ils indiquent en outre la couleur de tous les points écran représentés par la combinaison de bits 00b (on parle alors également de couleur de fond).

— Le bit 4 est égal à 1 pour spécifier l'intensité de la couleur de fond en mode texte.

— La palette pour le mode graphique 320*200 points est sélectionnée à travers le bit 5. Si ce bit vaut 1, la première palette de couleur (turquoise, violet, blanc) sera sélectionnée, sinon ce sera la seconde (vert, jaune, rouge).

7.4.6 Initialisation du contrôleur graphique

Pour la carte CGA, le contrôleur graphique MC6845 est initialisé de la façon suivante :

Registre	Signification	Texte 40x25	Texte 80x25	Graphique
0	Nombre total de caractères	56	113	56
1	Nombre de caractères visibles	40	80	40
2	Position HS	45	90	45
3	Largeur HS	10	10	10
4	Nombre total de lignes	31	31	127
5	Ajustement vertical	6	6	6
6	Nombre de lignes visibles	25	25	100
7	Position VS	28	28	112
8	Mode d'entrelacement	2	2	2
9	Nombre de lignes d'un motif	7	7	1
10	Début curseur	6	6	6
11	Fin curseur	7	7	7
12	Début mémoire (haut)	0	0	0
13	Début mémoire (bas)	0	0	0
14	Adresse curseur (haut)	0	0	0
15	Adresse curseur (bas)	0	0	0

On peut peut-être s'étonner du nombre de lignes visibles, à savoir 100, en mode graphique. Cela est dû au mode d'entrelacement. La constitution des images est obtenue après un double balayage de l'écran : l'affichage des lignes de numéro pair se fait lors de la descente, l'affichage des lignes de de numéro impair s'effectue lors de la remontée.

7.5 Commentaire du BIOS : l'ISR de l'interruption 10h

7.5.1 Zone de communication vidéo du BIOS

Nous avons vu au chapitre 3 qu'une partie de la zone de communication du BIOS concerne l'affichage, celle comprise entre les adresses `40:49h` et `40:66h`.

7.5.2 Mémoire graphique

Une autre partie de la mémoire vive est ce qui s'appelle la **mémoire graphique**, en fait celle de la carte CGA :

```
                    222    ;-----------------------------------
                    223    ;     VIDEO DISPLAY BUFFER          :
                    224    ;-----------------------------------
----                225    VIDEO_RAM     SEGMENT AT 0B800H
0000                226    REGEN   LABEL BYTE
0000                227    REGENW  LABEL WORD
0000 (16384         228            DB    16384 DUP(?)
     ??
     )
----                229    VIDEO_RAM     ENDS
```

constituée de 16 384 octets, soit 16 KiO, située à l'adresse B800h.

7.5.3 Détermination de la fonction

Le début du code de la routine de service de l'interruption 10h est clairement indiqué à partir de la ligne 3202 :

```
                    3202
                    3203    ;---- INT 10 ----------------------------------------------------
                    3204    ; VIDEO_IO                                                      :
                    3205    ;       THESE ROUTINES PROVIDE THE CRT INTERFACE                :
                    3206    ;       THE FOLLOWING FUNCTIONS ARE PROVIDED:                   :
                    3207    ;       (AH)=0  SET MODE (AL) CONTAINS MODE VALUE               :
                    3208    ;               (AL)=0 40X25 BW (POWER ON DEFAULT)              :
                    3209    ;               (AL)=1 40X25 COLOR                              :
                    3210    ;               (AL)=2  80X25 BW                                :
                    3211    ;               (AL)=3  80X25 COLOR                             :
                    3212    ;               GRAPHICS MODES                                  :
                    3213    ;               (AL)=4  320X200 COLOR                           :
                    3214    ;               (AL)=5  320X200 BW                              :
                    3215    ;               (AL)=6  640X200 BW                              :
                    3216    ;               CRT MODE=7 80X25 CARD (USED INTERNAL TO VIDEO ONLY):
                    3217    ;               *** NOTE BW MODES OPERATE SAME AS COLOR MODES, BUT :
                    3218    ;                   COLOR BURST IS NOT ENABLED                  :
                    3219    ;       (AH)=1  SET CURSOR TYPE                                 :
------------------
                    3331            ASSUME CS:CODE,DS:DATA,ES:VIDEO_RAM
F045                3332            ORG    0F045H
F045                3333    M1      LABEL  WORD                    ; TABLE OF ROUTINES WITHIN VIDEO I/O
F045 FCF0           3334            DW     OFFSET  SET_MODE
F047 CDF1           3335            DW     OFFSET  SET_CTYPE
F049 EEF1           3336            DW     OFFSET  SET_CPOS
F04B 39F2           3337            DW     OFFSET  READ_CURSOR
F04D 9CF7           3338            DW     OFFSET  READ_LPEN
F04F 17F2           3339            DW     OFFSET  ACT_DISP_PAGE
F051 96F2           3340            DW     OFFSET  SCROLL_UP
F053 38F3           3341            DW     OFFSET  SCROLL_DOWN
F055 74F3           3342            DW     OFFSET  READ_AC_CURRENT
F057 B9F3           3343            DW     OFFSET  WRITE_AC_CURRENT
F059 ECF3           3344            DW     OFFSET  WRITE_C_CURRENT
F05B 4EF2           3345            DW     OFFSET  SET_COLOR
F05D 2FF4           3346            DW     OFFSET  WRITE_DOT
F05F 1EF4           3347            DW     OFFSET  READ_DOT
F061 18F7           3348            DW     OFFSET  VIDEO_STATE
F063 74F2           3349            DW     OFFSET  VIDEO_STATE
  0020              3350    M1L     EQU    $-M1
                    3351
F065                3352            ORG    0F065H
F065                3353    VIDEO_IO        PROC    NEAR
F065 FB             3354            STI                            ; INTERRUPTS BACK ON
F066 FC             3355            CLD                            ; SET DIRECTION FORWARD
F067 06             3356            PUSH    ES
F068 1E             3357            PUSH    DS                     ; SAVE SEGMENT REGISTERS
F069 52             3358            PUSH    DX
F06A 51             3359            PUSH    CX
F06B 53             3360            PUSH    BX
F06C 56             3361            PUSH    SI
F06D 57             3362            PUSH    DI
F06E 50             3363            PUSH    AX
F06F 8AC4           3364            MOV     AL,AH                  ; GET INTO LOW BYTE
F071 32E4           3365            XOR     AH,AH                  ; ZERO TO HIGH BYTE
F073 01E0           3366            SAL     AX,1                   ; *2 FOR TABLE LOOKUP
F075 8BF0           3367            MOV     SI,AX                  ; PUT INTO SI FOR BRANCH
F077 3D2000         3368            CMP     AX,M1L                 ; TEST FOR WITHIN RANGE
F07A 7204           3369            JB      M2                     ; BRANCH AROUND BRANCH
F07C 5B             3370            POP     AX                     ; THROW AWAY THE PARAMETER
F07D E94501         3371            JMP     VIDEO_RETURN           ; DO NOTHING IF NOT IN RANGE
F080                3372    M2:
F080 E8D609         3373            CALL    DDS
F083 B880B8         3374            MOV     AX,0B80                ; SEGMENT FOR COLOR CARD
F086 8B3E1000       3375            MOV     DI,EQUIP_FLAG          ; GET EQUIPMENT SETTING
F08A 81E73000       3376            AND     DI,30H                 ; ISOLATE CRT SWITCHES
F08E 83FF30         3377            CMP     DI,30H                 ; IS SETTING FOR BW CARD?
F091 7502           3378            JNE     M3
F093 B4B0           3379            MOV     AH,0B0H                ; SEGMENT FOR BW CARD
F095                3380    M3:
```

```
F095 8EC0          3381            MOV     ES,AX                   ; SET UP TO POINT AT VIDEO RAM AREAS
F097 58            3382            POP     AX                      ; RECOVER VALUE
F098 8A264900      3383            MOV     AH,CRT_MODE             ; GET CURRENT MODE INTO AH
F09C 2EFFA445F0    3384            JMP     WORD PRT CS:[SI+OFFSET M1]
                   3385    VIDEO_IO        ENDP
```

Commentaires.- 1^o) Nous n'avons pas repris ici tout le commentaire qui débute cette routine, consistant à expliciter les diverses fonctions de celle-ci. Nous les replacerons en début de chaque fonction.

- 2^o) Le segment de code est celui de la ROM résidente, le segment des données correspond à la zone de communication du BIOS, DATA étant défini à la ligne 80 comme nous l'avons déjà vu, le segment supplémentaire est la mémoire graphique (ligne 3331).

- 3^o) Vient ensuite (lignes 3333 à 3349) la définition de la table des adresses des fonctions de la routine, qui sera placée à l'adresse F045h (ligne 3332) et dont la longueur est précisée ligne 3350. Les noms symboliques de ces fonctions sont définis au moment d'écrire le corps de chacune de celles-ci.

- 4^o) Le code de cette routine sera placé à l'adresse F065h de la mémoire vive (ligne 3352), adresse donnée par le BIOS par ailleurs (comme nous le verrons) comme début de la routine de service de l'interruption 10h dans le tableau des vecteurs des interruptions.

- 5^o) Comme toute routine de service d'interruption, elle est considérée comme un sous-programme (proche, ligne 3353). Les interruptions de priorité plus élevée prendront le pas (ligne 3354). Lors des recherches dans les tables on partira de la fin (ligne 3355). Tous les registres que l'on utilisera sont sauvegardés sur la pile (lignes 3356 à 3363).

- 6^o) Le numéro de fonction est placé dans AL (ligne 3363) et on met AH à 0 (ligne 3365) de façon à ce que AX contienne le numéro de fonction. On multiplie ce numéro par 2 (ligne 3366) pour faciliter la recherche dans la table des parties de la routine consacrées à chacune des fonctions, puisque chaque adresse y occupe deux octets. On place le contenu de AX dans SI pour effectuer cette recherche (ligne 3367). Si le contenu de AX est supérieur au nombre de fonctions (lignes 3368 et 3369), il ne s'agit pas d'une fonction valide, donc on ne fait rien, ou plus exactement on termine la routine de service (lignes 3370 et 3371).

Le retour de la routine est défini à la fin de la première sous-routine, celle de la fonction 0 :

```
                   3556    ;---- NORMAL RETURN FROM ALL VIDEO RETURNS
                   3557
F1C5               3558    VIDEO_RETURN:
F1C5 5F            3559            POP     DI
F1C6 5E            3560            POP     SI
F1C7 5B            3561            POP     BX
F1C8               3562    M15:                                    ; VIDEO_RETURN_C
F1C8 59            3563            POP     CX
F1C9 5A            3564            POP     DX
F1CA 1F            3565            POP     DS
F1CB 07            3566            POP     ES                      ; RECOVER SEGMENTS
F1CC CF            3567            IRET                            ; ALL DONE
                   3568    SET_MODE        ENDP
```

Elle consiste à restaurer la valeur des registres utilisés dans la routine.

- 7^o) Sinon le segment des données utilisé dans cette routine de service est celui de la zone de communication du BIOS (ligne 3373, qui renvoie au sous-programme DDS déjà étudié à propos du clavier).

Il s'agit d'une redondance avec la ligne 3331.

- 8^o) On détermine ensuite le type de carte graphique utilisé (monochrome ou couleur) et on initialise le registre de segment supplémentaire avec la mémoire graphique associée.

Pour cela, on place dans AX l'adresse pour la carte couleur (ligne 3374). Si l'interrupteur spécifiant l'équipement (lignes 3375 à 3378) indique qu'il s'agit d'une carte monochrome, on

change la valeur l'octet de poids fort de AX (ligne 3379) pour correspondre à l'adresse de la mémoire graphique d'une telle carte. On initialise ES en conséquence (ligne 3381) et on restaure la valeur de AX (ligne 3382) dont on ne se servira plus. On place le numéro de mode en cours dans AH (ligne 3383) et on se rend au début de la sous-routine correspondant au numéro de fonction (ligne 3384).

7.5.4 Détermination de la position d'un caractère

La sous-routine POSITION détermine, à partir des coordonnées d'un caractère, numéro de colonne dans AL et numéro de ligne dans AH, le décalage dans la mémoire graphique, conformément à la formule :

$$offset = 2 \times (ligne \times nbrcol + col)$$

où *ligne* est le numéro de ligne, *col* le numéro de colonne et *nbrcol* le nombre de colonnes à l'écran. La multiplication par deux provient du fait que chaque caractère occupe deux octets dans la mémoire graphique.

Le début du code de cette fonction commence ligne 3748 :

```
                3748   ;--------------------------------------------------------
                3749   ; POSITION                                              :
                3750   ;       THIS SERVICE ROUTINE CALCULATES THE REGEN       :
                3751   ;       BUFFER ADDRESS OF A CHARACTER IN THE ALPHA MODE :
                3752   ; INPUT                                                 :
                3753   ;       AX = ROW, COLUMN POSITION                       :
                3754   ; OUTPUT                                                :
                3755   ;       AX = OFFSET OF CHAR POSITION IN REGEN BUFFER    :
                3756   ;--------------------------------------------------------
F285            3757   POSITION        PROC    NEAR
F285 53         3758           PUSH    BX                      ; SAVE REGISTER
F286 88D8       3759           MOV     BX,AX
F288 8AC4       3760           MOV     AL,AH                   ; ROWS TO AL
F28A F6264A00   3761           MUL     BYTE PTR CRT_COLS       ; DETERMINE BYTES TO ROW
F28E 32FF       3762           XOR     BH,BH
F290 03C3       3763           ADD     AX,BX                   ; ADD IN COLUMN VALUE
F292 D1E0       3764           SAL     AX,1                    ; * 2 FOR ATTRIBUTE BYTES
F294 5B         3765           POP     BX
F295 C3         3766           RET
                3767   POSITION        ENDP
```

Commentaires.- 1^o) Le contenu du registre BX, qui va être utilisé pour effectuer les calculs, est sauvegardé sur la pile (ligne 3758). Il sera récupéré à la fin de la sous-routine (lignes 2765).

- 2^o) Les coordonnées sont placées dans BX (ligne 2759). Le numéro de ligne est déplacé dans AL (ligne 2760), on le multiplie par le nombre de colonnes (ligne 2761 ; rappelons que celui-ci a été défini dans la zone de communication vidéo du BIOS, ligne 167). On annule le numéro de ligne dans BX (ligne 2762), il n'y reste donc plus que le numéro de colonne, que l'on ajoute à AX (ligne 2763) et on multiplie le tout par deux (ligne 2764).

7.5.5 Affichage d'un caractère à la position du curseur

L'affichage d'un caractère est ce qui nous intéresse le plus et la fonction qui semble la plus compliquée. En fait, puisque tout est déporté vers le CRCT, l'affichage est simple : il suffit de placer le mot constitué du code ASCII et de l'attribut dans la mémoire graphique.

Le début du code de cette fonction commence ligne 4001 :

```
                4001   ;----------------------------------------------
                4002   ; WRITE_AC_CURRENT                            :
                4003   ;       THIS ROUTINE WRITES THE ATTRIBUTE     :
                4004   ;       AND CHARACTER AT THE CURRENT CURSOR   :
                4005   ;       POSITION                              :
                4006   ; INPUT                                       :
                4007   ;       (AH) = CURRENT CRT MODE               :
                4008   ;       (BH) = DISPLAY PAGE                   :
                4009   ;       (CX) = COUNT OF CHARACTERS TO WRITE   :
                4010   ;       (AL) = CHAR TO WRITE                  :
                4011   ;       (BL) = ATTRIBUTE OF CHAR TO WRITE     :
                4012   ;       (DS) = DATA SEGMENT                   :
                4013   ;       (ES) = REGEN SEGMENT                  :
                4014   ; OUTPUT                                      :
                4015   ;       NONE                                  :
                4016   ;----------------------------------------------
F3B9            4017   WRITE_AC_CURRENT        PROC    NEAR
F3B9 80FC04     4018           CMP     AH,4                    ; IS THIS GRAPHICS
F3BC 7208       4019           JC      P6
F3BE 8DFC07     4020           CMP     AH,7                    ; IS THIS BW CARD
F3C1 7403       4021           JE      P6
F3C3 E98201     4022           JMP     GRAPHICS_WRITE
F3C6            4023   P6:                                     ; WRITE_AC_CONTINUE
F3C6 8AE3       4024           MOV     AH,BL                   ; GET ATTRIBUTE TO AH
F3C8 50         4025           PUSH    AX                      ; SAVE ON STACK
F3C9 51         4026           PUSH    CX                      ; SAVE WRITE COUNT
F3CA E8D1FF     4027           CALL    FIND_POSITION
F3CD 8BFB       4028           MOV     DI,BX                   ; ADDRESS TO DI REGISTER
F3CF 59         4029           POP     CX                      ; WRITE COUNT
F3D0 5B         4030           POP     BX                      ; CHARACTER IN BX REG
F3D1            4031   P7:                                     ; WRITE_LOOP
                4032
                4033   ;----- WAIT FOR HORIZONTAL RETRACE
                4034
F3D1 8B16630D   4035           MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F3D5 83C206     4036           ADD     DX,6                    ; POINT AT STATUS PORT
F3D8            4037   P8:
F3D8 EC         4038           IN      AL,DX                   ; GET STATUS
F3D9 A801       4039           TEST    AL,1                    ; IS IT LOW
F3DB 75FB       4040           JNZ     P8                      ; WAIT UNTIL IT IS
F3DD FA         4041           CLI                             ; NO MORE INTERRUPTS
F3DE            4042   P9:
F3DE EC         4043           IN      AL,DX                   ; GET STATUS
F3DF A801       4044           TEST    AL,1                    ; IS IT HIGH
F3E1 74FB       4045           JZ      P9                      ; WAIT UNTIL IT IS
F3E3 8BC3       4046           MOV     AX,BX                   ; RECOVER THE CHAR/ATTR
F3E5 AB         4047           STOSW                           ; PUT THE CHAR/ATTR
F3E6 FB         4048           STI                             ; INTERRUPTS BACK ON
F3E7 E2E8       4049           LOOP    P7                      ;  AS MANY TIMES AS REQUESTED
F3E9 E9D9FD     4050           JMP     VIDEO_RETURN
                4051   WRITE_AC_CURRENT        ENDP
```

Commentaires.- 1^o) Dans le cas d'un mode graphique ou couleur, on est renvoyé à une autre procédure (lignes 4018 à 4022), ce qui ne nous intéressera pas pour l'instant.

- 2^o) Sinon l'attribut est placé dans le registre AH (ligne 4024), si bien que AX contient maintenant ce qu'il y a à placer dans la mémoire graphique. On sauvegarde momentanément cette valeur sur la pile (ligne 4025), ainsi que le nombre de fois que l'on veut écrire le caractère (ligne 4026).

- 3^o) On détermine alors le déplacement dans la mémoire graphique de l'endroit auquel sera placé ce caractère (ligne 4027), que l'on place dans DI (ligne 4028).

On utilise pour cela la sous-routine FIND_POSITION, définie à partir de la ligne 3984 :

```
                3984
F39E            3985   FIND_POSITION   PROC    NEAR
F39E 8ACF       3986           MOV     CL,BH                       ; DISPLAY PAGE TO CX
F3A0 32ED       3987           XOR     CH,CH
F3A2 8BF1       3988           MOV     SI,CX                       ; MOVE TO SI FOR INDEX
F3A4 D1E6       3989           SAL     SI,1                        ; * 2 FOR WORD OFFSET
F3A6 884450     3990           MOV     AX,[SI+OFFSET CURSOR_POSN]  ; GET ROW/COLUMN OF THIS PAGE
F3A9 330B       3991           XOR     BX,BX                       ; SET START ADDRESS TO ZERO
F3AB E306       3992           JCXZ    P5                          ; NO_PAGE
F3AD            3993   P4:                                         ; PAGE_LOOP
F3AD 031E4C00   3994           ADD     BX,CRT_LEN                  ; LENGTH OF BUFFER
F3B1 E2FA       3995           LOOP    P4
F3B3            3996   P5:                                         ; NO_PAGE
F3B3 E8CFFE     3997           CALL    POSITION                    ; DETERMINE LOCATION OF BUFFER
F3B6 03D8       3998           ADD     BX,AX                       ; ADD TO START OF REGEN
F3B8 C3         3999           RET
                4000   FIND_POSITION   ENDP
```

On place le numéro de page dans SI (lignes 3786 à 3788), que l'on multiplie par deux (ligne 3789) car cela concerne des mots et non des octets. On récupère dans AX les coordonnées du curseur (ligne 3790 ; rappelons que CURSOR_POSN est un emplacement de la zone de communication vidéo du BIOS, défini à la ligne 170). On place dans BX le décalage du début de la zone de mémoire graphique consacrée à cette page (lignes 3791 à 3795). On place dans AX le décalage de la position du caractère par rapport à celui-ci (ligne 3797), ce qui ajouté à ce qui précède (ligne 3798) donne le décalage de la position du caractère dans la mémoire graphique.

- 4^o) On attend un rebroussement horizontal avant de changer quoi que ce soit dans la mémoire graphique, afin déviter de se retrouver dans un état instable.

Pour cela on lit le statut du contrôleur graphique sur le registre d'index 6 du 6845 (lignes 4035 à 4038 ; rappelons que l'adresse de base ADDR_6845 du 6845 est une donnée de la zone de communication vidéo du BIOS). On attend que le bit 0 soit à niveau bas (lignes 4039 et 4040). On ne permet pas les interruptions masquables (ligne 4041) et on teste à nouveau (lignes 4043 à 4045).

- 5^o) Lorsqu'un rebroussement horizontal est en train de se produire, on transfère ce qu'il faut placer dans la mémoire graphique du registre BX au registre AX (ligne 4046) et on le transfère autant de fois qu'il faut (lignes 4047 à 4049).

7.5.6 Affichage d'un caractère à la position du curseur sans changement d'attribut

Le début du code de cette fonction commence ligne 4052 :

```
                4052   ;--------------------------------------------
                4053   ; WRITE_C_CURRENT                           :
                4054   ;       THIS ROUTINE WRITES THE CHARACTER AT   :
                4055   ;       THE CURRENT CURSOR POSITION, ATTRIBUTE :
                4056   ;       UNCHANGED                           :
                4057   ; INPUT                                     :
                4058   ;       (AH) = CURRENT CRT MODE             :
                4059   ;       (BH) = DISPLAY PAGE                 :
                4060   ;       (CX) = COUNT OF CHARACTERS TO WRITE    :
                4061   ;       (AL) = CHAR TO WRITE                :
                4062   ;       (DS) = DATA SEGMENT                 :
                4063   ;       (ES) = REGEN SEGMENT                :
                4064   ; OUTPUT                                    :
                4065   ;       NONE                                :
                4066   ;--------------------------------------------
F3EC            4067   WRITE_C_CURRENT PROC    NEAR
F3EC 80FC04     4068           CMP     AH,4                    ; IS THIS GRAPHICS
F3EF 7208       4069           JC      P10
F3F1 8DFC07     4070           CMP     AH,7                    ; IS THIS BW CARD
F3F4 7403       4071           JE      P10
F3F6 E97F01     4072           JMP     GRAPHICS_WRITE
F3F9            4073   P10:
F3F9 50         4074           PUSH    AX                      ; SAVE ON STACK
F3FA 51         4075           PUSH    CX                      ; SAVE WRITE COUNT
F3FB E8A0FF     4076           CALL    FIND_POSITION
F3FE 8BFB       4077           MOV     DI,BX                   ; ADDRESS TO DI
F400 59         4078           POP     CX                      ; WRITE COUNT
F401 5B         4079           POP     BX                      ; BL HAS CHAR TO WRITE
F402            4080   P11:                                    ; WRITE_LOOP
                4081
                4082   ;----- WAIT FOR HORIZONTAL RETRACE
                4083
F402 8B166300   4084           MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F406 83C206     4085           ADD     DX,6                    ; POINT AT STATUS PORT
F409            4086   P12:
F409 EC         4087           IN      AL,DX                   ; GET STATUS
F40A A801       4088           TEST    AL,1                    ; IS IT LOW
F40C 75FB       4089           JNZ     P12                     ; WAIT UNTIL IT IS
F40E FA         4090           CLI                             ; NO MORE INTERRUPTS
F40F            4091   P13:
F40F EC         4092           IN      AL,DX                   ; GET STATUS
F410 A801       4093           TEST    AL,1                    ; IS IT HIGH
F412 74FB       4094           JZ      P13                     ; WAIT UNTIL IT IS
F414 8AC3       4095           MOV     AL,BL                   ; RECOVER CHAR
F416 AA         4096           STOSB                           ; PUT THE CHAR/ATTR
F417 FB         4097           STI                             ; INTERRUPTS BACK ON
F418 47         4098           INC     DI                      ; BUMP POINTER PAST ATTRIBUTE
F419 E2E7       4099           LOOP    P11                     ;  AS MANY TIMES AS REQUESTED
F41B E9A7FD     4100           JMP     VIDEO_RETURN
                4101   WRITE_C_CURRENT ENDP
```

Il est inutile de le commenter. Il suffit de se référer à la fonction précédente.

7.5.7 Spécification du mode graphique

Le début du code de la routine de service de l'interruption 10h est clairement indiqué à partir de la ligne 3386 :

```
                        3386   ;--------------------------------------------------
                        3387   ; SET_MODE                                       :
                        3388   ;       THIS ROUTINE INITIALIZES THE ATTACHMENT TO   :
                        3389   ;       THE SELECTED MODE.  THE SCREEN IS BLANKED.   :
                        3390   ; INPUT                                          :
                        3391   ;       (AL) = MODE SELECTED (RANGE 0-9)            :
                        3392   ; OUTPUT                                         :
                        3393   ;       NONE                                        :
                        3394   ;--------------------------------------------------
                        3395
                        3396   ;----- TABLES FOR USE IN SETTING OF MODE
                        3397
F0A4                    3398           ORG     0F0A4H
F0A4                    3399   VIDEO_PARMS     LABEL  BYTE
                        3400   ;----- INIT_TABLE
F0A4 38                 3401           DB      38H,28H,2DH,0AH,1FH,6,19H      ; SET UP FOR 40X25
F0A5 28
F0A6 2D
F0A7 0A
F0A8 1F
F0A9 06
F0AA 19
F0AB 1C                 3402           DB      1CH,2,7,6,7
F0AC 02
F0AD 07
F0AE 06
F0AF 07
F0B0 00                 3403           DB      0,0,0,0
F0B1 00
F0B2 00
F0B3 00
  0010                  3404   M4      EQU     $-VIDEO_PARMS
                        3405
F0B4 71                 3406           DB      71H,50H,5AH,0AH,1FH,6,19H      ; SET UP FOR 80X25
F0B5 50
F0B6 5A
F0B7 0A
F0B8 1F
F0B9 06
F0BA 19
F0BB 1C                 3407           DB      1CH,2,7,6,7
F0BC 02
F0BD 07
F0BE 06
F0BF 07
F0C0 00                 3408           DB      0,0,0,0
F0C1 00
F0C2 00
F0C3 00
                        3409
F0C4 38                 3410           DB      38H,28H,2DH,0AH,7FH,6,64H      ; SET UP FOR GRAPHICS
F0C5 28
F0C6 2D
F0C7 0A
F0C8 7F
F0C9 06
F0CA 64
F0CB 70                 3411           DB      70H,2,1,6,7
F0CC 02
F0CD 01
F0CE 06
F0CF 07
F0D0 00                 3412           DB      0,0,0,0
F0D1 00
F0D2 00
F0D3 00
                        3413
F0D4 61                 3414           DB      61H,50H,52H,0FH,19H,6,19H      ; SET UP FOR 80x25 B&W CARD
F0D5 50
```

```
F0D6 52
F0D7 0F
F0D8 19
F0D9 06
F0DA 19
F0DB 19          3415          DB      19H,2,0DH,0BH,0CH
F0DC 02
F0DD 0D
F0DE 0B
F0DF 0C
F0E0 00          3415          DB      0,0,0,0
F0E1 00
F0E2 00
F0E3 00
                 3417
F0E4             3418  M5      LABEL   WORD                            ; TABLE OF REGEN LENGTHS
F0E4 0D0B        3419          DW      2048                            ; 40X25
F0E6 0D10        3420          DW      4096                            ; 80X25
F0E8 0D4D        3421          DW      16384                           ; GRAPHICS
F0EA 0D40        3422          DW      16384
                 3423
                 3424  ;----- COLUMNS
                 3425
F0EC             3426  M6      LABEL   BYTE
F0EC 28          3427          DB      40,40,80,80,40,40,80,80
F0ED 28
F0EE 50
F0EF 50
F0F0 28
F0F1 28
F0F2 50
F0F3 50
                 3428
                 3429  ;----- C_REG_TAB
                 3430
F0F4             3431  M7      LABEL   BYTE                            ; TABLE OF MODE SETS
F0F4 2C          3432          DB      2CH,28H,2DH,29H,2AH,2EH,1EH,29H
F0F5 28
F0F6 2D
F0F7 29
F0F8 2A
F0F9 2E
F0FA 1E
F0FB 29
                 3433
F0FC             3434  SET_MODE        PROC      NEAR
F0FC BAD403      3435          MOV     DX,03D4H                        ; ADDRESS OF COLOR CARD
F0FF B300        3436          MOV     BL,0                            ; MODE SET FOR COLOR CARD
F101 83FF30      3437          CMP     DI,30H                          ; IS BW CARD INSTALLED
F104 7506        3438          JNE     M8                              ; OK WITH COLOR
F106 B007        3439          MOV     AL,7                            ; INDICATE BW CARD MODE
F108 B2B4        3440          MOV     DL,0B4H                         ; ADDRESS OF BW CARD (3B4)
F10A FEC3        3441          INC     BL                              ; MODE SET FOR BW CARD
F10C             3442  M8:
F10C 8AE0        3443          MOV     AH,AL                           ; SAVE MODE IN AH
F10E A24900      3444          MOV     CRT_MODE,AL                     ; SAVE IN GLOBAL VARIABLE
F111 89166300    3445          MOV     ADDR_6845,DX                    ; SAVE ADRRESS OF BASE
F115 1E          3446          PUSH    DS                              ; SAVE POINTER TO DATA SEGMENT
F116 50          3447          PUSH    AX                              ; SAVE MODE
F117 52          3448          PUSH    DX                              ; SAVE OUTPUT PORT VALUE
F118 83C204      3449          ADD     DX,4                            ; POINT TO CONTROL REGISTER
F11B 8AC3        3450          MOV     AL,BL                           ; GET MODE SET FOR CARD
F11D EE          3451          OUT     DX,AL                           ; RESET VIDEO
F11E 5A          3452          POP     DX                              ; BACK TO BASE REGISTER
F11F 2BC0        3453          SUB     AX,AX                           ; SET UP FOR ABS0 SEGMENT
F121 8ED8        3454          MOV     DS,AX                           ; ESTABLISH VECTOR TABLE ADDRESSING
                 3455          ASSUME  DS:ABS0
F123 C51E7400    3456          LDS     BX,PARAM_PTR                    ; GET POINTER TO VIDEO PARMS
F127 58          3457          POP     AX                              ; RECOVER PARMS
                 3458          ASSUME  DS:CODE
F128 B91000      3459          MOV     CX,M4                           ; LENGTH OF EACH ROW OF TABLE ADDRESSING
F12B 80FC02      3460          CMP     AH,2                            ; DETERMINE WHICH ONE TO USE
F12E 7210        3461          JC      M9                              ; MODE IS 0 OR 1
F130 0309        3462          ADD     BX,CX                           ; MOVE TO NEXT ROW OF INIT TABLE
```

```
F132 80FC04       3463          CMP     AH,4
F135 7209         3464          JC      M9                              ; MODE IS 2 OR 3
F137 03D9         3465          ADD     BX,CX                           ; MOVE TO GRAPHICS ROW OF INIT_TABLE
F139 80FC07       3466          CMP     AH,7
F13C 7202         3467          JC      M9                              ; MODE IS 4,5 OR 6
F13E 0309         3468          ADD     BX,CX                           ; MOVE TO BW CARD ROW OF INIT_TABLE
                  3469
                  3470  ;----- BX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
                  3471
F140              3472  M9:                                             ; OUT_INIT
F140 50           3473          PUSH    AX                              ; SAVE MODE IN AH
F141 32E4         3474          XOR     AH,AH                           ; AH WILL SERVE AS REGISTER
                  3475                                                  ;  NUMBER DURING LOOP
                  3476
                  3477  ;----- LOOP TROUGH TABLE, OUTPUTTING REG ADDRESS, THEN VALUE FROM TABLE
                  3478
F143              3479  M10:                                            ; INIT LOOP
F143 8AC4         3480          MOV     AL,AH                           ; GET 6845 REGISTER NUMBER
F145 EE           3481          OUT     DX,AL
F146 42           3482          INC     DX                              ; POINT TO DATA PORT
F147 FEC4         3483          INC     AH                              ; NEXT REGISTER VALUE
F149 8A07         3484          MOV     AL,[BX]                         ; GET TABLE VALUE
F14B EE           3485          OUT     DX,AL                           ; OUT TO CHIP
F14C 43           3486          INC     BX                              ; NEXT IN TABLE
F14D 4A           3487          DEC     DX                              ; BACK TO POINTER REGISTER
F14E E2F3         3488          LOOP    M10                             ; DO THE WHOLE TABLE
F150 58           3489          POP     AX                              ; GET MODE BACK
F151 1F           3490          POP     DS                              ; RECOVER SEGMENT VALUE
                  3491          ASSUME  DS:DATA
                  3492
                  3493  ;----- FILL REGEN AREA WITH BLANK
                  3494
F152 33FF         3495          XOR     DI,DI                           ; SET UP POINTER FOR REGEN
F154 893E4E00     3496          MOV     CRT_START,DI                    ; START ADDRESS SAVED IN GLOBAL
F158 C606620000   3497          MOV     ACTIVE_PAGE,0                   ; SET PAGE VALUE
F15D B90020       3498          MOV     CX,8192                         ; NUMBER OF WORDS IN COLOR CARD
F160 80FC04       3499          CMP     AH,4                            ; TEST FOR GRAPHICS
F163 720B         3500          JC      M12                             ; NO_GRAPHICS_INIT
F165 80FC07       3501          CMP     AH,7                            ; TEST FOR BW CARD
F168 7404         3502          JE      M11                             ; BW_CARD_INIT
F16A 33C0         3503          XOR     AX,AX                           ; FILL FOR GRAPHICS MODE
F16C EB05         3504          JMP     SHORT M13                       ; CLEAR_BUFFER
F16E              3505  M11:                                            ; BW_CARD_INIT
F16E B508         3506          MOV     CH,08H                          ; BUFFER SIZE ON BW CARD
F170              3507  M12:                                            ; NO_GRAPHICS_INIT
F170 B82007       3508          MOV     AX,' '+7*256                    ; FILL CHAR FOR ALPHA
F173              3509  M13:                                            ; CLEAR_BUFFER
F173 F3           3510          REP     STOSW                           ; FILL THE REGEN BI=UFFER WITH BLANKS
                  3511
                  3512  ;----- ENABLE VIDEO AND CORRECT PORT SETTING
                  3513
F175 C70660000706 3514          MOV     CURSOR_MODE,607H                ; SET CURRENT CURSOR MODE
F17B A04900       3515          MOV     AL,CRT_MODE                     ; GET THE MODE
F17E 32E4         3516          XOR     AH,AH                           ;  INTO AX REGISTER
F180 8BF0         3517          MOV     SI,AX                           ; TABLE POINTER, INDEXED BY MODE
F182 8B166300     3518          MOV     DX,ADDR_6845                    ; PREPARE TO OUTPUT TO
                  3519                                                  ;  VIDEO ENABLE PORT
F186 83C204       3520          ADD     DX,4
F189 2E8A84F4F0   3521          MOV     AL,CS:[SI+ OFFSET M7]
F18E EE           3522          OUT     DX,AL                           ; SET VIDEO ENABLE PORT
F18F A26500       3523          MOV     CRT_MODE_SET,AL                 ; SAVE THAT VALUE
                  3524
                  3525  ;----- DETERMINE NUMBER OF COLUMNS, BOTH FOR ENTIRE DISPLAY
                  3526  ;----- AND THE NUMBER TO BE USED FOR TTY INTERFACE
                  3527
F192 2E8A84ECF0   3528          MOV     AL,CS:[SI+ OFFSET M6]
F197 32E4         3529          XOR     AH,AH
F199 A34A00       3530          MOV     CRT_COLS,AX                     ; NUMBER OF COLUMNS IN THIS SCREEN
                  3531
                  3532  ;----- SET CURSOR POSITIONS
                  3533
F19C 81E60E00     3534          AND     SI,0EH                          ; WORD OFFSET INTO CLEAR LENGTH TABLE
F1A0 2E888CE4F0   3535          MOV     CX,CS:[SI+OFFSET M5]            ; LENGTH TO CLEAR
F1A5 890E4C00     3536          MOV     CRT_LEN,CX                      ; SAVE LENGTH OF CRT -- NOT USED FOR BW
```

```
F1A9 8908D0        3537          MOV     CX,8                            ; CLEAR ALL CURSOR POSITIONS
F1AC BF5000        3538          MOV     DI,OFFSET CURSOR_POSN
F1AF 1E            3539          PUSH    DS                              ; ESTABLISH SEGMENT
F1B0 07            3540          POP     ES                              ;  ADDRESSING
F1B1 33C0          3541          XOR     AX,AX
F1B3 F3            3542          REP     STOSW                           ; FILL WITH ZEROES
F1B4 AB
                   3543
                   3544   ;----- SET UP OVERSCAN REGISTER
                   3545
F1B5 42            3546          INC     DX                              ; SET OVERSCAN PORT TO A DEFAULT
F1B6 B030          3547          MOV     AL,30H                          ; VALUE OF 30H FOR ALL MODES
                   3548                                                  ;  EXCEPT 640X200
F1B8 803E490006    3549          CMP     CRT_MODE,6                      ; SEE IF THE MODE IS 640X200 BW
F1BD 7502          3550          JNZ     M14                             ; IF IT ISNT 640X200, THEN GOTO REGULAR
F1BF B03F          3551          MOV     AL,3FH                          ; IF IT IS 640X200, THEN PUT IN 3FH
F1C1               3552   M14:
F1C1 EE            3553          OUT     DX,AL                           ; OUTPUT THE CORRECT VALUE TO 3D9 PORT
F1C2 A26600        3554          MOV     CRT_PALETTE,AL                  ; SAVE THE VALUE FOR FUTURE USE
                   3555
                   3556   ;----- NORMAL RETURN FROMM ALL VIDEO RETURNS
```

Commentaires.- 1^o) On place cette sous-routine à l'adresse absolue F0A4h (ligne 3398), de façon à correspondre à l'adresse annoncée dans les table des vecteurs d'interruption.

- 2^o) Suit la table des paramètres d'initialisation du MC6845 pour les différents modes (lignes 3400 à 3415), une ligne de la table par mode. La longueur de la ligne est définie ligne 3404, mais elle est évidemment égale à 16, le nombre de registres internes du MC6845.

Pour le mode 0 (25 lignes de 40 caractères), défini lignes 3401 à 3403, on a 56 caractères en tout par ligne (registre d'index 0), 40 caractères visibles (registre d'index 1), 45 pour que le rebroussement horizontal commence au quarante-sixième caractère (registre d'index 2), la durée de l'impulsion HS est 10 (registre d'index 3), on a 31 lignes en tout (registre d'index 4), l'équivalent de 6 lignes pour le rebroussement vertical (registre d'index 5), 25 lignes visibles (registre d'index 6), le rebroussement vertical commençant à la ligne 28 (registre d'index 7), ce qui laisse 3 lignes de bord, on est en mode non entrelacé (2 pour le registre d'index 8), on a 8 (= 7 + 1) lignes par caractère (registre d'index 9), le début du curseur est 6 (registre d'index 10), il commence donc à la ligne 6 et est non clignotant, la fin du curseur est à la ligne 7 (registre d'index 11), ce qui donne un curseur de deux lignes, le décalage dans la mémoire graphique est 0 après un rebroussement vertical (registres d'index 12 et 13), c'est-à-dire qu'on est au début de la page 0, le curseur se trouve initialement au coin supérieur gauche (0 dans les registres 14 et 15).

- 3^o) Suit la table des tailles d'une page en mémoire graphique (lignes 3418 à 3422) : on a évidemment 2 048 pour une page texte de 25 lignes de 40 caractères, un caractère occupant deux octets, 4 096 pour une page texte de 25 lignes de 80 caractères et 16 384 dans le cas graphique.

- 4^o) Suit la table du nombre de caractères par ligne dans les 8 modes (lignes 3424 à 3427).

- 5^o) La dernière table (lignes 3429 à 3432) spécifie l'octet de contrôle à placer dans le registre de contrôle du 6845, par exemple, comme nous l'avons vu, 29h pour la carte MDA.

- 6^o) La première action de la sous-routine consiste à placer le numéro de mode dans la variable CRT_MODE de la zone de communication du BIOS (ligne 3444). Ce n'est pas exactement le contenu de AL passé en paramètre, car il faut rectifier s'il s'agit d'une carte monochrome (mode 7 ; lignes 3437 et 3439).

Rappelons qu'on a placé dans le registre DI le contenu de l'interrupteur de configuration à la ligne 3375.

- 7^o) La seconde action de la sous-routine consiste à placer l'adresse (de base) du MC6845 dans la variable ADDR_6845 de la zone de communication du BIOS (ligne 3445) : il s'agit de 3D4h pour l'adaptateur couleur et graphique (en fait 3D0h mais on prend 3D4h pour émuler l'adaptateur monochrome) et de 3B4h pour l'adaptateur monochrome (lignes 3435, 3437, 3438 et 3440).

On conserve le type d'adaptateur (0 pour l'adaptateur couleur et graphique, 1 pour l'adaptateur monochrome) dans le registre BL (lignes 3436 et 3441).

- 8^o) On réinitialise alors le MC6845 en lui envoyant le contenu de BL dans le registre de contrôle, dont l'adresse est l'adresse de base plus quatre (lignes 3446 à 3452).

On envoie 1 pour la carte MDA car le bit 0 doit toujours être positionné à 1.

On envoie 0 pour la carte CGA.

- 9^o) On place l'adresse de la table d'initialisation dans le registre BX (lignes 3453 à 3458). Rappelons que le segment absolu (d'adresse 0) ABSO est défini ligne 19 et PARAM_PTR ligne 51.

On place dans CX la longueur d'une ligne de la table (ligne 3459).

On se déplace au début de la ligne correspondant au mode (lignes 3460 à 3468).

On sauvegarde le mode (ligne 3473) car AH va servir d'index de parcours de la ligne du tableau d'initialisation, que l'on initialise donc à 0 (ligne 3474).

On initialise enfin les seize registres du 6845 avec les valeurs de la ligne du tableau d'initialisation (lignes 3479 à 3491).

- 10^o) On initialise ensuite la mémoire graphique pour avoir une page blanche, c'est-à-dire noire avec le curseur en haut à gauche.

Le registre DI, qui va décrire les caractères, est initialisé à zéro (ligne 3495). On sauvegarde cette position dans la variable globale de la zone de dommunication du BIOS (ligne 3496). On initialise le registre CX avec le nombre de caractères (cas d'une page texte) ou de pixels (cas d'une page graphique) d'éléments à initialiser (lignes 3498 à 3502 et 3505). On initialise le registre AX avec la valeur de l'élément (lignes 3503 et 3504 pour une page graphique ; ligne 3508 pour une page texte) : dans le premier cas, on a 0 ; dans le second cas, le caractère est un espace ayant pour attribut : caractère non clignotant, non intense, blanc sur fond noir. On remplit alors la mémoire graphique avec cet élément (ligne 3510).

- 11^o) On initialise le mode du curseur, dans la zone de communication du BIOS, à 607h (ligne 3514), en disant qu'il occupera les lignes 6 et 7.

- 12^o) On initialise le registre SI avec le bon numéro de ligne de la table des octets de contrôle (lignes 3515 à 3517). On initialise le registre DX avec le port du registre de contrôle du 6845 (lignes 3518 et 3519). On place dans le registre de contrôle l'octet de contrôle par défaut (lignes 3521 et 3522), par exemple 29h pour la carte MDA. On le place également dans la zone de communication du BIOS (ligne 3523).

- 13^o) Le nombre de colonne de l'écran est récupéré dans la table d'initialisation et est placé dans la zone de communication du BIOS (lignes 3525 à 3530).

- 14^o) De même, la taille de la mémoire graphique à utiliser est récupérée dans la table adéquate et est placée dans la zone de communication du BIOS (lignes 3534 à 3536).

- 15^o) On initialise la position du curseur au coin supérieur gauche pour toutes les pages (lignes 3537 à 3542).

- 16^o) On initialise enfin la palette (lignes 3544 à 3554), à la fois dans le registre de la carte CGA et dans la zone de communication du BIOS.

7.5.8 Spécification de la forme du curseur

La fonction SET_CTYPE spécifie la forme du curseur. Le début du code de la fonction commence ligne 3569 :

```
                3569   ;--------------------------------------------------------
                3570   ; SET_CTYPE                                             :
                3571   ;       THIS ROUTINE SETS THE CURSOR VALUE              :
                3572   ; INPUT                                                 :
                3573   ;       (CX) HAS CURSOR VALUE CH-START LINE, CL STOP LINE   :
                3574   ; OUTPUT                                                :
                3575   ;       NONE                                            :
                3576   ;--------------------------------------------------------
F1CD            3577   SET_CTYPE       PROC    NEAR
F1CD B40A       3578           MOV     AH,10                   ; 6845 REGISTER FOR CURSOR SET
F1CF 890E6000   3579           MOV     CURSOR_MODE,CX          ; SAVE IN DATA AREA
F1D3 E80200     3580           CALL    M16                     ; OUTPUT IN DATA AREA
F1D6 EBED       3581           JMP     VIDEO_RETURN
                3582
                3583   ;----- THIS ROUTINE OUTPUT THE CX REGISTER TO THE 6845 REGS NAMED IN AH
                3584
F1D8            3585   M16:
F1D8 8B166300   3586           MOV     DX,ADDR_6845            ; ADDRESS REGISTER
F1DC BAC4       3587           MOV     AL,AH                   ; GET VALUE
F1DE EE         3588           OUT     DX,AL                   ; REGISTER SET
F1DF 42         3589           INC     DX                      ; DATA REGISTER
F1E0 8AC5       3590           MOV     AL,CH                   ; DATA
F1E2 EE         3591           OUT     DX,AL
F1E3 4A         3592           DEC     DX
F1E4 8AC4       3593           MOV     AL,AH
F1E6 FEC0       3594           INC     AL                      ; POINT TO OTHER DATA REGISTER
F1E8 EE         3595           OUT     DX,AL                   ; SET FOR SECOND REGISTER
F1E9 42         3596           INC     DX
F1EA 8AC1       3597           MOV     AL,CL                   ; SECOND VALUE
F1EC EE         3598           OUT     DX,AL
F1ED C3         3599           RET                             ; ALL DONE
                3600   SET_CTYPE       ENDP
```

Commentaire.- On place le contenu de CH dans le registre d'index 10 du 6845 et le contenu de CL dans le registre d'index 11.

7.5.9 Spécification de la position du curseur

La fonction SET_CPOS place le curseur à la position spécifiée par DX (ligne et colonne) et BH (page). Le début du code de la fonction commence ligne 3601 :

```
                3601  ;--------------------------------------------------------
                3602  ; SET_CPOS                                              :
                3603  ;       THIS ROUTINE SETS THE CURRENT CURSOR            :
                3604  ;       POSITION TO THE NEW X-Y VALUES PASSED           :
                3605  ; INPUT                                                 :
                3606  ;       DX - ROW,COLUMN OF NEW CURSOR                   :
                3607  ;       BH - DISPLAY PAGE OF CURSOR                     :
                3608  ; OUTPUT                                                :
                3609  ;       CURSOR IS SET AT 6845 IF DISPLAY PAGE           :
                3610  ;       IS CURRENT DISPLAY                              :
                3611  ;--------------------------------------------------------
F1EE            3612  SET_CPOS        PROC    NEAR
F1EE 8ACF       3613          MOV     CL,BH
F1F0 32ED       3614          XOR     CH,CH                   ; ESTABLISH LOOP COUNT
F1F2 D1E1       3615          SAL     CX,1                    ; WORD OFFSET
F1F4 8BF1       3616          MOV     SI,CX                   ; USE INDEX REGISTER
F1F6 895450     3617          MOV     [SI+OFFSET CURSOR_POSN],DX  ; SAVE THE POINTER
F1F9 383E6200   3618          CMP     ACTIVE_PAGE,BH
F1FD 7505       3619          JNZ     M17                     ; SET_CPOS_RETURN
F1FF 88C2       3620          MOV     AX,DX                   ; GET ROW/COLUMN TO AX
F201 E80200     3621          CALL    M18                     ; CURSOR_SET
F204            3622  M17:                                    ; SET_CPOS_RETURN
F204 EBBF       3623          JMP     VIDEO_RETURN
                3624  SET_CPOS        ENDP
                3625
                3626  ;----- SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR
                3627
F206            3628  M18     PROC    NEAR
F206 EB7C00     3629          CALL    POSITION                ; DETERMINE LOCATION IN REGEN BUFFER
F209 88C8       3630          MOV     CX,AX
F20B 030E4E00   3631          ADD     CX,CRT_START            ; ADD IN THE START ADDR FOR THIS PAGE
F20F D1F9       3632          SAR     CX,1                    ; DIVIDE BY 2 FOR CHAR ONLY COUNT
F211 B40E       3633          MOV     AH,14                   ; REGISTER NUMBER FOR CURSOR
F213 EBC2FF     3634          CALL    M16                     ; OUTPUT THE VALUE TO THE 6845
F216 C3         3635          RET
                3636  M18     ENDP
```

Commentaires.- 1^o) On initialise le registre SI avec le double du numéro de page (lignes 3613 à 3616).

- 2^o) On sauvegarde la position du curseur passée en paramètre à son emplacement prévu dans la zone de communication du BIOS (ligne 3617).

- 3^o) Si le numéro de page passé en paramètre n'est pas la page active, on a terminé (lignes 3618 et 3619).

- 4^o) Sinon on déplace le curseur sur l'écran en cours (lignes 3620 et 3621 et 3626 à 3635).

7.5.10 Spécification de la page active

La fonction ACT_DISP_PAGE permet de spécifier la page active, dont le numéro est passé grâce au registre AL. Le début du code de la fonction commence ligne 3637 :

```
                3637   ;--------------------------------------------------------------
                3638   ; ACT_DISP_PAGE                                               :
                3639   ;       THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING THE :
                3640   ;       FULL USE OF THE RAM SET ASIDE FOR THE VIDEO ATTACHMENT  :
                3641   ; INPUT                                                       :
                3642   ;       AL HAS THE NEW ACTIVE DISPLAY PAGE                    :
                3643   ; OUTPUT                                                      :
                3644   ;       THE 6845 IS RESET TO DISPLAY THAT PAGE                :
                3645   ;--------------------------------------------------------------
F217            3646   ACT_DISP_PAGE   PROC    NEAR
F217 A26200     3647           MOV     ACTIVE_PAGE,AL              ; SAVE ACTIVE PAGE VALUE
F21A 8BDE4C00   3648           MOV     CX,CRT_LEN                  ; GET SAVED LENGTH OF REGEN BUFFER
F21E 98         3649           CBW                                 ; CONVERT AL TO WORD
F21F 50         3650           PUSH    AX                          ; SAVE PAGE VALUE
F220 F7E1       3651           MUL     CX                          ; DISPLAY PAGE TIMES REGEN LENGTH
F222            3652           MOV     CRT_START,AX                ; SAVE START ADDRESS FOR
                3653                                               ;  LATER REQUIREMENTS
F225 8BC8       3654           MOV     CX,AX                       ; START ADDRESS TO CX
F227 D1F9       3655           SAR     CX,1                        ; DIVIDE BY 2 FOR 6845 HANDLING
F229 B4DC       3656           MOV     AH,12                       ; 6845 REGISTER FOR START ADDRESS
F22B E8AAFF     3657           CALL    M16
F22E 5B         3658           POP     BX                          ; RECOVER PAGE VALUE
F22F D1E3       3659           SAL     BX,1                        ; *2 FOR WORD EFFECT
F231 884750     3660           MOV     AX,[BX + OFFSET CURSOR_POSN] ; GET CURSOR FOR THIS PAGE
F234 E8CFFF     3661           CALL    M18                         ; SET THE CURSOR POSITION
F237 EB8C       3662           JMP     SHORT VIDEO_RETURN
                3663   ACT_DISP_PAGE   ENDP
```

Commentaires.- 1^o) On place la valeur de la page active passée en paramètre dans la zone de communication du BIOS (ligne 3647) tout en la sauvegardant sur la pile (lignes 3649 et 3650).

- 2^o) La taille d'une page est placée dans le registre CX (ligne 3648), on la multiplie par le numéro de page (ligne 3651), ce qui donne le déplacement par rapport au début de la mémoire graphique, que l'on place dans la variable adéquate de la zone de communication du BIOS (ligne 3652).

- 3^o) On transmet ce déplacement au registre adéquat du 6845 (lignes 3654 à 3657).

- 4^o) On place le numéro de page dans le registre BX (ligne 3658), que l'on multiplie par 2 car la position du curseur est codée sur un mot (ligne 3659), ce qui permet d'obtenir la position du curseur pour cette page (ligne 3660), que l'on place à l'écran (ligne 3661).

7.5.11 Consultation des données du curseur

La fonction READ_CURSOR permet de lire les données du curseur (position et forme) d'une page donnée. Le début du code de la fonction commence ligne 3664 :

```
                3664  ;--------------------------------------------------------------
                3665  ; READ_CURSOR                                                 :
                3666  ;       THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE    :
                3667  ;       6845, FORMATS IT, AND SENDS IT BACK TO THE CALLER       :
                3668  ; INPUT                                                       :
                3669  ;       BH - PAGE OF CURSOR                                    :
                3670  ; OUTPUT                                                      :
                3671  ;       DX - ROW,COLUMN OF THE CURRENT CURSOR POSITION         :
                3672  :       CX - CURRENT CURSOR MODE                               :
                3673  ;--------------------------------------------------------------
F239            3674  READ_CURSOR     PROC    NEAR
F239 8ADF       3675          MOV     BL,BH
F23B 32FF       3676          XOR     BH,BH
F23D D1E3       3677          SAL     BX,1                        ; WORD OFFSET
F23F 885750     3678          MOV     DX,[BX+OFFSET CURSOR_POSN]
F242 8B0E6000   3679          MOV     CX,CURSOR_MODE
F246 5F         3680          POP     DI
F247 5E         3681          POP     SI
F248 5B         3682          POP     BX
F249 58         3683          POP     AX                          ; DISCARD SAVED CX AND DX
F24A 58         3684          POP     AX
F24B 1F         3685          POP     DS
F24C 07         3686          POP     ES
F24D CF         3687          IRET
                3688  READ_CURSOR     ENDP
```

Commentaires.- 1^o) On calcule le décalage adéquat, en fonction du numéro de page, que l'on place à l'emplacement CURSOR_POSN de la zone de communication du BIOS (lignes 3675 à 3677), ce qui permet de placer la position dans DX (ligne 3678).

- 2^o) On récupère la forme du curseur à partir de la variable adéquate de la zone de communication du BIOS (ligne 3779).

7.5.12 Spécification de la palette de couleurs pour le mode graphique

La fonction SET_COLOR permet d'initialiser les couleurs de fond et de premier plan pour la résolution graphique moyenne. Le début du code de la fonction commence ligne 3689 :

```
                3689  ;--------------------------------------------------------------------------
                3690  ; SET COLOR                                                               :
                3691  ;       THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN :
                3692  ;       COLOR, AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION      :
                3693  ;       GRAPHICS                                                        :
                3694  ; INPUT                                                                 :
                3695  ;       (BH) HAS COLOR ID                                               :
                3696  ;               IF BH=0, THE BACKGROUND COLOR VALUE IS SET              :
                3697  ;                    FROM THE LOW BITS OF BL (0-31)                     :
                3698  ;               IF BH=1, THE PALETTE SELECTION IS MADE                  :
                3699  ;                    BASED ON THE LOW BIT OF BL:                        :
                3700  ;                            0=GREEN, RED, YELLOW FOR COLORS 1,2,3      :
                3701  ;                            1=BLUE, CYAN, MAGENTA FOR COLORS 1,2,3     :
                3702  ;       (BL) HAS THE COLOR VALUE TO BE USED                             :
                3703  ; OUTPUT                                                                :
                3704  ;       THE COLOR SELECTION IS UPDATED                                  :
                3705  ;--------------------------------------------------------------------------
F24E            3706  SET_COLOR       PROC    NEAR
F24E 8B166300   3707          MOV     DX,ADDR_6845            ; I/O PORT FOR PALETTE
F252 83C205     3708          ADD     DX,5                    ; OVERSCAN PORT
F255 A06600     3709          MOV     AL,CRT_PALETTE          ; GET THE CURRENT PALETTE VALUE
F258 0AFF       3710          OR      BH,BH                   ; IS THIS COLOR 0?
F25A 750E       3711          JNZ     M20                     ; OUTPUT COLOR 1
                3712
                3713  ;----- HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
                3714
F25C 24E0       3715          AND     AL,0E0H                 ; TURN OFF LOW 5 BITS OF CURRENT
F25E 80E31F     3716          AND     BL,01FH                 ; TURN OFF HIGH 3 BITS OF INPUT VALUE
F261 0AC3       3717          OR      AL,BL                   ; PUT VALUE INTO REGISTER
F263            3718  M19:                                    ; OUTPUT THE PALETTE
F263 EE         3719          OUT     DX,AL                   ; OUTPUT COLOR SELECTIONTO 309 PORT
F264 A26600     3720          MOV     CRT_PALETTE,AL          ; SAVE THE COLOR VALUE
F267 E958FF     3721          JMP     VIDEO_RETURN
                3722
                3723  ;--- HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
                3724
F26A            3725  M20:
F26A 24DF       3726          AND     AL,0DFH                 ; TURN OFF PALETTE SELECT BIT
F26C 0DEB       3727          SHR     BL,1                    ; TEST THE LOW ORDER BIT OF BL
F26E 73F3       3728          JNC     M19                     ; ALREADY DONE
F270 0C20       3729          OR      AL,20H                  ; TURN ON PALETTE SELECT BIT
F272 EBEF       3730          JMP     M19                     ; GO DO IT
                3731  SET_COLOR       ENDP
```

Commentaires.- 1^o) La sélection des couleurs se fait sur la carte CGA *via* le port 3D8h, soit le port de base plus cinq (lignes 3706 et 3707).

- 2^o) On récupère la palette en cours à partir de la variable adéquate de la zone de communication du BIOS (ligne 3709).

- 3^o) Si l'identificateur de couleur, passé en paramètre par le registre BH, est 0 (lignes 3710 et 3711), c'est que l'on veut changer la couleur de fond (ligne 3696). On ne s'intéresse donc pas aux 5 bits de poids faible de AL, que l'on annule (ligne 3715). On ne s'intéresse pas non plus aux 3 bits de poids fort de BL, que l'on annule aussi (ligne 3716). On place la nouvelle valeur de la couleur de fond dans AL (ligne 3717) et on la transmet au 6845 (ligne 3719). On sauvegarde la nouvelle valeur dans la variable adéquate de la zone de communication du BIOS (ligne 3720).

- 4^o) Si l'identificateur de couleur, passé en paramètre par le registre BH, est 1 (lignes 3710 et 3711), c'est que l'on veut changer la palette (ligne 3698). On teste le bit de poids faible de BL pour déterminer la palette (lignes 3699 et 3727). On transmet la palette choisie au 6845 (lignes 3728 à 3730).

7.5.13 Consultation du mode graphique

La fonction VIDEO_STATE permet d'obtenir le mode graphique. Le début du code de la fonction commence ligne 3732 :

```
                3732   ;--------------------------------------------------------------
                3733   ; VIDEO STATE                                                 :
                3734   ;  RETURNS THE CURRENT VIDEO STATE IN AX                      :
                3735   ;  AH = NUMBER OF COLUMNS ON THE SCREEN                       :
                3736   ;  AL = CURRENT VIDEO MODE                                    :
                3737   ;  BH = CURRENT ACTIVE PAGE                                   :
                3738   ;--------------------------------------------------------------
F274            3739   VIDEO_STATE     PROC    NEAR
F274 8A264A00   3740           MOV     AH,BYTE PTR CTR_COLS     ; GET NUMBER OF COLUMNS
F278 A04900     3741           MOV     AL,CRT_MODE              ; CURRENT MODE
F27B 8A3E6200   3742           MOV     BH,ACTIVE_PAGE           ; GET CURRENT ACTIVE PAGE
F27F 5F         3743           POP     DI                       ; RECOVER REGISTERS
F280 5E         3744           POP     SI
F281 59         3745           POP     CX                       ; DISCARD SAVED BX
F282 E943FF     3746           JMP     M15                      ; RETURN TO CALLER
                3747   VIDEO_STATE     ENDP
```

Commentaires.- 1^o) On récupère le nombre de colonne à partir de la variable adéquate de la zone de communication du BIOS (ligne 3740), de même pour le mode graphique (ligne 3741) et le numéro de page (ligne 3742).

- 2^o) On termine proprement l'interruption (lignes 3743 à 3746) avec un petit changement par rapport au retour normal (VIDEO_RETURN déjà étudié) : on ne récupère évidemment pas l'ancienne valeur de BX qui contient maintenant des résultats.

7.5.14 Défilement vers le haut

La fonction SCROLL_UP exécute un défilement vers le haut. Le début du code de la fonction commence ligne 3768 :

```
                3768  ;--------------------------------------------------------------
                3769  ; SCROLL UP                                                   :
                3770  ;       THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP           :
                3771  ;       ON THE SCREEN                                         :
                3772  ; INPUT                                                      :
                3773  ;       (AH) = CURRENT CRT MODE                               :
                3774  ;       (AL) = NUMBER OF ROWS TO SCROLL                       :
                3775  ;       (CX) = ROW/COLUMN OF UPPER LEFT CORNER                :
                3776  ;       (DX) = ROW/COLUMN OF LOWER RIGHT CORNER               :
                3777  ;       (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE           :
                3778  ;       (DS) = DATA SEGMENT                                   :
                3779  ;       (ES) = REGEN BUFFER SEGMENT                           :
                3780  ; OUTPUT                                                     :
                3781  ;       NONE -- THE REGEN BUFFER IS MODIFIED                  :
                3782  ;--------------------------------------------------------------
                3783          ASSUME  CS:CODE,DS:DATA,ES:DATA
F296            3784  SCROLL_UP       PROC    NEAR
F296 8AD8       3785          MOV     BL,AL                   ; SAVE LINE COUNT IN BL
F298 80FC04     3786          CMP     AH,4                    ; TEST FOR GRAPHICS MODE
F29B 7208       3787          JC      N1                      ; HANDLE SEPARATELY
F29D 80FC07     3788          CMP     AH,7                    ; TEST FOR BW CARD
F2A0 7403       3789          JE      N1
F2A2 E9F001     3790          JMP     GRAPHICS_UP
F2A5            3791  N1:                                     ; UP_CONTINUE
F2A5 53         3792          PUSH    BX                      ; SAVE FILL ATTRIBUTE IN BH
F2A6 8BC1       3793          MOV     AX,CX                   ; UPPER LEFT POSITION
F2A8 E83700     3794          CALL    SCROLL_POSITION         ; DO SETUP FOR SCROLL
F2AB 7431       3795          JZ      N7                      ; BLANK_FIELD
F2AD 03F0       3796          ADD     SI,AX                   ; FROM ADDRESS
F2AF 8AE6       3797          MOV     AH,DH                   ; # ROWS IN BLOCK
F2B1 2AE3       3798          SUB     AH,BL                   ; # ROWS TO BE MOVED
F2B3            3799  N2:                                     ; ROW_LOOP
F2B3 E87200     3800          CALL    N10                     ; MOVE ONE ROW
F2B6 03F5       3801          ADD     SI,BP
F2B8 03FD       3802          ADD     DI,BP                   ; POINT TO NEXT LINE IN BLOCK
F2BA FECC       3803          DEC     AH                      ; COUNT OF LINES TO MOVE
F2BC 75F5       3804          JNZ     N2                      ; ROW_LOOP
F2BE            3805  N3:                                     ; CLEAR_ENTRY
F2BE 58         3806          POP     AX                      ; RECOVER ATTRIBUTE IN AH
F2BF B020       3807          MOV     AL,' '                  ; FILL WITH BLANKS
F2C1            3808  N4:                                     ; CLEAR_LOOP
F2C1 E86D00     3809          CALL    N11                     ; CLEAR THE ROW
F2C4 03FD       3810          ADD     DI,BP                   ; POINT TO NEXT LINE
F2C6 FECB       3811          DEC     BL                      ; COUNTER OF LINES TO SCROLL
F2C8 75F7       3812          JNZ     N4                      ; CLEAR_LOOP
F2CA            3813  N5:                                     ; SCROLL_END
F2CA E88C07     3814          CALL    DDS
F2CD BD3E49D007 3815          CMP     CRT_MODE,7              ; IS THIS THE BLACK AND WHITE CARD
F2D2 7407       3816          JE      N6                      ; IF SO, SKIP THE MODE RESET
F2D4 A06500     3817          MOV     AL,CRT_MODE_SET         ; GET THE VALUE OF THE MODE SET
F2D7 BAD803     3818          MOV     DX,03D8H                ; ALWAYS SET COLOR CARD PORT
F2DA EE         3819          OUT     DX,AL
F2DB            3820  N6:                                     ; VIDEO_RET_HERE
F2DB E9E7FE     3821          JMP     VIDEO_RETURN
F2DE            3822  N7:                                     ; BLANK_FIELD
F2DE 8ADE       3823          MOV     BL,DH                   ; GET ROW COUNT
F2E0 EB0C       3824          JMP     N3                      ; GO CLEAR THAT AREA
                3825  SCROLL_UP       ENDP
```

Commentaires.- 1^o) Puisqu'on va effectuer des déplacements, les registres de segments DS et ES sont initialisés (à la méme valeur, celle de la zone de communication du BIOS), et le registre de segment de code au segment contenant le BIOS (ligne 3783).

- 2^o) Puisque le registre AX va être utilisé, le nombre de lignes à faire défiler, passé en paramètre grâce au registre AL, est placé dans le registre BL (ligne 3785).

- 3^o) On ne va traiter ici que le mode texte de la carte MDA et les modes texte de la carte CGA (lignes 3785 à 3789), les modes graphiques étant traités à part (ligne 3790, renvoyant à la ligne 4226).

- 4^o) Pour les modes texte donc, on sauvegarde l'attribut et le nombre de lignes (ligne 3792), puisqu'on va utiliser le registre BX.

- 5^o) On détermine les paramètres du défilement en faisant appel à la sous-routine SCROLL_POSITION (lignes 3793 et 3794), définie à partir de la ligne 3826 :

```
                3826
                3827   ;----- HANDLE COMMON SCROLL SET UP HERE
                3828
F2E2            3829   SCROLL_POSITION PROC   NEAR
F2E2 803E49D002 3830           CMP     CRT_MODE,2              ; TEST FOR SPECIAL CASE HERE
F2E7 7218       3831           JB      N9                      ; HAVE TO HANDLE 80X25 SEPARATELY
F2E9 803E490003 3832           CMP     CRT_MODE,3
F2EE 7711       3833           JA      N9
                3834
                3835   ;----- 80X25 COLOR CARD SCROLL
                3836
F2F0 52         3837           PUSH    DX
F2F1 BADA03     3838           MOV     DX,3DAH                 ; GUARANTEED TO BE COLOR CARD HERE
F2F4 50         3839           PUSH    AX
F2F5            3840   N8:                                     ; WAIT_DISP_ENABLE
F2F5 EC         3841           IN      AL,DX                   ; GET PORT
F2F6 AB08       3842           TEST    AL,8                    ; WAIT FOR VERTICAL RETRACE
F2F8 74FB       3843           JZ      N8                      ; WAIT_DISP_ENABLE
F2FA B0D8       3844           MOV     AL,25H
F2FC B2D8       3845           MOV     DL,0D8H                 ; DX=3D8
F2FE EE         3846           OUT     DX,AL                   ; TURN OFF VIDEO
F2FF 58         3847           POP     AX                      ; DURING VERTICAL RETRACE
F300 5A         3848           POP     DX
F301            3849   N9:
F301 E881FF     3850           CALL    POSITION                ; CONVERT TO REGEN POINTER
F304 03064E00   3851           ADD     AX,CRT_START            ; OFFSET OF ACTIVE PAGE
F308 88F8       3852           MOV     DI,AX                   ; TO ADDRESS FOR SCROLL
F30A 88F0       3853           MOV     SI,AX                   ; FROM ADDRESS FOR SCROLL
F30C 2BD1       3854           SUB     DX,CX                   ; DX = # ROWS, #COLS
F30E FEC6       3855           INC     DH
F310 FEC2       3856           INC     DL                      ; INCREMENT FOR 0 ORIGIN
F312 32ED       3857           XOR     CH,CH                   ; SET HIGH BYTE OF COUNT TO ZERO
F314 8B2E4A00   3858           MOV     BP,CRT_COLS             ; GET NUMBER OF COLUMNS IN DISPLAY
F318 03ED       3859           ADD     BP,BP                   ; TIMES 2 FOR ATTRIBUTE BYTE
F31A 8AC3       3860           MOV     AL,BL                   ; GET LINE COUNT
F31C F6264A00   3861           MUL     BYTE PTR CRT_COLS       ; DETERMINE OFFSET TO FROM ADDRESS
F320 03CD       3862           ADD     AX,AX                   ; *2 FOR ATTRIBUTE BYTE
F322 06         3863           PUSH    ES                      ; ESTABLISH ADDRESSING TO REGEN BUFFER
F323 1F         3864           POP     DS                      ;  FOR BOTH POINTERS
F324 80FB00     3865           CMP     BL,0                    ; 0 SCROLL MEANS BLANK FIELD
F327 C3         3866           RET                             ; RETURN WITH FLAGS SET
                3867   SCROLL_POSITION ENDP
```

a) Dans le cas de la carte CGA en mode 40 caractètes par ligne, le défilement ne peut s'effectuer que durant un rebroussement vertical. Dans ce cas (lignes 3830 à 3833), puisqu'on va alors effectuer une lecture sur le registre d'état de la carte CGA pour déterminer le début d'un tel rebroussement, on commence par sauvegarder le contenu en cours du registre DX (lignes 3837 et 3848, puisque celui-ci est le seul registre à pouvoir être utilisé pour une lecture sur un périphérique) puis on l'initialise à la valeur du port du registre d'état de la carte CGA (ligne 3838). De même, puisque AL est le seul registre à pouvoir être utilisé, on sauvegarde le contenu du registre AX (lignes 3839 et 3847). On attend un rebroussement vertical (lignes 3840 à 3843). On arrête l'affichage durant le défilement en mémoire graphique (lignes 3844 à 3846).

b) Rappelons que la sous-routine POSITION, appelée ligne 3850, détermine, à partir des coordonnées (numéro de colonne dans AL et numéro de ligne dans AH), le décalage dans la page, placé également dans AX. Le registre AX contient ici au départ les coordonnées du coin supérieur gauche de la fenêtre ; on a donc, après appel de la sous-routine, le déplacement dans AX.

En lui ajoutant le décalage de la page active (ligne 3851), on obtient le décalage en mémoire graphique.

c) Les origines de la destination et de la source sont, pour l'instant, égales à ce déplacement (ligne 3852).

On place dans le registre DX le nombre de lignes et de colonnes à déplacer (lignes 3854 à 3856), plus précisément DH contient la différence entre la ligne supérieure plus un et la ligne inférieure et DL la différence entre la colonne de droite et celle de gauche plus un de la fenètre.

En mettant à zéro CH (ligne 3857),

d) On place dans BP (lignes 3858 et 3859) le double du nombre de colonnes de l'écran, ce qui correspond à l'incrément nécessaire après chaque défilement de ligne.

On place dans le registre AX (lignes 3860 à 3862) le double du nombre de lignes à faire défiler fois le nombre de colonnes de l'écran.

Le drapeau ZF contient 0 s'il n'y a pas réellement de défilement à effectuer (ligne 3865), mais plutôt à effacer l'écran.

- 6^o) Le défilement est constitué de deux étapes : déplacer des lignes vers le haut et effacer les lignes du bas. S'il n'y a qu'à effacer l'écran (ligne 3795), on initialise le nombre de lignes à effacer (ligne 3823) et on passe directement à la seconde phase.

- 7^o) S'il y a des lignes à faire déplacer (ligne 3795), on ajoute le contenu du registre AX ayant été calculé dans SCROLL_POSITION à SI (ligne 3796) et on place dans AH le nombre de lignes dans le bloc moins le nombre de lignes à déplacer (lignes 3797 et 3798).

- 8^o) Le déplacement d'une ligne fait l'objet de la sous-routine N10, définie à partir de la ligne 3868 :

```
                3868
                3869   ;----- MOVE_ROW
                3870
F328            3871   N10     PROC    NEAR
F328 8ACA       3872           MOV     CL,DL                   ; GET # OF COLS TO MOVE
F32A 56         3873           PUSH    SI
F32B 57         3874           PUSH    DI                      ; SAVE START ADDRESS
F32C F3         3875           REP     MOVSW                   ; MOVE THAT LINE ON SCREEN
F32D A5
F32E 5F         3876           POP     DI
F32F 5E         3877           POP     SI                      ; RECOVER ADDRESSES
F330 C3         3878           RET
                3879   N10     ENDP
```

On place le nombre de colonnes à déplacer (puisque la ligne du bloc n'occupe pas nécessairement toute la ligne de l'écran) dans le compteur CL (ligne 3872). Les adresses de la source et de destination sont sauvegardées (lignes 3873, 3874, 3876 et 3877) pour les mêmes raisons ; elles seront modifiées « à la main » ensuite. Il suffit alors d'utiliser une instruction MOVSW (ligne 3875).

- 9^o) On modifie « à la main » les contenus des registres SI et DI (lignes 3801 et 3802). On décrémente le nombre de lignes à faire défiler (ligne 3803) et on recommence à faire défiler une ligne jusqu'à ce que ce nombre soit nul (ligne 3804).

- 10^o) On remplit ensuite les lignes du bas par le caractère blanc, ce qui revient à effacer ces lignes. On récupère l'attribut (ligne 3806) et le caractère blanc (ligne 3807).

- 11^o) Pour remplir une ligne de blancs sur la largeur voulue, on fait appel à la sous-routine N11, définie à partir de la ligne 3880 :

```
                3880
                3881   ;----- CLEAR_ROW
                3882
F331            3883   N11     PROC    NEAR
F331 8ACA       3884           MOV     CL,DL                   ; GET # COLUMNS TO CLEAR
F333 57         3885           PUSH    DI
F334 F3         3886           REP     STOSW                   ; STORE THE FILL CHARACTER
F335 AB
F336 5F         3887           POP     DI
F337 C3         3888           RET
                3889   N11     ENDP
```

On place le nombre de colonne à remplir (puisque la ligne du bloc n'occupe pas nécessairement toute la ligne de l'écran) dans le compteur CL (ligne 3884). L'adresse de destination est sauvegardée (lignes 3885 et 3887) pour les mêmes raisons ; elle sera modifiée « à la main » ensuite. Il suffit alors d'utiliser l'instruction STOSW (ligne 3886).

- 12^o) On modifie « à la main » le contenu du registre DI (ligne 3810). On décrémente le nombre de lignes à effacer (ligne 3811) et on recommence à effacer une ligne jusqu'à ce que ce nombre soit nul (ligne 3812).

- 13^o) On revient au segment de la zone de communication du BIOS (ligne 3814). S'il s'agit de la carte CGA (lignes 3815 et 3816), on communique le mode à la carte (lignes 3817 à 3819), dont l'affichage avait été interrompu.

- 14^o) On termine la procédure nomalement (ligne 3821).

7.5.15 Défilement vers le bas

La fonction SCROLL_DOWN permet un défilement vers le bas. Le début du code de cette fonction commence ligne 3890 :

```
                3890   ;--------------------------------------------------------------
                3891   ; SCROLL_DOWN                                                 :
                3892   ;       THIS ROUTINE MOVES THE CHARACTERS WITHIN A            :
                3893   ;       DEFINED BLOCK DOWN ON THE SCREEN, FILLING THE         :
                3894   ;       TOP LINES WITH A DEFINED CHARACTER                    :
                3895   ; INPUT                                                       :
                3896   ;       (AH) = CURRENT CRT MODE                               :
                3897   ;       (AL) = NUMBER OF LINES TO SCROLL                      :
                3898   ;       (CX) = UPPER LEFT CORNER OF REGION                    :
                3899   ;       (DX) = LOWER RIGHT CORNER OF REGION                   :
                3900   ;       (BH) = FILL CHARACTER                                 :
                3901   ;       (DS) = DATA SEGMENT                                   :
                3902   ;       (ES) = REGEN SEGMENT                                  :
                3903   ; OUTPUT                                                      :
                3904   ;       NONE -- SCREEN IS SCROLLED                            :
                3905   ;--------------------------------------------------------------
F338            3906   SCROLL_DOWN     PROC    NEAR
F338 FD         3907           STD                             ; DIRECTION FOR SCRLL DOWN
F339 8AD8       3908           MOV     BL,AL                   ; LINE COUNT IN BL
F33B 80FC04     3909           CMP     AH,4                    ; TEST FOR GRAPHICS
F33E 7208       3910           JC      N12
F340 80FC07     3911           CMP     AH,7                    ; TEST FOR BW CARD
F343 7403       3912           JE      N12
F345 E9A601     3913           JMP     GRAPHICS_DOWN
F348            3914   N12:                                    ; CONTINUE_DOWN
F348 53         3915           PUSH    BX                      ; SAVE ATTRIBUTE IN BH
F349 8BC2       3916           MOV     AX,DX                   ; LOWER RIGHT CORNER
F34B E894FF     3917           CALL    SCROLL_POSITION         ; GET REGEN LOCATION
F34E 742D       3918           JZ      N16
F350 2BF0       3919           SUB     SI,AX                   ; SI IS FROM ADDRESS
F352 8AE6       3920           MOV     AH,DH                   ; GET TOTAL # ROWS
F354 2AE3       3921           SUB     AH,BL                   ; COUNT TO MOVE IN SCROLL
F356            3922   N13:
F356 E8CFFF     3923           CALL    N10                     ; MOVE ONE ROW
F359 2BF5       3924           SUB     SI,BP
F35B 2BFD       3925           SUB     DI,BP
F35D FECC       3926           DEC     AH
F35F 75F5       3927           JNZ     N13
F361            3928   N14:
F361 58         3929           POP     AX                      ; RECOVER ATTRIBUTE IN AH
F362 B020       3930           MOV     AL,' '
F364            3931   N15:
F364 E8CAFF     3932           CALL    N11                     ; CLEAR ONE ROW
F367 2BFD       3933           SUB     DI,BP                   ; GO TO THE NEXT ROW
F369 FECB       3934           DEC     BL
F36B 75F7       3935           JNZ     N15
F36D E95AFF     3936           JMP     N5                      ; SCROLL_END
F370            3937   N16:
F370 8ADE       3938           MOV     BL,DH
F372 EBED       3939           JMP     N14
                3940   SCROLL_DOWN     ENDP
```

Commentaires.- 1^o) On peut s'apercevoir du projet en équipe sur cet exemple : les commentairers du début et ligne par ligne sont différents de la fonction précédente pour des fonctions analogues.

- 2^o) Puisqu'on va utiliser le registre AX, le nombre de lignes à faire défiler, passé en paramètre grâce au registre AL, est placé dans le registre BL (ligne 3908).

- 3^o) On ne traite ici que le mode texte de la carte MDA et les modes texte de la carte CGA (lignes 3909 à 3912), les modes graphiques étant traités à part (ligne 3913, renvoyant à la ligne 4310).

- 4^o) Pour les modes texte donc, on sauvegarde l'attribut et le nombre de lignes (ligne 3915), puisqu'on va utiliser le registre BX.

- 5^o) On détermine les paramètres du défilement en faisant appel à la sous-routine SCROLL_POSITION (lignes 3916 et 3917).

- 6^o) Le défilement s'effectue en deux étapes : déplacer des lignes vers le bas et effacer les lignes du haut. S'il ne s'agit que d'effacer l'écran (ligne 3918), on initialise le nombre de lignes à effacer (ligne 3939) et on passe directement à la seconde étape.

- 7^o) Si, par contre, il y a réellement des lignes à déplacer (ligne 3918), on ajoute le contenu du registre AX ayant été calculé dans SCROLL_POSITION à SI (ligne 3919) et on place dans AH le nombre de lignes dans le bloc moins le nombre de lignes à déplacer (lignes 3920 et 3921).

- 8^o) On déplace une ligne en faisant appel à la sous-routine N10 déjà étudiée.

- 9^o) On modifie « à la main » les contenus des registres SI et DI (lignes 3924 et 3925). On décrémente le nombre de lignes à faire défiler (ligne 3926) et on recommence à déplacer une ligne jusqu'à ce que ce nombre soit nul (ligne 3927).

- 10^o) On va maintenant remplir les lignes du haut par le caractère blanc, ce qui revient à effacer ces lignes. Pour cela, on récupère l'attribut (ligne 3929) et le caractère blanc (ligne 3930).

- 11^o) On remplit une ligne de blancs sur la largeur voulue en faisant appel à la sous-routine N11 déjà étudiée.

- 12^o) On modifie « à la main » le contenu du registre DI (ligne 3933). On décrémente le nombre de lignes à effacer (ligne 3934) et on recommence à effacer une ligne jusqu'à ce que ce nombre soit nul (ligne 3935).

- 13^o) On termine comme dans le cas du défilement vers le haut (ligne 3936).

7.5.16 Lecture du caractère affiché à la position du curseur

La fonction READ_AC_CURRENT permet de lire le caractère affiché à la position du curseur. Le début du code de cette fonction commence ligne 3941 :

```
                3941   ;--------------------------------------------------------------
                3942   ; READL_AC_CURRENT                                           :
                3943   ;       THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER        :
                3944   ;       AT THE CURRENT CURSOR POSITION AND RETURNS THEM       :
                3945   ;       TO THE CALLER                                        :
                3946   ; INPUT                                                      :
                3947   ;       (AH) = CURRENT CRT MODE                              :
                3948   ;       (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )             :
                3949   ;       (DS) = DATA SEGMENT                                  :
                3950   ;       (ES) = REGEN SEGMENT                                 :
                3951   ; OUTPUT                                                     :
                3952   ;       (AL) = CHAR READ                                     :
                3953   ;       (AH) = ATTRIBUTE READ                                :
                3954   ;--------------------------------------------------------------
                3955           ASSUME  CS:CODE,DS:DATA,ES:DATA
F374            3956   READ_AC_CURRENT PROC    NEAR
F374 80FC04     3957           CMP     AH,4                    ; IS THIS GRAPHICS
F377 7208       3958           JC      P1
F379 80FC07     3959           CMP     AH,7                    ; IS THIS BW CARD
F37C 7403       3960           JE      P1
F37E E9A802     3961           JMP     GRAPHICS_READ
F381            3962   P1:                                     ; READ_AC_CONTINUE
F381 E81A00     3963           CALL    FIND_POSITION
F384 88F3       3964           MOV     SI,BX                   ; ESTABLISH ADDRESSING IN SI
                3965
                3966   ;----- WAIT FOR HORIZONTAL TRACE
                3967
F386 8B166300   3968           MOV     DX,ADDR_6845            ; GET BASE ADDRESS
F38A 83C206     3969           ADD     DX,6                    ; POINT AT STATUS PORT
F38D 06         3970           PUSH    ES
F38E 1F         3971           POP     DS                      ; GET SEGMENT FOR QUICK ACCESS
F38F            3972   P2:                                     ; WAIT FOR RETRACE LOW
F38F EC         3973           IN      AL,DX                   ; GET STATUS
F390 A801       3974           TEST    AL,1                    ; IS HORZ RETRACE LOW
F392 75FB       3975           JNZ     P2                      ; WAIT UNTIL IT IS
F394 FA         3976           CLI                             ; NO MORE INTERRUPTS
F395            3877   P3:                                     ; WAIT FOR RETRACE HIGH
F395 EC         3878           IN      AL,DX                   ; GET STATUS
F396 A801       3979           TEST    AL,1                    ; IS IT HIGH
F398 74FB       3980           JZ      P3                      ; WAIT UNTIL IT IS
F39A AD         3981           LODSW                           ; GET THE CHR/ATTR
F39B E927FE     3982           JMP     VIDEO_RETURN
                3984   READ_AC_CURRENT ENDP
```

Commentaires.- 1^o) On replace le code au bon endroit (ligne 3955).

- 2^o) Dans le cas d'un mode graphique de la carte CGA (lignes 3957 à 3960), on renvoie à une sous-routine particulière (ligne 3961), commençant ligne 4584, qui ne nous intéresse ici.

- 3^o) On détermine le déplacement dans la mémoire graphique du caractère voulu, que l'on place dans DI grâce à l'appel de la sous-routine FIND_POSITION (ligne 3963).

On place le numéro de page dans SI (ligne 3964).

- 4^o) On attend un rebroussement vertical (lignes 3965 à 3975).

- 5^o) On ne permet pas aux interruptions de prendre la main (ligne 3976), on attend un rebroussement vertical (lignes 3877 à 3980), on charge le caractère (ligne 3981) et on termine la fonction normalement (ligne 3982).

7.6 Historique

7.6.1 Tube cathodique

Le physicien et chimiste anglais Michael FARADAY (1791–1867) note en 1833 que, dans un tube contenant des électrodes, lorsque la quantité d'air diminue, on aperçoit une faible lueur entre les électrodes : « la raréfaction de l'air favorise le phénomène d'incandescence » (*Recherches expérimentales sur l'électricité*). Il examine la luminescence dans différents gaz sous basse pression. Il décrit la beauté de cette incandescence et observe la zone obscure près de l'anode qui porte maintenant son nom.

Une équipe allemande composée de Heinrich GEISSLER (1815–1879) et de Julius PLÜCKER (1801–1868) initie l'étude plus approfondie de ce phénomène. GEISSLER est un verrier très habile, employé par l'université de Bonn comme fabricant d'instruments scientifiques. Il rencontre PLÜCKER, alors jeune professeur. Vers 1855, PLÜCKER demande à GEISSLER de concevoir un appareil pour évacuer presque complètement l'air d'un tube de verre. GEISSLER construit une pompe à mercure à manivelle. Ces nouveaux tubes à vide deviennent très populaires et sont appelés « tubes de Geissler ».

Utilisant ce tube à vide amélioré, PLÜCKER fait quelques découvertes surprenantes. En premier lieu, il est capable de produire une lueur brillante ressemblant à un jet entre les électrodes. La lueur est beaucoup plus brillante que tout ce qu'on avait vu jusque-là. En second lieu, il a l'idée d'approcher un aimant du tube à vide pour voir ce qui arrive au jet et découvre ainsi qu'il est dévié par un puissant champ magnétique [Plu-58]. Cette découverte tente à montrer que le jet est composé de particules et non de rayons. L'année suivante, il rapporte avoir vu une brillante phosphorescence verte dans le verre du tube près de la cathode et qu'il peut changer la position des taches phosphorescentes en utilisant un aimant. Cependant il ne peut aller plus loin, son vide étant par trop insuffisant.

Étudiant de PLÜCKER, Johann HITTORF (1824–1914) améliore grandement la méthode pour créer du vide dans un tube de verre. Il observe en 1869 que la lueur augmente fortement lorsque la pression dans le tube diminue. Il place de petits obstacles entre les deux électrodes du tube. Lorsqu'un courant est établi, la lueur peut être partiellement obscurcie par ces obstacles, donnant lieu à des ombres. Ce phénomène renforce l'idée que le jet est causé par une émission de particules.

Le savant allemand Eugen GOLDSTEIN (1850–1930) baptise « rayons cathodiques » ces rayons en 1876.

Afin de confirmer les expériences de PLÜCKER et de HITTORF, le physicien anglais William CROOKES (1832–1919) conçoit son propre tube à vide dans lequel l'air est pratiquement absent. Il s'agit d'une telle amélioration par rapport aux tubes de Geissler que le « tube de Crookes » devient rapidement le standard des tubes à vide dans les expériences scientifiques. CROOKES continue les expériences de Plücker sur les champs magnétiques, confirmant que le jet est facilement dévié. Dans *On radiant matter*, conférence à la *British Association for the Advancement of Science* à Sheffield, le vendredi 22 août 1879, CROOKES montre 21 tubes différents (chaque tube porte un numéro spécifique) et parle d'un *quatrième* état de la matière, le *plasma*. On en retient en général le tube numéro 9 (celui à la croix de Malte) (figure 5 de [Cro-96]) et la petite girouette qu'il a installé dans un de ses tubes (figure 7).

En 1892, Heinrich HERTZ (1857–1894) fait état d'une expérience censée prouver que les rayons cathodiques ne peuvent pas être des particules et doivent donc être des ondes : il montre que ces rayons peuvent traverser une fine paroi de métal sans y percer de trous. Il fait également passer un faisceau de rayons cathodiques entre deux plaques parallèles, l'une ayant une charge électrique positive et l'autre négative. On peut observer ce faisceau grâce aux vives luminescences produites sur certaines parties de la paroi en verre. Ne détectant aucune déviation de ce faisceau,

Hertz en conclut qu'il s'agit bien d'ondes.

Phillip Lenard (1862–1947) poursuit les travaux de Hertz sur le passage des rayons cathodiques à traver une fine fenêtre de métal sur l'un des côtés d'un tube de Crookes. Il prouve que les rayons cathodiques ne sont pas un phénomène exclusif du vide [Len-06].

En effectuant une expérience analogue en 1895, le physicien allemand Wilhelm Roentgen (1845–1923) découvre accidentellement une forme de radiation beaucoup plus pénétrante, qu'il appelle rayons X, mais c'est une autre histoire.

12. *Ueber ein Verfahren zur Demonstration und zum Studium des zeitlichen Verlaufes variabler Ströme; von Ferdinand Braun.*

1. Die im Folgenden beschriebene Methode benutzt die Ablenkbarkeit der Kathodenstrahlen durch magnetische Kräfte. Diese Strahlen wurden in Röhren erzeugt, von deren einer ich die Maasse angebe, da mir diese die im allgemeinen günstigsten zu sein scheinen (Fig. 1). *K* ist die Kathode aus Aluminiumblech, *A* Anode, *C* ein Aluminiumdiaphragma; Oeffnung des Loches = 2 mm. *D* ein mit phosphorescirender Farbe überzogener Glimmerschirm. Die Glaswand *E* muss möglichst gleichmässig und ohne Knoten, der phosphorescirende Schirm

Fig. 1.

Figure 7.12 – Annonce de l'invention de l'oscilloscope en février 1897

Alors que beaucoup de savants essaient de lever les secrets des rayons cathodiques, d'autres recherchent des façons de les appliquer à des fins pratiques. La première application est le tube à rayons cathodiques (CRT pour *Cathodic Ray Tube*), utilisé comme oscilloscope, construit en 1897 par Karl Ferdinand Braun (prix Nobel en 1909), qui travaille de 1894 à sa mort à l'institut de physique de l'université de Strasbourg. Cet appareil utilise un tube à rayons cathodiques pour produire une luminescence sur un écran traité chimiquement. Les rayons cathodiques passent à travers une petite ouverture, ce qui permet de les faire converger en un faisceau apparaissant comme un point sur l'écran. Le point décrit l'écran selon la fréquence du signal d'entrée. Un observateur regardant l'écran de l'oscilloscope voit alors une représentation visuelle du courant alternatif de Strasbourg, ce qui n'était pas possible autrement.

Les améliorations techniques suivent : c'était au début grâce à un système de miroirs tournants qu'on pouvait voir la sinusoïde ; Braun et son assistant Jonathan Zenneck inventent la *base de temps* qui permet d'observer directement la sinusoïde à l'écran, puis différents dispositifs permettant d'améliorer la netteté de l'image. Étonnamment, cette invention n'est pas très utilisée dans les années qui suivent ; les oscilloscopes apparaissent dans les laboratoires dans l'entre-deux guerres.

Ces tubes avaient tous une cathode froide. Le premier tube à rayons cathodiques à utiliser une cathode chaude est conçu par John B. Johnson (1887–1970) (le même qui donna son nom au *bruit de Johnson*) et Harry Weiner Weinhart de la *Western Electric*. Il devient un produit commercial en 1922.

Figure 7.13 – Le tube de Braun est un tube de Crookes ayant un écran de mica interne recouvert d'une peinture phosphorescente

Durant les trois premières décades du vingtième siècle, les inventeurs ont continué à trouver des utilisations aux rayons cathodiques. Inspiré par l'oscilloscope de Braun, A. A. Campbell-Swinton suggère qu'on pourrait utiliser un tube à rayons cathodiques pour projeter une image animée sur un écran. Malheureusement, la technologie de l'époque ne permet pas de mettre en place l'idée de Campbell-Swinton. Ce n'est qu'en 1922 que Philo T. Farnsworth utilise un électro-aimant pour faire converger un faisceau d'électrons sur un écran, permettant de faire apparaître une image grossière. Cet appareil est cependant très rapidement remplacé par le *kinescope* de Vladimir Zworykin, l'ancêtre de la télévision moderne.

En 1907, le savant russe Boris Rosing (qui travaillait avec Vladimir Zworykin) utilise un tube à rayons cathodiques comme receveur d'un système de télévision dont la caméra utilise un miroir tournant.

Le premier ensemble de télévision commercial utilisant des tubes à rayons cathodiques est fabriqué par *Telefunken* en Allemagne en 1934.

7.7 Bibliographie

[Bra-97] Braun, Ferdinand, **Annalen der Physik und Chemie**, 1897. Engl. tr. *Over a procedure for the demonstration and for the study of the course of the variable currents* :

`http://www.crtsite.com/Annalen\%20der\%20Physik.html`

[Cro-96] Crookes, William, *The true catode rays*, **The New York Times**, February 23, 1896 :

`http://query.nytimes.com/gst/abstract.html?res=9406EEDD123EE333A25750C2A9649C94679ED7CF`

[Len-06] Lenard, Philipp, *Über Kathodenstrahlen* (Sur les rayons cathodiques), 1906, 44 p. Numérisé par *Google*.

[Plu-58] Plücker, Julius, *Ueber die Einwirkung des Magneten auf die elektrischen Entladungen in verdünnten Gasen*, **Poggendorffs annalen der Physik und Chemie**, 1858, t. 179, pp 88–106. Téléchargeable sur *gallica.bnf.fr*.