Framebuffer HOWTO

Alex Buell, alex.buell@tahallah.clara.co.uk

v1.2, 27 Février 2000

Ce document décrit l'emploi des périphériques d'accès à la mémoire vidéo avec diverses configurations matérielles munies de Linux. La gestion de plusieurs écrans est également traitée.

Contents

1 History

Revision history

19990607 - Release of 1.0

19990722 - Release of 1.1

20000222 - Release of 1.2

2 Contributions

Merci aux personnes dont les noms suivent pour avoir aidé à l'amélioration du HOWTO Framebuffer.

- Jeff Noxon `jeff@planetfall.com`
- Francis Devereux `f.devereux@cs.ucl.ac.uk`
- Andreas Ehliar `ehliar@futurniture.se`
- Martin McCarthy `marty@ehabitat.demon.co.uk`
- Simon Kenyon `simon@koala.ie`
- David Ford `david@kalifornia.com`
- Chris Black `cblack@cmpteam4.unil.ch`
- N Becker `nbecker@fred.net`
- Bob Tracy `rct@gherkin.sa.wlk.com`
- Marius Hjelle `marius.hjelle@roman.uib.no`
- James Cassidy `jcassidy@misc.dyn.ml.org`
- Andreas U. Trottmann `andreas.trottmann@werft22.com`
- Lech Szychowski `lech7@lech.pse.pl`
- Aaron Tiensivu `tiensivu@pilot.msu.edu`
- Jan-Frode Myklebust pour ses informations sur les cartes Permedia `janfrode@ii.uib.no`
- Et les autres, trop nombreux pour tous figurer ici. Un grand merci à eux.

Un grand merci à Rick Niles frederick.a.niles@gsfc.nasa.gov qui a accepté que son Mini-HOWTO Multi-Head soit inclus dans ce HOWTO. Merci aux personnes suivantes pour avoir compilé les versions libc5/glibc2 du gestionnaire XF86_FBdev pour X11 sur les architectures Intel :

- Brion Vibber brion@pobox.com
- Gerd Knorr kraxel@cs.tu-berlin.de

bien sûr l'auteur du code :

- Martin Schaller - auteur du concept originel de périphérique d'accès à la mémoire vidéo.
- Roman Hodek Roman.Hodek@informatik.uni-erlangen.de
- Andreas Schwab schwab@issan.informatik.uni-dortmund.de
- Guenther Kelleter
- Geert Uytterhoeven Geert.Uytterhoeven@cs.kuleuven.ac.be
- Roman Zippel roman@sodom.obdg.de
- Pavel Machek pavel@atrey.karlin.mff.cuni.cz
- Gerd Knorr kraxel@cs.tu-berlin.de
- Miguel de Icaza miguel@nuclecu.unam.mx
- David Carter carter@compsci.bristol.ac.uk
- William Rucklidge wjr@cs.cornell.edu
- Jes Sorensen jds@kom.auc.dk
- Sigurdur Asgeirsson
- Jeffrey Kuskin jsk@mojave.stanford.edu
- Michal Rehacek michal.rehacek@st.mff.cuni.edu
- Peter Zaitcev zaitcev@lab.ipmce.su
- David S. Miller davem@dm.cobaltmicro.com
- Dave Redman djhr@tadpole.co.uk
- Jay Estabrook
- Martin Mares mj@ucw.cz
- Dan Jacobowitz dan@debian.org
- Emmanuel Marty core@ggi-project.org
- Eddie C. Dost ecd@skynet.be
- Jakub Jelinek jj@ultra.linux.cz
- Phil Blundell philb@gnu.org
- S'il y en a d'autres, qu'ils se manifestent et ils seront cités. :o)

3 Qu'est-ce qu'un tampon de mémoire vidéo ?

Un tampon de mémoire vidéo définit une abstraction logicielle d'accès aux périphériques vidéo. Il correspond à la mémoire d'affichage de certains contrôleurs graphiques et propose une interface unifiée aux logiciels qui n'ont alors plus à se soucier des détails de bas niveau relatifs au matériel [Extrait du fichier framebuffer.txt écrit par Geert Uytterhoeven's. Se reporter aux sources du noyau]

4 Quels avantages présente le tampon de mémoire vidéo ?

Le logo ! Plus sérieusement, on dispose d'une interface indépendante de l'architecture matérielle. Les gestionnaires de console des machines de type Intel sont restés radicalement différents de ceux des autre plate-formes jusqu'à une phase de développement avancée des noyaux 2.1.x. Avec l'introduction dans le noyau 2.1.109 de cette interface, les choses se sont améliorées : la gestion des consoles sur PC s'est uniformisée, les consoles en mode graphique affichant le logo du pingouin ont fait leur apparition et le code s'est propagé aux autres types de machines. Notez que les noyaux 2.0.x ne disposent pas du gestionnaire d'accès à la mémoire vidéo. Peut-être quelqu'un finira-t-il par intégrer le code des versions 2.1.x dans ces noyaux. Le portage version 0.9.x pour les machines m68k font exception en ce qu'elles intègrent le pilote. *Avec la disponibilité des noyaux 2.2.x, le gestionnaire de mémoire vidéo s'avère stable et robuste. Vous devriez l'utiliser si votre carte vidéo le supporte et si vous employez un noyau 2.2.x. La question ne se pose pas si vous travaillez avec un 2.0.x, du moins sur un PC.*

- 0.9.x (m68k) - introduction du gestionnaire. Notez que les versions 0.9.x équivalent d'un point de vue fonctionnel à la version 1.0.9 sur architecture Intel avec les améliorations des 1.2.x.
- 2.1.107 - apparition du gestionnaire de mémoire vidéo sur PC ainsi que des nouveaux pilotes pour les consoles. Le défilement en arrière n'est pas encore disponible.
- 2.1.113 - ajout du défilement en arrière au pilote vgacon.
- 2.1.116 - ajout du défilement en arrière au pilote vesafb.
- 2.2.x - matroxfb et atyfb (cartes graphiques Matrox et ATI respectivement).

Le gestionnaire de mémoire vidéo offre des possibilités intéressantes si on précise quelques options au noyau lors du démarrage. Certaines sont spécifiques à un type de carte donné.

- `video=xxx:off` - désactive l'auto-détection d'un pilote
- `video=map:octal-number` - associe des consoles virtuelles (VC) à un gestionnaire de mémoire vidéo
  - `video=map:01` VC0 est associée à FB0, VC1 à FB1, VC2 à FB0, VC3 à FB1...
  - `video=map:0132` VC0 est associée à FB0, VC1 à FB1, VC2 à FB3, VC4 à FB2, VC5 à FB0...

La détection des gestionnaires de mémoire vidéo a lieu dans un ordre fixé au niveau du noyau. Vous pouvez l'altérer grâce à l'option `video=xxx` qui permet de forcer la détection de certains périphériques avant les autres.

5 Utilisation du tampon de mémoire vidéo sur architecture Intel

5.1 Vesafb, quès acco ?

Vesafb est un gestionnaire de mémoire vidéo sur compatible PC dédié aux cartes cartes graphiques conformes aux spécifications VESA 2.0. Son fonctionnement est lié de près aux gestionnaires de mémoire vidéo génériques du noyau.

Vesafb permet le recours aux modes graphiques sur PC pour l'utilisation des consoles textes en point par point. Vesafb autorise également l'affichage d'un logo et c'est vraisemblablement ce pour quoi vous voulez vous en servir :o)

On ne peut malheureusement pas utiliser vesafb avec des cartes VESA 1.2. En effet, ces cartes n'utilisent pas un adressage linéaire. Par ce terme, on entend que tous les octets de la mémoire vidéo sont accessibles à un instant donné. Historiquement, les anciennes cartes vidéo ne rendaient la mémoire graphique disponible qu'au travers d'une fenêtre de 64 ko qui correspondait à la taille de la plus grande zone de mémoire contigüe gérable directement par le microprocesseur (d'où les limitations des cartes CGA/EGA). Quelqu'un écrira peut-être un gestionnaire de périphériques vesafb12 pour ce type de cartes, mais il consommera une mémoire par ailleurs précieuse pour le noyau et ça restera de toute façon un sale bricolage. :o(

Il existe cependant un moyen détourné d'accéder aux fonctionnalités VESA 2.0 sur une carte VESA 1.2. Peut-être pouvez vous charger depuis le DOS un programme de type TSR qui, utilisé conjointement avec loadlin, aidera à configurer la carte pour les modes graphiques voulus. Cela ne marchera pas toujours. Ainsi, certaines cartes de chez Cirrus Logic, telles les VLB 54xx, se retrouvent à une position en mémoire (par exemple entre 15 et 16 Mo) qui en interdit l'utilisation sur les systèmes munis de plus de 32 Mo de mémoire. Rien de rédhibitoire si on dispose d'un BIOS permettant de ne pas affecter de mémoire entre 15 et 16 Mo ("Memory Hole") mais il m'a semblé comprendre que Linux n'aime pas ça. Si l'expérience vous tente, vous pouvez essayer UNIVBE (disponible sur l'Internet).

Vous pouvez aussi essayer divers patches noyaux. Il en existe notamment pour les anciennes cartes S3 telles la S3 Trio ou la Virge qui se conforment à la norme VESA 1.2. Les patches sont disponibles via : >.

5.2 Comment faire fonctionner le gestionnaire vesafb ?

A supposer que vous utilisiez menuconfig, vous devrez passer par les étapes suivantes : Dans le menu "Console drivers" :

- VGA Text Console
- Video Selection Support
- Support for frame buffer devices (experimental)
- VESA VGA Graphic console
- Advanced Low Level Drivers
- Choisissez les gestionnaires Mono, 2bpp, 4bpp, 8bpp, 16bpp, 24bpp et 32bpp .
- VGA character/attributes support

Si votre processeur (de type x86) supporte le MTRR, activez le. Il permet d'accélérer les copies entre la mémoire système et la carte graphique. Vous pouvez naturellement le mettre en marche une fois la console opérationnelle. *IMPORTANT : pour les noyaux 2.1.x, activez le choix des fonctionnalités expérimentales via le menu "Code Maturity Level". Ceci est inutile pour les noyaux 2.2.x.*

- Prompt for development and/or incomplete code/drivers

Le support des composants VGA (en mode texte) - vgafb - appartenait à la liste ci-dessus mais il en a été supprimé en raison de son obsolescence. Il disparaîtra sous peu. Sélectionnez plutôt "VGA Text Console".

Vérifiez bien que le support "Mac variable bpp packed pixel" n'est pas activé. [En 2.2.111/112, il semblerait qu'il le soit si "Advanced Low Level Drivers" l'est. Ce n'est plus le cas en 2.1.113] Les fontes peuvent également être stockées en mémoire mais rien n'y oblige et l'emploi de setfont du paquetage kbd-0.99 reste possible pour charger les fontes adéquates (reportez vous à la section relative aux fontes).

Assurez vous que rien n'est modularisé. [J'ai des doutes quand aux possibilités de modularisation de l'ensemble - corrections bienvenues]

Vous devrez ensuite créer les périphériques associés au gestionnaire de mémoire vidéo dans le répertoire /dev. Pour le premier, il vous suffit de taper

```
# mknod /dev/fb0 c 29 0
```

Les suivants doivent être multiples de 32, soit, pour /dev/fb1 :

```
# mknod /dev/fb1 c 29 32
```

et ainsi de suite jusqu'au huitième si vous le souhaitez :

```
# mknod /dev/fb7 c 29 224
```

Recompilez votre noyau,modifiez l'/etc/lilo.conf de façon à ajouter le paramètre VGA=ASK, lancez lilo. Ceci vous permettra de choisir le mode graphique que vous voulez.

Voici mon lilo.conf personnel :

```
# LILO configuration file
boot = /dev/hda3
delay = 30
prompt
vga = ASK # L'utilisateur devra entrer le mode
image = /vmlinuz
  root = /dev/hda3
  label = Linux
  read-only # Les systemes de fichiers autres que UMSDOS doivent etre montes
            # en lecture seule pour la phase de verification
```

Redémarrez le noyau et essayez comme test d'entrer 0301 au prompt VGA. Vous devriez vous retrouver en 640x480 sur 256 couleurs avec un délicieux petit logo de ping^H^H^H^Hmanchot.

A l'invite LILO, vous DEVEZ fournir un chiffre sous la forme d'un “0” suivi de 3 digits sans “x” hexadécimal. Si LILO fournit directement l'argument au noyau, ceci n'est pas nécessaire.

Une fois que tout marche convenablement, vous pouvez explorer les différents modes (voir plus bas) et une fois choisi celui qui vous convient, il sera temps de le fixer via le paramètre “VGA=x” du fichier de configuration de lilo. La table plus bas vous fournira le nombre correspondant au mode. Par exemple, pour du 1280x1024 en 256 couleurs, vous emploierez “VGA=0x307” et relancerez lilo. Le reste se trouve dans les HOWTO relatifs à lilo et à loadlin.

NOTE ! vesafb n'active pas automatiquement le défilement vers l'arrière. Vous devez le préciser au noyau : video=vesa:ypan ou video=vesa:ywrap. Les deux font la même chose mais de façon un peu différente. ywrap est bien plus rapide qu'ypan mais risque de ne pas fonctionner sur des cartes VESA 2.0 ne respectant pas tout à fait les spécifications. L'option n'est disponible qu'à partir du noyau 2.1.116. Les noyaux précédents ne permettent pas le défilement vers l'arrière.

6 De quels modes VESA puis-je me servir ?

Cela dépend de votre carte graphique, en particulier de la quantité de mémoire dont elle dispose. A vous de voir quels sont les modes qui fonctionnent le mieux.

La table suivante fournit les numéros des modes que vous pouvez passer à l'invite VGA (en fait les indices se sont vus ajouter 0x200 afin de s'y retrouver plus facilement dans la table).

Couleurs	640x400	640x480	800x600	1024x768	1152x864	1280x1024	1600x1200
4 bits	?	?	0x302	?	?	?	?
8 bits	0x300	0x301	0x303	0x305	0x161	0x307	0x31C
15 bits	?	0x310	0x313	0x316	0x162	0x319	0x31D
16 bits	?	0x311	0x314	0x317	0x163	0x31A	0x31E
24 bits	?	0x312	0x315	0x318	?	0x31B	0x31F
32 bits	?	?	?	?	0x164	?	

8 bits = 256 couleurs, 15 bits = 32768 couleurs, 16 bits = 65536 couleurs, 24 bits = 16,8 millions de couleurs, 32 bits : la même chose qu'en 24 bits mais les 8 bits restant peuvent servir à diverses fins et l'ensemble s'adapte parfaitement aux bus 32 bits PCI/VLB/EISA. Les modes supplémentaires sont à la discrétion du fabricant puisque la spécification VESA 2.0 s'arrête au mode 0x31f. Il vous faudra sûrement tâtonner pour les trouver.

6.1 Utilisation des cartes graphiques Matrox

Si vous disposez d'une carte Matrox, vous emploierez le pilote matroxfb au lieu de vesafb. Matroxfb gère les Mystique Millenium I, II ainsi que les G100 et G200. Il permet aussi d'avoir plusieurs cartes dans la même machine. La configuration d'une carte Matrox passe par les étapes suivantes :

Mise à jour du BIOS Matrox que vous trouverez à >. Attention, vous aurez besoin du DOS pour procéder à la mise à jour.

Allez dans le menu "Code Maturity Level" et activez l'option suivante :

- Prompt for development and/or incomplete code/drivers

[Ceci peut changer dans les futurs noyaux. Le HOWTO sera alors modifié]

Dans le menu "Console Drivers", sélectionnez :

- VGA Text Console
- Video Selection Support
- Support for frame buffer devices (experimental)
- Matrox Acceleration

- Suivant votre type de carte :
  - Millennium I/II support
  - Mystique support
  - G100/G200 support
- Pour employer plusieurs cartes Matrox simultanément, activez l'option “Multihead support”.
- Advanced Low Level Drivers
- Choisissez les pilotes Mono, 2bpp, 4bpp, 8bpp, 16bpp, 24bpp et “32bpp packed pixel”.

Recompilez votre noyau et modifiez le fichier `/etc/lilo.conf`. Inspirez vous du mien, vous irez plus vite.

```
# Fichier de configuration de LILO
boot = /dev/hda3
delay = 30
prompt
vga = 792  # Nécessaire pour une réinitialisation dans un état normal
# Linux bootable partition config begins
image = /vmlinuz
  append = "video=matrox:vesa:440" # On bascule sur le pilote Matroxfb
  root = /dev/hda3
  label = Linux
  read-only # Non-UMSDOS filesystems should be mounted read-only for checking
```

Vous devrez ensuite créer les périphériques associés au gestionnaire de mémoire vidéo dans le répertoire /dev. Pour le premier, il vous suffit de taper :

```
# mknod /dev/fb0 c 29 0
```

Les suivants doivent être multiples de 32, soit, pour /dev/fb1 :

```
# mknod /dev/fb1 c 29 32
```

et ainsi de suite jusqu'au huitième si vous le souhaitez :

```
# mknod /dev/fb7 c 29 224
```

C'est tout ! Si l'un d'entre vous se sert simultanément de plusieurs cartes, qu'il me contacte aussi vite que possible afin que je documente davantage.

6.2 Utilisation des cartes graphiques Permedia.

Les cartes de type Permedia ne sont pas supportées par le pilote vesafb. Heureusement, il existe un gestionnaire de mémoire vidéo spécifique aux cartes Permedia. En supposant que vous employez menuconfig pour paramétrer le noyau avant une compilation, exécutez les instructions suivantes :

Allez dans le menu “Code Maturity Level” et activez l'option suivante :

- Prompt for development and/or incomplete code/drivers

[Ceci peut changer dans les futurs noyaux. Le HOWTO sera alors modifié]

Dans le menu “Console Drivers”, sélectionnez :

- VGA Text Console
- Video Selection Support
- Support for frame buffer devices (experimental)
- Permedia2 support (experimental)
- Generic Permedia2 PCI board support
- Advanced Low Level Drivers
- Choisissez les pilotes Mono, 2bpp, 4bpp, 8bpp, 16bpp, 24bpp et “32bpp packed pixel”.
- Si vous souhaitez incorporer les fontes, activez les options suivantes :
  - Compiled-in fonts
  - Sparc console 12x22 font

Recompilez votre noyau et modifiez le fichier `/etc/lilo.conf`. Inspirez vous du mien pour aller plus vite.

```
# Fichier de configuration de LILO
boot = /dev/hda3
delay = 30
prompt
vga = 792  # Nécessaire pour une réinitialisation dans un état normal
# Linux bootable partition config begins
image = /vmlinuz
  append = "video=pm2fb:mode:1024x768-75,font:SUN12x22,ypan" # then switch to pm2fb
  root = /dev/hda3
  label = Linux
  read-only # Non-UMSDOS filesystems should be mounted read-only for checking
```

La ligne “pm2fb:mode:1024x768-75,font:SUN12x22,ypan” indique que le pilote opérera dans une résolution de 1024 par 768 à 75Hz avec les fontes SUN 12 par 22 (si vous les avez incluses). Ypan autorise le défilement. Vous pouvez employer un autre mode.

Vous devrez ensuite créer les périphériques associés au gestionnaire de mémoire vidéo dans le répertoire /dev. Pour le premier, il vous suffit de taper

```
# mknod /dev/fb0 c 29 0
```

Les suivants doivent être multiples de 32, soit, pour /dev/fb1 :

```
# mknod /dev/fb1 c 29 32
```

et ainsi de suite jusqu'au huitième si vous le souhaitez :

```
# mknod /dev/fb7 c 29 224
```

Pour davantage de renseignements concernant les fonctionnalités du pilote Permedia, consultez >.

video=pm2fb:[option[,option[,option...]]]

où vous disposez des options suivantes :

- off pour désactiver le pilote.
- mode:resolution pour fixer la résolution. Les modes proviennent du fichier fb.modes.ATI contenu dans le paquetage logiciel pbset de Geert Uytterhoeven. Tous les modes sont en 8 bits par pixel. Voici ceux disponibles :
  - 640x480-(60,72,75,90,100)
  - 800x600-(56,60,70,72,75,90,100)
  - 1024x768-(60,70,72,75,90,100,illo) illo=80KHz 100Hz
  - 1152x864-(60,70,75,80)
  - 1280x1024-(60,70,74,75)
  - 1600x1200-(60,66,76)
- Par défaut, la console fonctionne en 640 par 480 à 60 Hz.
- font:fontname pour fixer la fonte. Par exemple : font:SUN12x22.
- ypan offre une taille virtuelle dans le sens vertical aussi importante que la mémoire vidéo l'autorise.
- oldmem ne servira qu'aux propriétaires d'une CybervisionPPC. Ajoutez cette option si votre carte est munie de SGRAM Fujitsu, ce qui est le cas des CyberVisionPPC antérieures au 30/12/1999.
- virtual (transitoire) est à employer si le noyau fixe lui-même les adresses d'accès sur les bus PCI.

6.3 Utilisation des cartes graphiques ATI

Remarque : les informations qui suivent ne viennent pas de moi vu que je ne dispose pas d'une carte ATI pour les vérifier. Si je me trompe, n'hésitez pas à me corriger, à m'insulter ou à m'envoyer votre carte ! 8-)

Les cartes ATI sont plus ou moins bien gérées par le pilote vesafb selon leur qualité intrinsèque. Heureusement, il existe un gestionnaire de mémoire vidéo spécifique aux cartes ATI. En supposant que vous employez menuconfig pour paramétrer le noyau avant une compilation, exécutez les instructions suivantes

Allez dans le menu “Code Maturity Level” et activez l'option suivante :

- Prompt for development and/or incomplete code/drivers

[ceci peut changer dans les futurs noyaux. Ce HOWTO sera alors modifié]

Dans le menu “Console Drivers”, sélectionnez :

- VGA Text Console
- Video Selection Support
- Support for frame buffer devices (experimental)
- ATI Mach64 display support
- Advanced Low Level Drivers

- Choisissez les pilotes Mono, 2bpp, 4bpp, 8bpp, 16bpp, 24bpp et “32bpp packed pixel”.
- Si vous souhaitez incorporer les fontes, activez les options suivantes :
  - Compiled-in fonts
  - Sparc console 12x22 font

Recompilez votre noyau et modifiez le fichier `/etc/lilo.conf`. Inspirez vous du mien, ce sera le plus rapide.

```
# Fichier de configuration de LILO
boot = /dev/hda3
delay = 30
prompt
vga = 792  # Nécessaire pour une réinitialisation dans un état normal
# Linux bootable partition config begins
image = /vmlinuz
  append = "video=atyfb:mode:1024x768,font:SUN12x22"
  root = /dev/hda3
  label = Linux
  read-only # Non-UMSDOS filesystems should be mounted read-only for checking
```

La ligne “atyfb:mode:1024x768,font:SUN12x22” indique que le pilote opérera dans une résolution de 1024 par 768.

Vous devrez ensuite créer les périphériques associés au gestionnaire de mémoire vidéo dans le répertoire /dev. Pour le premier, il vous suffit de taper :

```
# mknod /dev/fb0 c 29 0
```

Les suivants doivent être multiples de 32, soit, pour /dev/fb1 :

```
# mknod /dev/fb1 c 29 32
```

et ainsi de suite jusqu’au huitième si vous le souhaitez :

```
# mknod /dev/fb7 c 29 224
```

video=atyfb:[option[,option[,option...]]]

où vous disposez des options suivantes :

- font:STRING pour fixer la fonte. Par exemple : font:SUN12x22
- noblink désactive l’extinction de l’écran
- noaccel désactive les routines d’accélération
- vram:ULONG précise au pilote atyfb la quantité de mémoire vidéo disponible
- pll:ULONG ?
- mclk:ULONG ?
- vmode:ULONG ?
- cmode:ULONG - fixe le nombre de bits par pixel : 0, 8, 15, 16, 24 ou 32

6.4 Quelles cartes graphiques se conforment aux spécifications VESA 2.0 ?

Voici une liste de cartes qui fonctionnent avec vesafb:

- ATI PCI VideoExpression 2MB (au maximum 1280 par 1024 en 8bit)
- ATI PCI All-in-Wonder
- Matrox Millennium PCI - BIOS v3.0
- Matrox Millennium II PCI - BIOS v1.5
- Matrox Millennium II AGP - BIOS v1.4
- Matrox Millennium G200 AGP - BIOS v1.3
- Matrox Mystique & Mystique 220 PCI - BIOS v1.8
- Matrox Mystique G200 AGP - BIOS v1.3
- Matrox Productiva G100 AGP - BIOS v1.4
- Toutes les cartes à base de Riva 128
- Diamond Viper V330 PCI 4MB
- Genoa Phantom 3D/S3 ViRGE/DX
- Hercules Stingray 128/3D avec une sortie pour la télévision
- Hercules Stingray 128/3D sans sortie pour la télévision - une mise à jour du BIOS est nécessaire (contactez support@hercules.com)
- SiS 6326 PCI/AGP 4MB
- STB Lightspeed 128 (à base de Nvida Riva 128) PCI
- STB Velocity 128 (à base de Nvida Riva 128) PCI
- Jaton Video-58P ET6000 PCI 2MB-4MB (au maximum 1600 par 1200 en 8bit)
- Voodoo2 2000

Une liste de cartes mères incluant un jeu de composants graphiques :

- Trident Cyber9397
- SiS 5598

Les cartes qui ne fonctionnent pas :

- TBA

6.5 Vesafb est-il modularisable ?

A ma connaissance, Vesafb ne peut pas être modularisé. Les développeurs de vesafb s'y atteleront peut-être un jour. De toute façon, si le pilote est modularisé, vous ne disposerez d'aucun affichage à l'écran tant que le module vesafb n'aura pas été modprobé. Il vaut sûrement mieux le laisser dans le noyau, des fois que le démarrage se passe mal.

6.6 Comment puis-je modifier le curseur ?

[Tiré du fichier VGA-softcursor.txt - merci à Martin Mares!]

Linux offre une certaine latitude pour modifier l'allure du curseur. En principe, vous pouvez fixer la taille de celui-ci et, par la même occasion, contourner quelques problèmes matériels de cartes Trident défectueuses (cf. #define TRIDENT_GLITCH dans le fichier drivers/char/vga.c). Si vous activez l'option de génération logicielle du curseur ("Software generated cursor"), des nouveautés se présentent : un curseur rouge, un qui intervertisse la couleur de premier plan et celle du fond, une mise en relief du caractère actif qui laisse le curseur matériel visible ou non. Je n'ai sûrement pas pensé à tout.

On contrôle l'allure du curseur via la séquence d'échappement

```
<ESC>[?1;2;3c
```

ou 1, 2 et 3 sont des paramètres que l'on va décrire à présent. Les paramètres absents prennent la valeur 0.

Le premier paramètre correspond à la taille du curseur (0=défaut, 1=transparent, 2=souligné, ..., 8=caractère plein). Ajoutez 16 pour rendre actif le curseur logiciel, 32 si la couleur de fond doit être systématiquement changée, 64 pour que les couleurs de premier plan et de fond soient distinctes. La graisse est ignorée pour les deux derniers attributs.

Le second paramètre indique quels sont les bits d'attributs à changer (un simple ou exclusif). Sur un écran VGA standard, les quatre bits de poids fort précisent le fond et les quatre de poids faible le premier plan. Dans chaque quartet, les trois bits de poids faible donnent la couleur et celui de poids fort active la mise en relief (ou active le clignotement suivant la configuration de la carte VGA).

Le troisième paramètre correspond aux valeurs que doivent prendre les bits que l'on souhaite modifier. Le positionnement d'un bit a lieu avant son masquage; on force donc à 0 un bit en l'activant à la fois dans le masque de sélection et dans celui de positionnement.

Un curseur qui souligne et clignote : echo -e '\033[?2c' Un bloc qui clignote : echo -e '\033[?6c' Un bloc rouge qui ne clignote pas : echo -e '\033[?17;0;64c'

7 Le pilote de mémoire vidéo sur les Atari m68k

Cette partie décrit les options offertes par le pilote de mémoire vidéo sur les machines Atari m68k.

7.1 Quels sont les modes disponibles sur les machines Atari m68k ?

```
Couleurs   320x200  320x480  640x200  640x400  640x480  896x608  1280x960
---------+------------------------------------------------------------
 1 bit   |                            sthigh   vga2     falh2    tthigh
 2 bits  |                   stmid             vga4
 4 bits  | stlow                             ttmid/vga16 falh16
 8 bits  |          ttlow                      vga256
```

`ttlow, ttmid et tthigh` sont seulement employés sur les modèles TT tandis que `vga2, vga4, vga15, vga256, falh3 et falh16` ne servent que sur le Falcon. Lorsqu'une option `video=xxx` est donnée au noyau, en l'absence toute sous-option, le noyau teste les modes vidéo dans l'ordre suivant jusqu'à ce qu'il en trouve un d'adapté au matériel:

- `ttmid`

- `tthigh`
- `vga16`
- `sthigh`
- `stmid`

Vous pouvez préciser le mode à employer pour éviter l'auto-détection. Par exemple, `video=vga16` procure un écran en 640 par 480 avec une profondeur de 4 bits.

7.2 Sous options supplémentaires sur les machines Atari m68k

Options supplémentaires disponibles avec le paramètre `video=xxx` :

- `inverse` - inversion des couleur de fond et de premier plan. Normalement le fond est noir; cette option le rend blanc.
- `font` - fonte à employer en mode texte. Les fontes suivantes sont actuellement disponibles : `VGA8x8`, `VGA8x16`, `PEARL8x8`. La fonte `VGA8x8` est utilisée par défaut si la dimension verticale de l'écran est inférieure à 400 pixels sans quoi la fonte `VGA8x16` est employée.
- `internal` - très intéressant. Se reporter à la section suivante.
- `external` - idem.
- `monitorcap` - description des modes multisync disponibles. PROSCRIT pour les moniteurs à fréquence fixe.

7.3 Sous option "internal" sur les machines Atari m68k

Syntaxe : `internal:(xres);(yres)[;(xres_max);(yres_max);(offset)]`

L'option indique les fonctionnalités ajoutés par certains périphériques vidéo tels les modes d'OverScan. `(xres)` et `(yres)` fournissent les dimensions étendues de l'écran. Si vos modes d'OverScan nécessitent une bordure noire, vous devrez expliciter les trois derniers arguments de la sous-option `internal:`. `(xres_max)` correspond à la plus grande dimension de ligne acceptée par le matériel tandis que `(yres_max)` donne le nombre maximal de lignes et `(offset)` le décalage en octets entre la partie visible de la mémoire vidéo et son emplacement physique.

Les matériel vidéo étendu requiert souvent une activation qui fait appel aux options `"switches=*"`. [L'auteur apprécierait de recevoir des informations supplémentaires à ce sujet. La documentation m68k du noyau manque de clarté sur ce sujet et l'auteur ne possède pas d'Atari! Des exemples seront également les bienvenus.]

7.4 Sous option "external" sur les machines Atari m68k

Syntaxe : `external:(xres);(yres);(depth);(org);(scrmem)[;(scrlen)[;(vgabase)[;(colw)[;(coltype)[;(xres`

On rentre dans le compliqué. Le présent document essaye d'être aussi clair que possible mais l'auteur n'a rien contre une relecture afin d'être sûr qu'il n'a rien loos^H^Hupé.

Cette sous-option indique que vous disposez de périphériques vidéo externes (vraisemblablement une carte vidéo) et indique comment Linux doit l'employer. Normalement, le noyau se limite à ce qu'il peut apprendre des périphériques vidéo internes. Vous devez donc lui fournir tous les paramètres nécessaires afin qu'il soit en mesure de gérer des périphériques externes. Il y a deux limitations : vous basculerez dans le mode adéquat avant l'initialisation et une fois celle-ci effectuée, vous ne pourrez pas changer de mode.

Les trois premiers paramètres sont évidents. Ils correspondent aux dimensions de la zone d'affichage : hauteur et largeur en pixel suivies de la profondeur. Le paramètre de profondeur servant d'exposant au nombre 2 donne le nombre de couleurs. Par exemple, pour un affichage en 256 couleurs, vous préciserez un paramètre de 8. Le paramètre dépend de l'adaptateur graphique externe bien que vous soyez de toute façon limité par le matériel.

Vous devez ensuite décrire au noyau l'organisation de la mémoire vidéo via le paramètre `(org)`.

- `n` - plans disposés normalement, les uns à la suite des autres.
- `i` - plans entrelacés, c'est à dire 16 bits du premier plan, puis du suivant etc. Seuls les modes vidéo natifs d'Atari utilisent ça et aucune carte vidéo ne le gère.
- `p` - pixels regroupés. Les bits constitutifs des différents plans d'un même pixel se suivent. Ce mode est le plus courant en 256 couleurs.
- `t` - couleurs vraies. Il s'agit du mode précédent en l'absence de toute table de correspondance des couleurs. Ces modes sont généralement sur 24 bits et procurent quelques 16,8 millions de couleurs.

A coté de ça, le paramètre `(org)` a une signification bien différente pour les modes monochromes.

- `n` - couleurs usuelles, c'est à dire 0 pour le blanc et 1 pour le noir;
- `i` - couleurs inversées, c'est à dire 0 pour le noir et 1 pour le blanc.

L'élément suivant ayant trait au périphérique vidéo fixe l'adresse de base de la mémoire vidéo. Il est donné par le paramètre `(scrmem)` sous forme hexadécimal (préfixé par `0x`). Vous devriez trouver cette information dans la documentation fournie avec le périphérique.

Le paramètre suivant, `(scrlen)`, fournit au noyau la taille de la zone de mémoire vidéo. S'il est absent, il est calculé à partir des valeurs de `(xres)`, `(yres)` et `(depth)`. En bref, il ne sert à rien de préciser une valeur. Si vous donnez à sa suite le paramètre `(vgabase)`, laissez le champ vide en rentrant deux point-virgules. Autrement, oubliez le.

Le paramètre `(vgabase)` est optionnel. En son absence, le noyau ne pourra lire ni écrire le moindre des registres de couleur du périphérique et il vous faudra donc installer les couleurs appropriées avant le démarrage de Linux. Si la carte est compatible VGA, vous pouvez donner au noyau l'adresse où se trouvent les registres vidéo de façon à ce qu'il modifie lui-même les tables des couleurs. Vous trouverez cette information dans la documentation fournie avec le périphérique. Afin d'être *clair*, `(vgabase)` est une adresse de *base*, donc alignée sur un multiple de 4k. Pour l'accès en lecture ou en écriture aux registres, le noyau utilise une plage d'adresses comprises entre `(vgabase) + 0x3c7` et `(vgabase) + 0x3c9`. La valeur est donnée en hexadécimal et doit être préfixée par `0x` (tout comme `(scrmem)`).

`(colw)` ne sert que si `(vgabase)` est spécifié. Il donne au noyau la taille des registres de couleur, c'est à dire le nombre de bits par couleur (rouge/verte/bleue). La valeur par défaut est de 6 bits mais il est courant d'en spécifier 8.

`(coltype)` s'emploie en conjonction avec `(vgabase)`. Il précise aux noyau le type des registres de la carte graphique. Actuellement, deux modèles sont gérés : `vga` et `mv300`. Par défaut, `vga` est employé.

`(xres_virtual)` n'est nécessaire qu'avec les cartes ProMST/ET4000 pour lesquelles la longueur physique des lignes diffère de leur taille visible. Avec une ProMST, on donnera la valeur 2048 tandis que pour l'ET4000 cela dépendra de l'initialisation de la carte vidéo.

8 Le pilote de mémoire vidéo avec les Amiga

Cette partie décrit les options offertes sur les Amiga, options voisines de celles de l'Atari m68k.

8.1 Quels sont les modes disponibles sur les machines Amiga ?

Ça dépend du composant employé dans votre Amiga. Il y en a essentiellement trois : `OCS, ECS et AGA`. Tous on recours au pilote de mémoire vidéo.

- Modes NTSC
  - `ntsc` - 640x200
  - `ntsc-lace` - 640x400
- PAL modes
  - `pal` - 640x256
  - `pal-lace` - 640x512
- Modes ECS - 2 bits de couleur avec les composants ECS, 8 bits avec les composants AGA
  - `multiscan` - 640x480
  - `multiscan-lace` - 640x960
  - `euro36` - 640x200
  - `euro36-lace` - 640x400
  - `euro72` - 640x400
  - `euro72-lace` - 640x800
  - `super72` - 800x300
  - `super72-lace` - 800x600
  - `dblntsc` - 640x200
  - `dblpal` - 640x256
  - `dblntsc-ff` - 640x400
  - `dblntsc-lace` - 640x800
  - `dblpal-ff` - 640x512
  - `dblpal-lace` - 640x1024
- Modes VGA - 2 bits de couleur avec les composants ECS, 8 bits avec les composants AGA
  - `vga` - 640x480
  - `vga70` - 640x400

8.2 Sous options supplémentaires sur les machines Amiga m68k

Elles sont voisines de celles de l'Atari m68k :

- `depth` - précise le nombre de bits par pixel.
- `inverse` - même chose que sur les Atari.

- `font` - même chose que sur les Atari, mais la fonte `PEARL8x8` remplace la fonte `VGA8x8` si la largeur de la zone d'affichage est inférieure à 400 pixels.
- `monitorcap` - description des modes multisync disponibles. PROSCRIT pour les moniteurs à fréquence fixe.

8.3 Cartes d'extension graphiques gérées sur Amiga

- `Phase5 CyberVision 64` (composant S3 Trio64)
- `Phase5 CyverVision 64-3D` (composant S3 ViRGE)
- `MacroSystems RetinaZ3` (composant NCR 77C32BLT)
- `Helfrich Piccolo, SD64, GVP ECS Spectrum, Village Tronic Picasso` `IIII+` and IV/ (Cirrus Logic GD542x/543x)

9 Le pilote de mémoire vidéo sur les Macintosh m68k

La version courante du gestionnaire de mémoire vidéo ne gère que les modes choisis sous MacOS avant l'initialisation de Linux ainsi que les modes couleur en 1, 2, 4 et 8 bits.

Le pilote gère les options de la forme :

```
video=macfb:<font>:<inverse>
```

Les fontes VGA8x8, VGA8x16, 6x11, etc... sont disponibles. L'option inverse permet bien sûr d'inverser la vidéo.

10 Le pilote de mémoire vidéo sur les PowerPC

L'auteur aimerait recevoir des informations relatives au gestionnaire de mémoire vidéo sur ces machines.

11 Le pilote de mémoire vidéo sur les Alpha

11.1 Modes disponibles

Pour l'instant il n'y a que la carte PCI TGA. Elle offre un mode de 80 lignes par 30 colonnes en 640x480 avec une profondeur de 8, 24 ou 32 bits.

11.2 Cartes graphiques gérées par le pilote de mémoire vidéo

La carte graphique suivante a été testée avec succès :

- `DEC TGA PCI (DEC21030)` - 640x480 @ 8 bit or 24/32 bit versions

12 Le pilote de mémoire vidéo sur les SPARC

12.1 Cartes graphiques gérées par le pilote de mémoire vidéo

- MG1/MG2 - version SBus ou intégrée (Sun3) - au maximum 1600x1280 monochrome (BWtwo)
- CGthree - semblable aux MG1/MG2 mais offrant la couleur - résolution maximale ?
- GX - SBus - au maximum 1152x900 en 8bits (CGsix)
- TurboGX - SBus - au maximum 1152x900 en 8 bits (CGsix)
- SX - SS10/SS20 - au maximum 1280x1024 en 24 bits - (CGfourteen)
- ZX(TZX) - SBus - carte accélératrice 3D 24 bits - résolution maximale ? (Leo)
- TCX - AFX - Sparc 4 - au maximum 1280x1024 en 8 bits
- TCX(S24) - AFX - Sparc 5 - au maximum 1152x900 en 24 bits
- Creator - SBus - au maximum 1280x1024 en 24 bits (FFB)
- Creator3D - SBus - au maximum 1920x1200 en 24 bits (FFB)
- ATI Mach64 - carte accélératrice 8/24 bits pour Sparc64 sur bus PCI

Une option de la PROM permet l'envoi des caractères d'affichage à l'écran ou sur une console série. Jetez un oeil à la FAQ du Frame Buffer sur Sparc : >.

12.2 Configuration du gestionnaire de mémoire vidéo

Pendant la configuration du noyau (make config ou autre), il vous faut choisir entre `promcon` ou `fbcon`. La compilation des deux est possible mais il faudra spécifier au noyau le pilote à employer. Par défaut, `fbcon` est essayé en premier au démarrage. Si `promcon` n'a pas été sélectionné, `dummycon` est activé pendant l'initialisation. Une fois les bus initialisés, si `fbcon` est compilé, le noyau recherche les périphériques précédents et se sert de `fbcon`. En l'absence de gestionnaires de mémoire vidéo, le noyau a recours à `promcon`.

Voici les options du noyau :

```
video=sbus:options
        options inclue les éléments suivants, séparés par une virgule :
                nomargins       marge nulle;
                margins=12x24   marge de 12 par 24 (calculé par défaut en
                fonction de la résolution);
                off             inhibition de la détection des pilotes de
                mémoire vidéo SBus/UPA;
                font=SUN12x22   emploi d'une fonte particulière.
```

Au démarrage, un paramétrage de la forme

```
video=sbus:nomargins,font=SUN12x22
```

procure une agréable console en mode texte, rapide, avec une résolution de 96 par 40 qui ressemble à une console Solaris avec la couleur et les terminaux virtuels en plus comme sur les compatibles PC.

Pour que l'affichage se fasse avec la fonte `SUN12x22`, vous devez l'activer durant la configuration du noyau (désactivez l'option `fontwidth != 8`). Le pilote de mémoire vidéo accéléré gère n'importe quelle fonte dont la largeur est comprise entre 1 et 16 pixels tandis que le pilote de base ne gère que les fontes larges de 4, 8, 12 ou 16 pixels. Un paquetage récent des consoletools est recommandé.

13 Le pilote de mémoire vidéo sur les MIPS

Il n'est pas besoin de modifier quoi que ce soit avec ce type de machines. Tout est géré automatiquement. En particulier, les Indys sont câblés de façon à offrir une console 160x64. Une réécriture du code de gestion de la console pour les Indys étant en cours, on gardera un oeil sur cette section.

14 Le pilote de mémoire vidéo sur les ARM

14.1 Netwinders

Pour les Netwinders (qui reposent sur le processeur RISC ARM SA110 au charme si délicieusement british), il existe deux versions du gestionnaire de mémoire vidéo pour les Cyber2000 : un pour les noyaux 2.0.x, l'autre pour les 2.2.y. Tant l'activation que l'emploi du pilote sont assez naturels avec les deux branches du noyau. Néanmoins, en 2.0.x, la résolution et la profondeur sont codées en dur (beuh...). Heureusement, la version 2.2.x est plus souple, du moins le sera une fois les pilotes davantage stabilisés. Le mieux que vous puissiez faire afin que tout fonctionne reste encore de lire la documentation fournie avec la portion ARM des sources du noyau. Les Netwinders intègrent un composant VGA mais il ne s'est malheureusement jusqu'ici trouvé personne pour porter vgafb. [Je m'y attelerai si quelqu'un me fournissait un Netwinder avec lequel jouer]

14.2 Archimedes Acorn

Les Acorns offrent un pilote de mémoire vidéo depuis les temps anciens des noyaux 1.9.x. Cependant, le gestionnaire Acornfb des noyaux 2.2.x est complètement nouveau puisque l'interface d'accès à la mémoire vidéo a été modifiée au cours du développement des noyaux 2.1.x (qui devinrent naturellement les 2.2.x). Comme précédemment, il n'est guère difficile d'activer le pilote et de configurer la profondeur et la résolution.

14.3 Autres architectures à base d'ARM (SA 7110s et variantes)

A ma surprise, même le Psion 5 et le Geofox disposent d'un pilote de mémoire vidéo ! On m'a dit que le manchot passait d'ailleurs plutôt bien. S'il vous plaît, donnez moi un Psion 5 !

15 Gestion de la mémoire vidéo avec plusieurs écrans

Cette partie du document a été fournie gracieusement par Frederick A. Niles qui conserve tous ses droits sur les informations données.

15.1 Introduction

Les quelques pages qui suivent sont censées permettre une première prise en main des configurations à deux écrans sous Linux. Bien que le processus se déroule naturellement, les occasions de se tromper ne manquent pas.

Je me suis focalisé sur la mise en place d'un serveur X sur un second moniteur. L'intérêt en est que l'on croise de temps à autre des personnes se débarrassant de vieux moniteurs de 19 ou 20 pouces à fréquence fixe car ils ne peuvent plus s'en servir. On peut ainsi démarrer avec un petit moniteur multisync et disposer de X sur un moniteur de grandes dimensions.

Comme il s'agit d'un domaine en plein développement, l'information évolue rapidement. Le contenu de ce document pourrait très bien être dépassé, voire complètement faux, lorsque vous le lirez.

** ATTENTION ** Ce texte a été rédigé avant la sortie de la version 4.0 de la XFree86 qui devrait modifier pas mal de choses. Essayez d'obtenir une nouvelle version de ce document si elle existe.

15.2 Retour

Le retour de la part des utilisateurs sera plus que certainement le bienvenu. Sans vos remarques et vos questions, ce document n'existerait pas. N'hésitez donc pas à me contacter à l'adresse suivante : Frederick.A.Niles@gsfc.nasa.gov.

15.3 Contributions

Les personnes suivantes ont participé à l'élaboration de ce Mini-HOWTO :

- Petr Vandrovec `vandrove@vc.cvut.cz`
- Andreas Ehliar `ehliar@lysator.liu.se` (x2x)
- Marco Bizzarri `m.bizzarri@icube.it` (multiple X servers)

15.4 Avertissements

L'auteur de ce document dégage toute responsabilité quant à son contenu. Vous employez les notions, exemples et tout ce qui figure ici à vos risques et périls. S'agissant d'une nouvelle version de ce document, des informations erronées ou inadéquates peuvent très bien entraîner la dégradation de votre matériel. Faîtes-y attention et, bien que ce soit hautement improbable, je me décharge de toute responsabilité à cet égard.

15.5 Propriété du document

15.6 Matériel supporté

La plupart des cartes vidéos supposent qu'elles assument seules cette fonction au sein du système. Elles occupent donc en permanence l'espace d'adressage de l'adaptateur graphique primaire. Il existe quelques exceptions :

- les cartes Matrox : Matrox Millennium, Matrox Millennium II, Matrox Mystique, Matrox Mystique 220, Matrox Productiva G100, Matrox Mystique G200, Matrox Millennium G200, Matrox Marvel G200.

- MDA : il s'agit essentiellement des cartes graphiques monochromes Hercules. Évidemment, on ne dispose que du mode texte.

Remarque : seul le second adaptateur graphique doit figurer dans la liste précédente.

15.7 Logiciels commerciaux

Ce Mini-HOWTO traite avant tout de logiciel libre. Certains serveurs X commerciaux sont néanmoins capables de gérer plusieurs moniteurs tels le serveur Metro-X de Metro Link (www.metrolink.com) et Accelerated-X de Xi Graphics (www.xig.com).

15.8 Logiciels nécessaires

Les patches et programmes suivants sont nécessaires :

- "fbset", examinez : > (remarque : ce programme est fourni avec la RedHat 6.0)
- patches du noyau pour la configuration à deux écrans "fbaddon" des cartes Matrox. Examinez : >
- "con2fb", examinez : >
- serveur X11 XF86_FBDev employant le gestionnaire de mémoire vidéo. Disponible en standard avec la version 3.3.1 de la XFree86.

15.9 Mise en route

Commencez par patcher votre version du noyau avec le patche "fbaddon". Ensuite, vous configurerez le noyau et activerez la gestion de la mémoire vidéo. Si vous disposez de cartes Matrox, incluez le pilote d'accélération unifié Matrox. Excluez le gestionnaire de mémoire vidéo VESA. Activez bien sûr la gestion de plusieurs adaptateurs, recompilez le noyau et réinitialisez le système.

A présent, installez l'utilitaire "fbset" et lisez attentivement la documentation relative à son paramètrage. La mise en place d'un fichier "/etc/fb.modes" est vivement recommandé une fois que vous vous serez décidé sur une configuration. Le paquetage fbset comprend un script Perl de conversion du fichier XF86Config en paramètres pour fb.modes. Vous trouverez mon script en shell Bourne dans les annexes A et B.

Vous devez vous mettre au point sur l'emploi du pilote de mémoire vidéo avec un seul adaptateur et bien identifier tout ce qui n'a rien à voir avec la gestion de plusieurs. Vous vous épargnerez ainsi pas mal de noeuds au cerveau. Je me focalise surtout sur la mise en place de X au niveau du second moniteur vu que la plupart des autres opérations de configuration en forment un sous-ensemble.

15.9.1 Déplacement d'une console

Compilez le programme "con2fb". Lancé sans arguments, il fournit le message suivant : "usage: con2fb fbdev console". Une commande telle que "con2fb /dev/fb1 /dev/tty6" attacherait la console virtuelle numéro 6 au second gestionnaire de mémoire vidéo. Ctrl-Alt-F6 vous basculera dans cette console qui s'affichera sur le second moniteur.

15.9.2 "fbset" et le paramètrage du second moniteur

La mise en place des paramètres "fbset" doit se cantonner au moniteur avec lequel "fbset" est employé. Faîtes donc attention à bien employer l'option "-fb" avec le second moniteur. Plus précisément, si vous ne voulez rien faire d'autre qu'accorder la résolution verticale virtuelle avec la résolution verticale réelle : "fbset -fb /dev/fb1 -vyres 600" (par exemple). L'affichage en mode texte en est sérieusement ralenti mais sans cela X reste vraiment hideux.

15.9.3 X et le gestionnaire de mémoire vidéo

Le fichier framebuffer.txt explique bien mieux que je ne puis le faire mais voici les deux points essentiels :

- vérifiez que le lien "X" pointe bien vers "XF86_FBDev",
- ajoutez une section Monitor à votre fichier XF86Config pour le gestionnaire de mémoire vidéo.

Par exemple :

```
# Serveur X s'appuyant sur le gestionnaire de mémoire vidéo.

Section "Screen"
    Driver      "fbdev"
    Device      "Millennium"
    Monitor     "NEC MultiSync 5FGp"
    Subsection "Display"
        Depth       8
        Modes       "default"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "default"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       24
        Modes       "default"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       32
        Modes       "default"
        ViewPort    0 0
    EndSubsection
EndSection
```

Restreignez vous aux modes "default" car je ne pense pas qu'il y en ait d'autres qui fonctionnent avec le pilote de mémoire vidéo Matrox.

15.9.4 Exécution du serveur X sur le second moniteur

Positionnez la variable d'environnement FRAMEBUFFER sur le second périphérique de mémoire vidéo : "export FRAMEBUFFER=/dev/fb1" ou : "setenv FRAMEBUFFER /dev/fb1" X doit être lancé avec des paramètres lui spécifiant à la fois la profondeur souhaitée au niveau des couleurs et un numéro correspondant à la console virtuelle employée. Par exemple : "startx – :0 -bpp 16 vt06". Le serveur X en 16 bits par pixel d'identifiant ":0" est attaché à la console virtuelle numéro 6. Utilisez ":1" au lancement d'un autre serveur X en le liant à une console dépendant de l'autre gestionnaire de mémoire vidéo et vous disposerez de deux serveurs X fonctionnant simultanément.

15.10 Résumé

Les étapes de mise en place d'un serveur X sur un second moniteur peuvent être ainsi résumées :

- se procurer le patch du noyau, fbset et con2fb;
- appliquer le patch, configurer le noyau, recompiler et reinitialiser;
- ajouter une section XF86_FBDev au fichier XF86Config et fixer le lien de X.

A chaque redémarrage :

- créer une console : "con2fb /dev/fb1 /dev/tty6";
- paramétrer : "fbset -fb /dev/fb1 1280x1024";
- positionner la variable FRAMEBUFFER : "export FRAMEBUFFER=/dev/fb1";
- lancer X : "startx – -bpp 16 vt06".

Un alias de shell permet d'automatiser ces tâches. Un script ne conviendrait pas puisqu'on a besoin de déterminer le numéro de la console courante. Voici mon alias (en C-shell) :

```
alias startxfb = "
setenv FRAMEBUFFER /dev/fb\!*;  # l'argument passe a l'alias est recupere
con2fb $FRAMEBUFFER /dev/$tty;  # positionne le pilote sur la console courante
fbset -fb $FRAMEBUFFER 1280x1024@62;  # Cf /etc/fb.modes
startx -- :\!* -bpp 16 vt0`echo $tty | cut -dy f 2`' # execution de X
"
```

Ces lignes correspondent au contenu de mon .cshrc aux commentaires près mais ils aident, avec les sauts de ligne, à en faciliter la lecture. Je fournis le numéro du pilote de mémoire vidéo comme argument à l'alias.

Si quelqu'un me fournit un équivalent pour bash, je l'incluerai ici. La commande tty vous fournira le nom de la console courante.

15.11 Remarques et problèmes

- fbset et startx *DOIVENT* être invoqués depuis une même console qui sera contrôlée par le pilote de mémoire vidéo. L'automatisation au moyen de scripts en est diminuée d'autant.
- la version 4.0 de la XFree86 gérera correctement les adaptateurs multiples mais la 3.3.1 en est encore incapable. Vous pouvez cependant disposer de deux serveurs avec la 3.3.1 et passer de l'un à l'autre avec x2x.

- Le pilote de mémoire vidéo non-actif conserve la dernière image sans la mettre à jour.
- L'écran qui n'est pas sélectionné ne conserve pas toujours son état durant ses périodes d'inactivité (mais en général il le fait). Geert Uytterhoeven, qui assure l'évolution du pilote de mémoire vidéo, et Linus Torvalds sont en désaccord sur les changements pour la gestion des adaptateurs multiples liés aux consoles (i.e. fbaddon) et ceux ci pourraient très bien ne jamais se retrouver dans l'arborescence officielle du noyau (cette information a une forte composante de type bruit de couloir).
- Si vous exécutez X de n'importe où, votre machine peut se retrouver dans un état passablement dégradé qui mélange les évènements de la souris et ceux du clavier.
- Le fichier framebuffer.txt dans l'arborescence du noyau mentionne la possibilité de modifier les paramètres Modeline du XF86Config alors que X fonctionne. Le gestionnaire de mémoire vidéo Matrox semble obliger le serveur X à tous les ignorer. On ne dispose donc que d'un paramètrage, celui qui est employé lors du basculement depuis le mode texte.
- XF86_FBDev ne dispose pas d'accélération. Des patches pour les cartes Matrox existent : >

15.11.1 Fonctionnement avec xdm

Je n'ai pas encore trouvé comment passer au niveau 5 dans une configuration à deux adaptateurs avec un serveur sur le second moniteur ou sur les deux. Bien que l'ajout d'une ligne au fichier Xservers de xdm/gdm soit aisé, la contrainte de démarrer le serveur X depuis la console gérée par le pilote de mémoire vidéo interdit cette solution. Si quelqu'un a une idée, qu'il m'en fasse part afin que je puisse l'ajouter.

15.11.2 L'utilitaire x2x

x2x vous permet de passer d'un serveur X à l'autre lorsque vous atteignez le bord d'un écran. Aux dernières nouvelles, ce programme se trouvait à l'adresse suivante : >. La distribution Debian en propose un paquetage. Je n'ai pas eu l'occasion de l'essayer mais plusieurs utilisateurs ont fait part d'expériences réussies.

15.11.3 Autres commandes utiles

Il est bon de garder présente à l'esprit l'existence de certaines commandes quand on dispose de plusieurs adaptateurs (surtout quand on écrit des scripts). * "chvt" permet de passer d'une console virtuelle (VT) à une autre. * "openvt" exécute un programme dans une console différente. * "tty" renvoie le nom de la console courante.

15.11.4 Annexe A. Script cvtmode.m pour Octave

Notez le positionnement de bpp.

```
#!/usr/bin/octave -q
bpp = 16;
DCF = sscanf(argv(1,:), "%f");
HR  = sscanf(argv(2,:), "%f");
SH1 = sscanf(argv(3,:), "%f");
SH2 = sscanf(argv(4,:), "%f");
HFL = sscanf(argv(5,:), "%f");
VR  = sscanf(argv(6,:), "%f");
```

```
SV1 = sscanf(argv(7,:), "%f");
SV2 = sscanf(argv(8,:), "%f");
VFL = sscanf(argv(9,:), "%f");
pixclock = 1000000 / DCF;
left_margin = HFL - SH2;
right_margin = SH1 - HR;
hsync_len = SH2 - SH1;

# 3) vertical timings:
upper_margin = VFL - SV2;
lower_margin = SV1 - VR;
vsync_len = SV2 - SV1;

RR = DCF / (HFL * VFL) *1e6;
HSF = DCF / HFL * 1e3;

printf("mode \"%dx%d\"\n",HR,VR);
printf("   # D: %3.2f MHz, H: %3.2f kHz, V: %2.2f Hz\n", DCF, HSF, RR);
printf("   geometry %d %d %d %d %d\n", HR, VR, HR, VR, bpp);
printf("   timings %d %d %d %d %d %d %d\n", ...
                                 pixclock, left_margin, right_margin, ...
                                 upper_margin, lower_margin, ...
                                 hsync_len, vsync_len);
printf("endmode\n");
```

15.11.5 Annexe B. Script "cvtfile" en Shell Bourne

Le script Octave "cvtmode" est utilisé.

```
#!/bin/sh

# Shell script to convert XF86Config file to fb.modes file.
# Uses octave script cvtmode.m

if [ -z $1 ]; then
  FILE=/etc/X11/XF86Config
else
  FILE=$1
fi

i=1
LEN=`grep Modeline $FILE | wc -l`
while expr $i \< $LEN > /dev/null ;
do
  CURLINE=`grep Modeline $FILE | cut -d'"' -f 3-20 | head -$i | tail -1 `
  ./cvtmode.m $CURLINE
  echo " "
  i=`expr $i + 1`
done
```

16 Gestion des fontes

Afin de pouvoir modifier les fontes, vous devez installer kbd-0.99. Le logiciel est disponible via >.

Le télé-chargement et l'installation de kbd-0.99 réside en ce que vous pourrez charger les fontes internationales (dont l'Euro) dans votre console. Je trouve très chic *en français dans le texte* d'avoir trois symboles sur mon clavier : le dollar, la livre et l'Euro.

17 Commutation de mode

Pour changer de mode (640x480, 800x800, etc ...), vous avez besoin de fbset (fbset-19990118.tar.gz pour l'instant) : > Le logiciel est fourni avec une documentation complète sur son emploi.

17.1 X11 et vesafb ?

Si votre version de XFree86 est antérieure à la 3.3.3.1, il est urgent de procéder à une mise à jour. Cette version comprend le pilote FBDev X pour les gestionnaires de mémoire vidéo. Autrement, vous pouvez compiler votre propre pilote FBDev pour des versions de XFree telles la 3.3.2 ou la 3.3.3.

Allez sur > et télé-chargez les dernières sources du serveur X. [NdT : le recours à un miroir ou à > sera peut-être plus rapide]

- Décompactez les sources.
- Éditez le fichier xc/config/cf/xf86site.def, et décommentez le #define relatif à XF68FBDevServer.
- Décommentez *toutes* les références à FB_VISUAL_STATIC_DIRECTCOLOR. Elles ne servent plus à rien. Si vous partez des sources de XFree86 3.3.3.1, sautez cette étape (la référence a été supprimée).
- Éditez xc/programs/Xserver/hw/xfree86/os-support/linux/lnx_io.c et changez K_RAW en K_MEDIUMRAW.

Recompilez le pilote. Ne vous souciez pas des références ayant trait à m68k : les architectures Intel sont supportées. Recompilez le tout. Ça va prendre un moment compte tenu de la taille des sources.

Si vous manquez de temps, les sites suivants proposent des versions pré-compilées. Notez que ces sites n'ont rien d'officiel et que vous utiliserez leurs binaires à vos risques et périls.

Pour une version libc5 : >. Pour une version glibc2 : >, >.

On signale qu'X11 ne fonctionne pas avec certaines cartes graphiques lorsque le gestionnaire vesafb est actif. Si vous êtes dans ce cas, essayez le nouveau pilote XF86_FBdev pour X11.

Utilisé conjointement à vesafb, ce pilote peut permettre l'emploi de X11 à des résolutions autrement inaccessibles au pilote X11 usuel (cartes MGA G200 par exemple).

XF86_FBdev requiert la configuration suivante du XF86Config :

```
Section "Screen"
        Driver          "FBDev"
        Device          "Primary Card"
        Monitor         "Primary Monitor"
        SubSection      "Display"
                Modes           "default"
        EndSubSection
```

```
EndSection
```

Vous devrez également positionner XkbDisable dans la section Keyboard ou bien exécuter XF86_FBDev avec l'option '-kb' afin de gérer correctement votre clavier. Sans XkbDisable, il vous faudra inclure les lignes suivantes dans votre .Xmodmap pour préciser les effets des touches. Le même résultat s'obtient en éditant son xkb si on le désire. *XFree86 3.3.3.1 ne présente plus ce défaut. Il est donc vivement conseillé d'effectuer une mise à jour vers cette version qui de plus corrige d'autres bugs et inclut FBDev parmi les serveurs.*

```
! Keycode settings required
keycode 104 = KP_Enter
keycode 105 = Control_R
keycode 106 = KP_Divide
keycode 108 = Alt_R Meta_R
keycode 110 = Home
keycode 111 = Up
keycode 112 = Prior
keycode 113 = Left
keycode 114 = Right
keycode 115 = End
keycode 116 = Down
keycode 117 = Next
keycode 118 = Insert
keycode 119 = Delete
```

Certaines adaptations seront sûrement nécessaires (copier les codes du gestionnaire X11 utilisé et positionner le nom du pilote sur FBDev) mais c'est en substance ce qu'il vous faudra faire pour que le pilote vesafb de X11 fonctionne. Les problèmes liés à X11 devraient être résolus dans les prochaines versions en ce qui concerne les cartes vidéo supportées.

18 Conversion des Modelines en paramètres d'affichage du pilote de mémoire vidéo

Rien n'est plus simple si XFree86 (X11) est installé sur votre machine et que vous pouvez vous en servir normalement.

Le pilote de mémoire vidéo requiert les champs suivants :

- pixclock - horloge pixel en picosecondes
- left_margin - durée entre la synchro et la zone affichée
- right_margin - durée entre la zone affichée et la synchro
- upper_margin - durée entre la synchro et la zone affichée
- lower_margin - durée entre la zone affichée et la synchro
- hsync_len - longueur de la synchro horizontale
- vsync_len - longueur de la synchro verticale

Une ligne "Modeline: XFree86 comprend les champs suivants :

```
Modeline  "1280x1024" DCF HR SH1 SH2 HFL VR SV1 SV2 VFL
```

Quelques calculs sont nécessaires pour la conversion. A titre d'exemple voici la conversion de valeurs extraites de mon XF86Config.

```
Modeline  "1280x1024" 110.00 1280 1328 1512 1712 1024 1025 1028 1054
```

Tout d'abord le paramètre pixclock. XFree86 l'exprime en MHz et le pilote de mémoire vidéo en picosecondes (pourquoi? mystère). On divise donc un million par DCF soit : 1,000,000 / 110.0 = 9090.9091

Pour les durées horizontales :

- left_margin = HFL - SH2
- right_margin = SH1 - HR
- hsync_len = SH2 - SH1

Soit, dans notre exemple :

- left_margin = 1712 - 1512 = 200
- right_margin = 1328 - 1280 = 48
- hsync_len = 1512 - 1328 = 184

Enfin les durées verticales :

- upper_margin = VFL - SV2
- lower_margin = SV1 - VR
- vsync_len = SV2 - SV1

Soit :

- upper_margin = 1054 - 1028 = 26
- lower_margin = 1025 - 1024 = 1
- vsync_len = 1028 - 1025 = 3

Les valeurs obtenues sont passées au gestionnaire de mémoire vidéo. Dans le cas du pilote matroxfb :

```
video=matrox:xres:<>,yres:<>,depth:<>,left:<>,right:<>,hslen:<>,upper:<>,lower:<>,vslen:<>
```

J'ai donc inséré la ligne suivante dans mon /etc/lilo.conf :

```
append = "video=matrox:xres:1280,yres:1024,depth:32,left:200,right:48,hslen:184,upper:26,lower:0,vsl
```

Notez que le pixclock n'est pas employé ici. Il n'est nécessaire que si celui par défaut ne vous convient pas. Il se fixe de la même façon ainsi qu'il a été auparavant expliqué dans ce document.

19 Modifier le logo Linux

Le logo se modifie par l'intermédiaire du fichier linux_logo.h dans le répertoire include/linux. Il s'agit d'un fichier d'en-tête C peu évident à manipuler, cependant il existe un module Gimp spécifique: >. Il suffit de disposer d'une image 80x80 comprenant au plus 224 couleurs. On peut laisser le module créer les trois variantes (2,16,224) ou on les crée soi-même et on les travaille avec le module. Le module réclame un répertoire où stocker le fichier. On spécifie ($SRCDIR)/include/linux/linux_logo.h et une fois la retouche d'image terminée, il ne reste plus qu'à recompiler et réinstaller le noyau. Si le noyau gère le tampon de mémoire vidéo, le logo apparait au prochain redémarrage.

20 Davantage d'informations ?

Que ceux qui sont intéressés aillent faire un tour du coté de > pour des informations relatives à la programmation du pilote.

La traduction originale de ce document en français se trouve à l'adresse suivante : >.