

Sommaire

Carte graphique i810 et Linux.....	1
Présentation.....	1
Le problème.....	1
Exemples de problèmes.....	1
Première solution.....	1
Bien configurer son noyau.....	2
Utilisation du DRI.....	3
Configuration de X.....	4
Conseils généraux.....	6
Copyright.....	7

Carte graphique i810 et Linux.

Carte graphique i810 et Linux.
par chez free.fr Martial Daumas

Ce document décrit comment utiliser et configurer la carte graphique i810 efficacement sur les différentes versions de Linux.

Présentation.

Le problème

Supportée depuis quelques temps directement par le noyau de Linux, la carte graphique Intel i810 peut parfois poser des problèmes avec certaines distributions, avec certains noyaux ou drivers ou dans des configurations matérielles spécifiques. Cette carte existe en différentes versions et modèles, mais la démarche est normalement la même à chaque fois.

Exemples de problèmes

Ceci est juste pour information, toutefois voici certains cas où j'ai rencontré des désagréments plus ou moins sévères (PC: Celeron 500Mhz):

SuSE Linux 6.4 : Problèmes fréquents avec XF86Setup (plantage de la machine), problème de disparition du device `/dev/agpgart` (bus AGP) lors des recompilations du noyaux dû à un noyau (trop?) lourdement patché par SuSE. Leur outil de configuration SAX plantait aussi souvent. Leur driver GLX semble défectueux.

SuSE Linux 7.0 : Problème encore et toujours avec la 3D, ainsi que des plantages de la machines avec SAX2 et XFree 4.

Mandrake7.1 : Supportait bien cette carte, mais les outils de configurations 'maison' avaient tendance à brider sans raison les possibilités de la carte (profondeur de couleur, résolution etc...)

Première solution

Après une compilation d'un noyau 2.2.16+ (ou avant avec des patch il me semble), il se peut que le système refuse de passer en mode graphique, avec des messages du style :

```
/dev/agpgart(i810) not found ....
```

En fait, cela ne m'est arrivé que sur des SuSE... Toujours est il qu'il faut faire un petit tour (toujours recommandable) par `/usr/src/linux/Documentation/devices.txt`, on y trouve ceci :

```
10 char Non-serial mice, misc features
...
...
    175 = /dev/agpgart    AGP Graphics Address Remapping Table
```

Ce qui signifie en clair que le 'device' agpart (bus AGP) a un Major de 10, type caractère avec un Minor de 175. le premier définit un type et une famille, et le deuxième est un identifiant spécifique au matériel.

Hardware-hard_image-i810

Nous allons donc créer ce 'device' par ces commandes:

```
(martial@lfs:martial)$ su
Password:*****
(root@lfs:martial)# cd /dev
(root@lfs:dev)# mknode /dev/agpgart c 10 175
```

Le 'c' précise qu'il s'agit d'un character device (et non pas block); on vérifie la création par:

```
(root@lfs:dev)# ls -l agpgart
crw----- 1 root root 10, 175 Aug 6 2000 agpgart
```

En cas de besoin, modifiez les permissions, mais normalement ça passe très bien ainsi. Nous venons de 'créer' la prise en charge par Linux du bus AGP (rappelez vous, tout est fichier), mais cela ne suffit pas si nous n'avons pas un 'device' spécifique pour notre carte i810, Ce device n'était malheureusement pas documenté dans le noyau 2.2.x (toujours pas en fait), mais une brève recherche sur internet nous apprend ce que l'on veut :

```
crw----- 1 root root 10, 240 Aug 8 2000 agpgarti810
```

Il faut donc refaire la manip précédente:

```
(root@lfs:dev)# mknode /dev/agpgarti810 c 10 240
(root@lfs:dev)# ls -l agpgarti810
crw----- 1 root root 10, 240 Aug 8 2000 agpgarti810
```

Là, on est bon !

Bien configurer son noyau.

Les manips précédentes sont utiles si vous aviez perdu les 'devices' corrects, mais ils ne seront utiles que dans le cas où votre noyau a bien le support du bus AGP, sinon ces devices sont aussi inutiles que /dev/mouse quand on n'a aucun driver souris (en statique ou module).

Pour que l'AGP avec la i810 marche, il nous faut sélectionner quelques options lors de la configuration d'un noyau, ce sont les mêmes avec les noyaux 2.2.x et 2.4.x.

```
(root@lfs:root)# cd /usr/src/linux&& make menuconfig
```

ou

```
(root@lfs:root)# cd /usr/src/linux&& make xconfig
```

Tout d'abord vérifiez que vous avez ceci :

```
Code maturity level options --->
[*] Prompt for development and/or incomplete code/drivers
(pour éviter d'avoir des options inaccessibles sur un noyau un peu ancien)

Processor type and features --->
(Pentium-Pro/Celeron/Pentium-II) Processor family
< > Toshiba Laptop support
< > /dev/cpu/microcode - Intel IA32 CPU microcode support
< > /dev/cpu/*/msr - Model-specific register support
< > /dev/cpu/*/cpuid - CPU information support
```

Hardware-hard_image-i810

```
(off) High Memory Support
[ ] Math emulation
[*] MTRR (Memory Type Range Register) support
[ ] Symmetric multi-processing support
[ ] APIC and IO-APIC support on uniprocessors
```

Mettez le processeur qui vous correspond au lieu du i386 par défaut, ça ira (un peu) plus vite. Les cartes mères i810 ont le mtrr normalement, sinon ça ne pose pas de problèmes, alors autant le mettre.

```
Character devices --->
<M> /dev/agpgart (AGP Support)
[ ] Intel 440LX/BX/GX and I815/I840/I850 support
[*] Intel I810/I815 (on-board) support
[ ] VIA chipset support
[ ] AMD Irongate support
[ ] Generic SiS support
[ ] ALI chipset support
```

L'option Intel n'est pas disponible en module, il faudra donc générer un nouveau noyau, et pas seulement compiler les modules. Nous passons donc à la compilation par :

```
(root@lfs:martial)# cp -R /lib/modules/2.4.5 /lib/modules/2.4.5.bak
(root@lfs:martial)# make bzImage
(root@lfs:martial)# make modules
(root@lfs:martial)# rm -R /lib/modules/2.4.5
(root@lfs:martial)# make modules_install
(root@lfs:martial)# mv /boot/vmlinuz /boot/vmlinuz.old
(root@lfs:martial)# cp arch/i386/boot/bzImage /boot/vmlinuz
```

En oubliant pas de mettre à jour LILO avec ce nouveau noyau :

```
(root@lfs:martial)# /sbin/lilo -v
```

Puis de mettre à jour les dépendances des modules du noyau par un :

```
(root@lfs:martial)# depmod -a
```

Utilisation du DRI.

Le DRI est une fonctionnalité de Xfree 4.x qui permet d'utiliser l'accélération matérielle de certaines cartes, dont la i810 sous X. La i810 n'est pas une carte 3D géniale, raison de plus de la booster un peu.

Pour information, je viens de réussir à utiliser cette fonction pour la première fois en utilisant une version de Linux faite 'maison' (cf. LFS). J'avais essayé avec des SuSE, Mandrake, RedHat etc... et toutes avaient plus ou moins ce qu'il fallait en théorie pour que ça marche, mais je n'ai jamais réussi. C'est sans doute possible... en tout cas si certains de vos RPMS semblent poser trop de problèmes, penser à essayer de les compiler vous-même.

(NDJC : si vous avez réussi à installer sur une de ces distributions, merci de nous transmettre votre méthode :-)

En tout cas, voici ce qu'il vous faut en plus dans le noyau :

```
[*] Direct Rendering Manager (XFree86 DRI support)
    < > 3dfx Banshee/Voodoo3+
    < > 3dlabs GMX 2000
```

```

< >   ATI Rage 128
< >   ATI Radeon
<*>   Intel I810
< >   Matrox g200/g400

```

C'est dans les noyaux 2.4.x, mais certaines distributions proposent des noyaux 2.2.x améliorés : à consommer avec la plus grande des modérations (NDLR : malheureusement on n'a pas le choix).

Configuration de X

Ensuite, il faut configurer votre `/etc/X11/XF86Config`. Plutôt qu'un long discours, je sort le mien, mais à titre d'exemple uniquement, ça ne passera sans doute pas tel quel chez vous :

```

##### /etc/X11/XF86Config #####
Section "Files"
    FontPath    "/usr/X11R6/lib/X11/fonts/misc:unscaled"
    FontPath    "/usr/X11R6/lib/X11/fonts/local"
    FontPath    "/usr/X11R6/lib/X11/fonts/misc:unscaled"
    .... et bla et bla ....

    ModulePath  "/usr/X11R6/lib/modules"
    # ca c important, ca doit pointer
    # sur l'endroit (variable) ou vous avez les modules
    #(dri, glx, freetype etc..) de X.

    RgbPath    "/usr/X11R6/lib/X11/rgb" #idem
EndSection

Section "ServerFlags"
    Option     "AllowMouseOpenFail"
EndSection

Section "Module"
    Load      "dbe" # le double buffer extension, parait que ca sert....
    Load      "type1"
    Load      "speedo"
    Load      "extmod"
    Load      "freetype" # si vous voulez les police .TTF
    Load      "glx" # important pour l'accel 3D
    Load      "dri" # imprtant aussi
EndSection

Section "InputDevice"
    Driver     "keyboard"

    .... ! depend de votre matériel ! .....

    UseModes   "Modes[0]"
EndSection

Section "Modes"
    Identifier  "Modes[0]"
    Modeline   "1024x768" 96.16 1024 1064 1200 1328 768 783 794 813
EndSection

Section "Screen"
    DefaultDepth 16
    SubSection "Display"
        Depth 16
        Modes "1024x768"

```

Hardware-hard_image-i810

```
EndSubSection
SubSection "Display"
    Depth 24
    Modes "1024x768"
EndSubSection
SubSection "Display"
    Depth 8
    Modes "1280x1024"
EndSubSection
Device "Device[0]"
Identifier "Screen[0]"
Monitor "Monitor[0]"
EndSection

Section "Device"
    BoardName "i810" # cette section est assez importante
    BusID "0:1:0" # ceci dépend du matériel :
    # X -scanpci
    # en mode console pour trouver le bon ID
    Driver "i810"
    VideoRam 8192 # la i810 se sert de la RAM générale, vous pouvez
    # en allouez plus si vous voulez
    Identifier "Device[0]"
    VendorName "Intel"
EndSection

Section "ServerLayout"
    Identifier "Layout[all]"
    InputDevice "Keyboard[0]" "CoreKeyboard"
    InputDevice "Mouse[1]" "CorePointer"
    Option "Xinerama" "off"
    Screen "Screen[0]"
EndSection

Section "DRI"
    Group "video" # pensez à créer le groupe video en tant que root
    Mode 0666 # normalemnt ca soit etre 0660, mais heuuuuhh.....
EndSection

#####fin de /etc/X11/XF86Config #####
```

N'oubliez pas que le DRI est supposé tourner en 16bits, ca passe aussi en 24bits, mais beaucoup moins bien.

Pour essayer le résultat, tester un des Xscreensaver (comme atlantis) ou un jeu en 3D (comme Xracer), si tout c'est bien passé ca doit aller très vite (au moins un quadruplement de vitesse chez moi).

Pensez aussi s'il n'existe pas à créer le device dri dans /dev :

```
(root@lfs:dev)# ls -l dri
total 0
crw-rw-rw-  1 root  root  10,  63 Jul 19 21:27 card0
```

donc un :

```
bash-2.05# mknod /dev/dri c 10 63
```

fera l'affaire.

Conseils généraux.

Dans tout les cas, faites le maximum pour opérer sur un système sain, c'est-à-dire installez les bons paquets, au besoin obtenez les mises à jours et/ou patches sur le site de votre distribution.

Utilisez une version stable de gcc, la version 3 est buggée (celle installée par RH7.1 ; NDLR : ce n'est pas la version 3 officielle..., mais une version 3 spéciale RedHat) (j'ai utilisé gcc version 2.95.2.1, impeccable).

Lisez bien la doc de Xfree et du noyau, il y a pleins de bons conseils importants dedans. Veillez à bien avoir les libs requises comme MESA, libGL, libGLUT, glx et c. Cela dépend malheureusement de votre distribution et peut s'avérer extrêmement compliqué. En installant tout à partir des sources, je n'ai eu aucun problème particulier.

Il vous faut dans tout les cas un serveur X prenant en charge la carte i810, pas de soucis pour les noyaux récents. Au besoin, mettez à jour, ou utilisez certains patches disponibles pour les vieux noyaux.

Soyez prudents lorsque vous touchez au noyau et/ou à X, faites des copies de sauvegardes, et gardez toujours un moyen de booter sur l'ancienne configuration.

...Et lisez le LEA-BOOK !

Cette page est issue de la documentation 'pré-wiki' de Léa a été convertie avec HTML::WikiConverter. Elle fut créée par Martial Daumas le 20/07/2001.

Copyright

Copyright © 20/07/2001, Martial Daumas



*Ce document est publié sous licence Creative Commons
Attribution, Partage à l'identique, Contexte non commercial 2.0 :*
<http://creativecommons.org/licenses/by-nc-sa/2.0/fr/>