

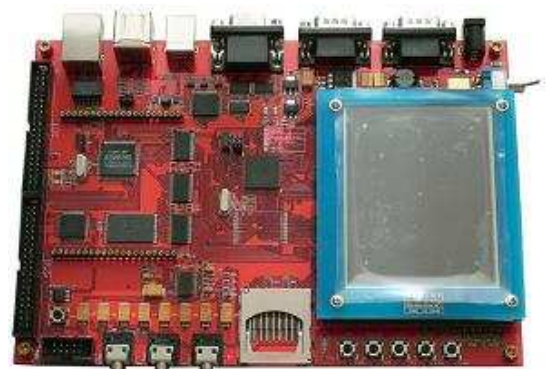
Linux2.6 - Mise en oeuvre

Pragmatec

Produits et services dédiés aux systèmes embarqués

Tutorial Linux2.6

ARM9 Development Starter Kit



Linux2.6 - Mise en oeuvre

Linux2.6 - Mise en oeuvre

Le kit de développement ARM9 est un kit réalisé par la société PRAGMATEC S.A.R.L., société située à Grenoble (www.pragmatec.net). Il est basé sur une carte de développement ARM9, largement utilisée en Asie. Il s'agit donc d'un produit efficace, fiable et disponible.

Pragmatec s'est attaché à faire de ce kit un environnement de développement complet et immédiatement opérationnel, avec une introduction en français et le reste des documents et exemples en langue anglaise. En cas de difficultés techniques vous bénéficiez de plus d'un support technique de la part de l'équipe support de Pragmatec sur notre forum www.pragmatux.net.

Ce document a pour but de démontrer la simplicité d'utilisation d'une telle plate-forme avec le système d'exploitation Linux 2.6. L'avantage du noyau 2.6 et qu'il vous permet de bénéficier d'un véritable PC embarqué similaire à votre station de développement. Vous pouvez donc recompiler vos applications pour la cible sans avoir recours à un portage fastidieux.

Nous n'aborderons pas ici les détails internes du noyau (modifications, drivers, gestion mémoire, ...) ceci étant destiné à un document plus spécifique. Toutefois vous découvrirez comment rapidement et simplement façonner votre carte en recompilant le noyau, le cramfs, la busybox et bien d'autres choses encore...

Ce document est la propriété de la société PRAGMATEC S.A.R.L. Il ne peut être reproduit et distribué sans l'accord de cette société.

TABLE DES MATIERES

1	<i>Préambule</i>	5
	La carte de développement	5
	Présentation de la carte ARM9	6
	Développement croisé sous Linux.....	7
2	<i>L'environnement de développement</i>	8
	Installation des sources de Linux	8
	Installation de la chaîne de compilation	9
	Paramétrage de minicom	10
3	<i>Création des images et des partitions</i>	11
	Compilation de Linux	11
	Génération des partitions NAND	13
	Busybox 1.2.1.....	14
	Microwindows	17
4	<i>Programmation de la cible</i>	19
	Préalable.....	19
	Programmation du noyau	20
	Cramfs	21
	Répertoires etc, usr et home	22
5	<i>Installation d'Eclipse</i>	24
	Pré-requis.....	24
	Debug distant.....	27
6	<i>Informations complémentaires</i>	32
	Connexion par telnet.....	32
	Transfert FTP	33
	Partage réseau par NFS	34
	Afficher une image sur le LCD.....	35
	Nano-X sans LCD.....	36
	Utilisation de l'écran tactile	37
	Interfaçage de composant I2C.....	37
	Montage d'une clef USB.....	38
	Montage d'une SDcard.....	39
	Montage d'un disque IDE.....	40
	Mise à jour du noyau depuis Linux	41
	Mise à l'heure de la carte	41

Linux2.6 - Mise en oeuvre

1 Préambule

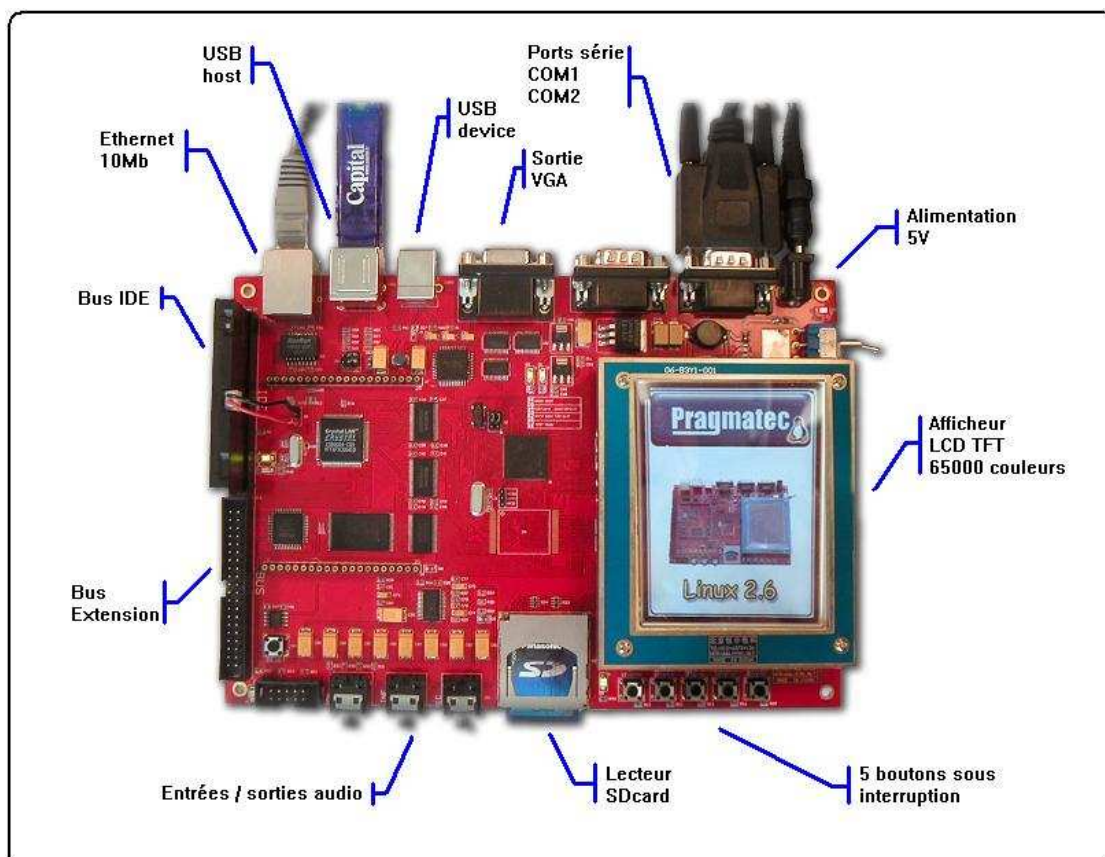


Le but de ce tutorial est de démontrer combien il est simple d'utiliser et de développer sur la plateforme ARM9/Linux2.6 sans pour autant devoir prétendre à une grande expérience dans le monde Linux et celui de l'embarqué. Vous y apprendrez comment installer votre environnement de développement, comment utiliser votre carte ARM9 avec Linux, les commandes de base et surtout comment réaliser votre propre application et la débuser sur la cible.

La carte de développement

La carte ARM9 qui vous est proposée est basée sur le processeur S3C2410 de Samsung. Elle possède de nombreux périphériques (USB, Ethernet, RS232, SDCard, ...) et vous permet de maquetter rapidement une application. Ce tutorial n'a pas pour prétention d'être exhaustif en vous présentant pas à pas la totalité des périphériques. Toutefois il sera enrichi dans le temps afin d'offrir un maximum d'informations liées à ces périphériques.

Le principe de la carte est d'offrir les périphériques nécessaires au développement (ports COM, Ethernet, flash, ...) mais aussi un certain nombre de périphériques qui vous permettront de réaliser des systèmes plus complexes (Sdcard, IDE, USB host, VGA, ...). Enfin la carte possède des connecteurs d'extension qui facilitent les développements de cartes filles.



Linux2.6 - Mise en oeuvre

Présentation de la carte ARM9

Les caractéristiques de la carte sont les suivantes :

- ✓ CPU S3C2410 à 200MHz
- ✓ 64 Mo de SDRAM
- ✓ 64 Mo de flash NAND
- ✓ Contrôleur Ethernet CS8900 10Mb
- ✓ 2 ports COM série
- ✓ sortie VGA
- ✓ USB device
- ✓ USB host
- ✓ Bus IDE
- ✓ Bus d'extension
- ✓ 5 boutons sur interruption
- ✓ LCD TFT 65000 couleurs (en option)
- ✓ Watchdog
- ✓ RTC sauvegardée par pile (fournie)

Les caractéristiques logicielles du kit sont les suivantes :

- ✓ BIOS hfrk
- ✓ Linux 2.6.14
- ✓ 6 partitions en flash NAND :
 - boot
 - kernel
 - rootfs
 - etc
 - usr
 - home
- ✓ Busybox 1.2.1
- ✓ Chaîne de compilation GCC 3.4.1
- ✓ Telnet, FTP, TFTP, NFS
- ✓ Environnement de développement Eclipse
- ✓ Debug distant d'applications via Ethernet
- ✓ Microwindows Nano-X (gestion graphique du LCD)
- ✓ Déport du LCD sur une station distante
- ✓ Gestion des polices TRUETYPE

Linux2.6 - Mise en oeuvre

[Développement croisé sous Linux](#)

Dans ce tutorial nous allons considérer un développement croisé depuis une plate-forme Linux. Cela signifie que nous développerons sur un PC Linux et que nous compilerons le noyau et les programmes utilisateurs pour notre cible S3C2410.

N'importe quel PC sous Linux fera l'affaire, mais nous avons choisi de réaliser ce tutorial depuis la FedoraCore4 ou FedoraCore5, téléchargeable depuis le site de RedHat. Nous vous recommandons vivement d'utiliser cette distribution car l'environnement de développement Eclipse utilisé pour le debug y est déjà présent.



Un développement croisé depuis Linux est préférable par rapport à un développement depuis Windows/VMWare. En effet, au fur et à mesure vous rajouterez des programmes et des services à votre carte ARM9 qui sont disponibles à l'identique sur votre PC Linux. N'hésitez pas à installer la FedoraCore sur votre PC déjà équipé de Windows sur une partition libre : un logiciel d'amorce sera installé afin de vous proposer au boot un choix quant au démarrage sous Windows ou Linux (logiciel GRUB).

Vous pouvez aussi télécharger Eclipse v3.1 directement du site officiel et procéder à son installation. N'oubliez pas d'installer aussi une machine virtuelle Java si vous n'en possédez pas une sur votre station (disponible sur le site de SUN). Pour la suite des opérations, vérifiez la présence de l'interpréteur de script TK « wish » (mettez à jour vos paquetages dans le cas contraire).

Le noyau linux compilé pour votre cible est de la génération v2.6.14.

La chaîne de compilation GCC utilisée comme cross-compiler est la v3.4.1.

Enfin nous rappelons que le noyau destiné à la cible est un noyau Linux 2.6. Les fichiers fournis sur le CDROM contiennent soit les sources des projets (noyau Linux, busybox) soit directement les binaires déjà compilés pour la cible (bibliothèques, debugger, ...).

Linux2.6 - Mise en oeuvre

2 L'environnement de développement



Nous abordons ici l'installation des outils et des sources du noyau Linux qui sera chargé sur la cible.

Installation des sources de Linux

Nous créons un répertoire de travail sous notre homedirectory : « /home/xavier/PRG_2410 » par exemple. Commencez par recopier les fichiers tar.gz du CDROM dans ce répertoire, puis décompressez les dans le répertoire PRG_2410 :

```
xavier@localhost:~/PRG_2410 - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
[xavier@localhost PRG_2410]$ ll
total 162280
-r--r--r-- 1 xavier xavier 15957983 Oct 19 11:20 images.tar.gz
-r--r--r-- 1 xavier xavier 51591279 Oct 19 11:20 linux-2.6.14.tar.gz
-r--r--r-- 1 xavier xavier 33282202 Oct 19 11:20 partitions.tar.gz
-r--r--r-- 1 xavier xavier 46394356 Oct 19 11:20 tools.tar.gz
-r--r--r-- 1 xavier xavier 18727900 Oct 19 11:20 user.tar.gz
[xavier@localhost PRG_2410]$ tar -xvzf images.tar.gz
[xavier@localhost PRG_2410]$ tar -xvzf linux-2.6.14.tar.gz
[xavier@localhost PRG_2410]$ tar -xvzf partitions.tar.gz
[xavier@localhost PRG_2410]$ tar -xvzf tools.tar.gz
[xavier@localhost PRG_2410]$ tar -xvzf user.tar.gz
[xavier@localhost PRG_2410]$
```

Ensuite créez un répertoire TGZ pour y placer les fichiers tar.gz (que vous conserverez ainsi en cas de ré-installation).

Linux2.6 - Mise en oeuvre

Parmi les différents répertoires créés, vous trouverez :

- ✓ **images** : contient l'ensemble des fichiers résultant de la compilation du noyau Linux (zImage) ainsi que le cramfs.
- ✓ **Linux-2.6.14** : répertoire dans lequel se trouvent les sources du noyau Linux (noyau lui-même, drivers, ...)
- ✓ **partitions** : contient les différentes partitions de la flash NAND vues par le BIOS :
 - ✓ **cramfs bb 1.2.1** : cramfs contenant les répertoires et fichiers principaux du système permettant ainsi au noyau de booter correctement.
 - ✓ **etc** : contenu du répertoire /etc de la cible. Contient de nombreux fichiers de paramétrage.
 - ✓ **usr** : contenu du répertoire /usr de la cible. Contient des programmes annexes nécessaires au fonctionnement complet de la cible.
 - ✓ **home** : contenu du répertoire /home de la cible. Correspond à la zone mémoire la plus importante sur la mémoire flash NAND. Contient les applications utilisateurs.
- ✓ **tools** : archivage des différents outils nécessaires à la reconstruction des logiciels de la cible. Contient également le serveur GDB pour cible ARM9 (debug de code sur la cible).
- ✓ **user** : répertoire destiné à contenir différentes applications phares pour la cible :
 - ✓ **busybox 1.2.1** : sorte de véritable couteau suisse regroupant la plupart des commandes classiques qu'on peut trouver sur une station Linux 2.6.
 - ✓ **microwindows-0.90-nxd** : client/serveur graphique utilisant le frame buffer de la cible afin d'afficher aisément du texte, des formes ou des images sur le LCD de la cible. Il s'agit de la version 0.90 patchée par Pragmatec afin de fonctionner en réseau ce qui vous évites de connecter un LCD TFT (la cible envoie les informations sur un PC Linux via Ethernet).
 - ✓ **freetype1** : contient la librairie freetype déjà compilée pour ARM9/Linux 2.6 afin de permettre à Microwindows de gérer les fonts TrueType telle que celles qui se trouve sur votre Pc sous Windows™ ou Linux.

Installation de la chaîne de compilation

Passez temporairement ROOT pour installer le package des outils de développement pour S3C2410 comme suit : « **tar jxvf arm-linux-gcc-3.4.1.tar.bz2** » .

Recopier ensuite le répertoire ainsi créé sous « **/usr/local/arm/3.4.1** » .

Vérifier votre installation : le compilateur « arm-linux-gcc » doit se trouver à cet endroit ci :
« **/usr/local/arm/3.4.1/bin/arm-linux-gcc** »

Attention : n'oubliez pas de passer « root » temporairement pour procéder à cette installation ! Ensuite quitter le compte root pour la suite des opérations.

Linux2.6 - Mise en oeuvre

Paramétrage de minicom

Comme équivalent à HyperTerminal sous Windows nous utilisons minicom. Si vous ne bénéficiez pas d'un port série sur votre ordinateur portable par exemple, utilisez une interface USB/RS232 très bien gérée de nos jours par la plupart des distributions Linux du commerce.

Pour paramétrer minicom, vous avez besoin de passe « root » :

```
A - Port série : /dev/ttyS0
B - Emplacement du fichier de verrouillage : /var/lock
C - Programme d'appel intérieur :
D - Programme d'appel extérieur :
E - D0bit/Parité/Bits : 115200 8N1
F - Contrôle de flux matériel : Non
G - Contrôle de flux logiciel : Non

Changer quel réglage ? █
```

Pour accéder aux diverses commandes, tapez simultanément sur les touches « CONTROL » et le caractère de votre commande (Faire CTRL+A, puis O, puis à l'aide de la flèche du pavé numérique, sélectionnez « configuration du port série » et tapez ENTER).

Vous pouvez aussi utiliser la commande « minicom -s » pour directement accéder aux paramètres.

Pour vérifier que votre paramétrage est correct, vous pouvez mettre sous tension votre carte une fois minicom lancé et paramétré : vous devriez arriver au prompt de Linux !

Surtout n'oubliez pas de dévalider votre pare-feu (firewall) ou au moins d'autoriser les transferts depuis l'adresse IP de la carte (192.168.0.30).

Linux2.6 - Mise en oeuvre

3 Création des images et des partitions

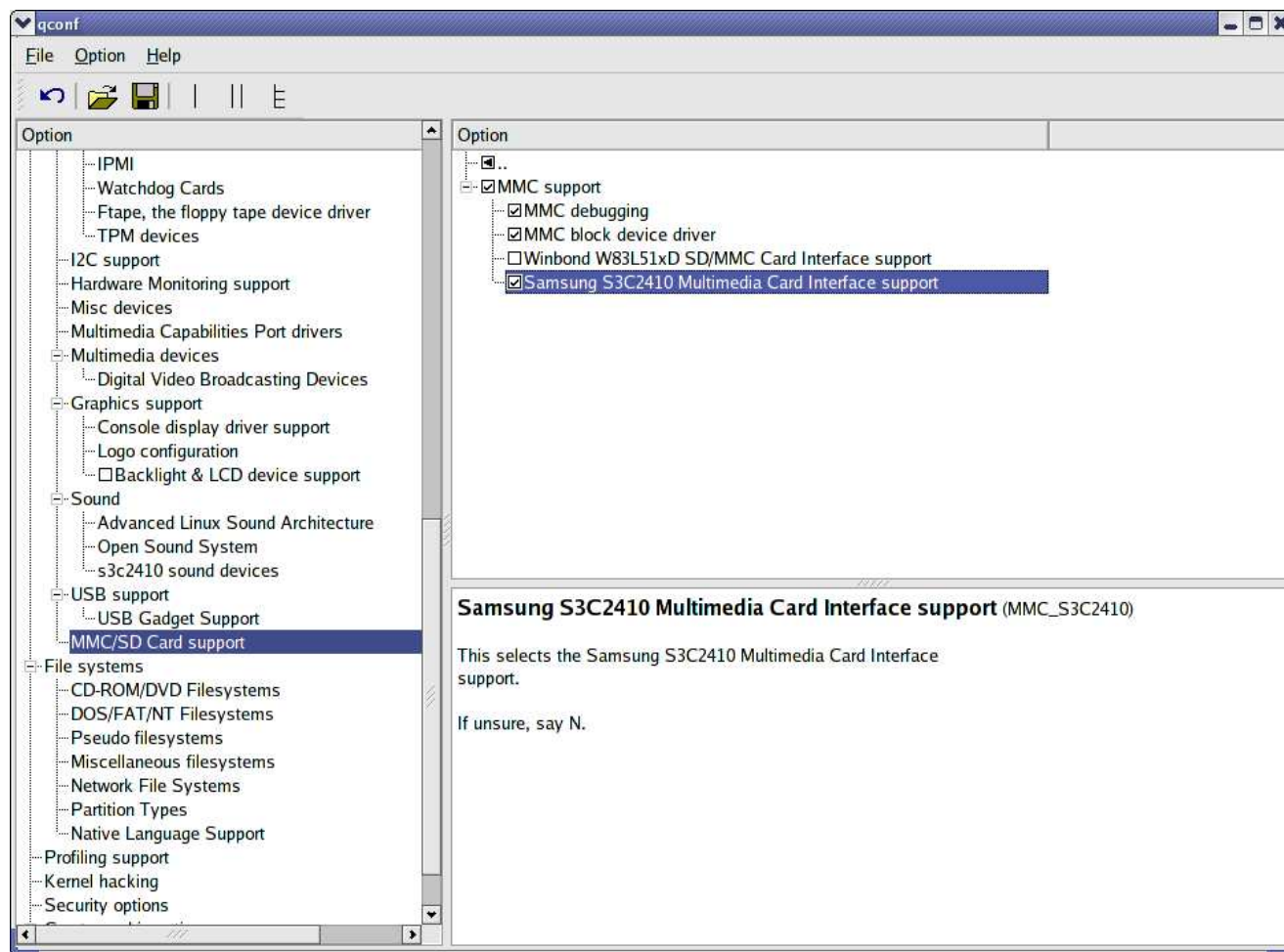


Nous allons à présent compiler le noyau Linux tel qu'il a été préconfiguré par Pragmatec. La carte dont vous avez fait l'acquisition est déjà pré-installé avec un noyau Linux 2.6.14 et tout un ensemble de programmes utilisateurs, il n'est donc pas nécessaire de charger un nouveau noyau dans la carte, pour pouvoir réaliser des applications.

Compilation de Linux

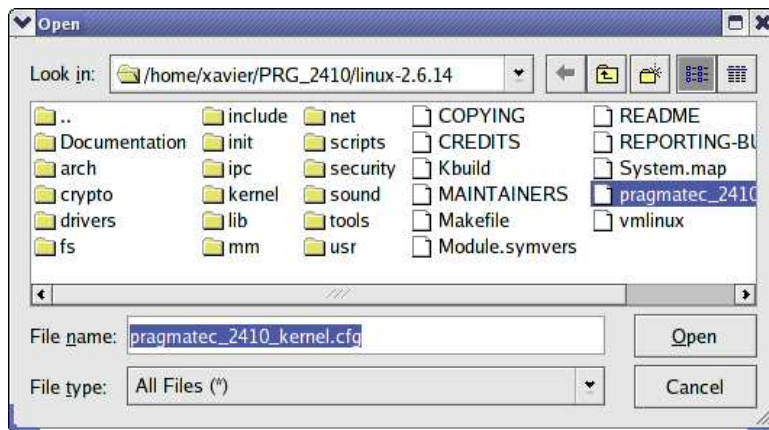
Comme vous allez pouvoir le constater, Linux 2.6 se paramètre et se compile très aisément.

Contrairement au noyau 2.4, la version 2.6 de Linux propose une interface graphique conviviale afin d'accéder à tous les paramètres du noyau dans une seule et même fenêtre :



Linux2.6 - Mise en oeuvre

Cliquez sur l'icône d'ouverture de fichier et sélectionnez le fichier de configuration suivant :



Vous pouvez consulter les différentes options qui ont été validées pour la cible, voir les modifier si vous le souhaitez. Pragmatec a choisi de n'avoir aucun driver généré avec le noyau : tous les drivers sont liés statiquement avec le noyau. De plus, seuls les drivers utiles et validés pour la cible ont été sélectionnés et aucune option superflue n'a été conservés.

Il suffit alors de compiler le noyau à l'aide de la commande « make » (le « make dep » du noyau 2.4 n'est plus nécessaire :

```
xavier@localhost:~/PRG_2410/linux-2.6.14 - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
CC      arch/arm/lib/ucmpdi2.o
AR      arch/arm/lib/lib.a
GEN     .version
CHK     include/linux/compile.h
UPD     include/linux/compile.h
CC      init/version.o
LD      init/built-in.o
LD      .tmp_vmlinux1
KSYM   .tmp_kallsyms1.5
AS      .tmp_kallsyms1.o
LD      .tmp_vmlinux2
KSYM   .tmp_kallsyms2.5
AS      .tmp_kallsyms2.o
LD      vmlinux
SYSMAP  System.map
SYSMAP  .tmp_System.map
OBJCOPY arch/arm/boot/Image
Kernel: arch/arm/boot/Image is ready
AS      arch/arm/boot/compressed/head.o
GZIP   arch/arm/boot/compressed/piggy.gz
AS      arch/arm/boot/compressed/piggy.o
CC      arch/arm/boot/compressed/misc.o
LD      arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
Building modules, stage 2.
MODPOST
[xavier@localhost linux-2.6.14]$ ll arch/arm/boot/zImage
-rwxr-xr-x 1 xavier xavier 1753152 Oct 19 11:40 arch/arm/boot/zImage
[xavier@localhost linux-2.6.14]$ cp arch/arm/boot/zImage ../images/
[xavier@localhost linux-2.6.14]$
```

Copiez ensuite le noyau compressé « zImage » dans le répertoire images si vous souhaitez reflasher la cible.

Linux2.6 - Mise en oeuvre

Génération des partitions NAND

Ensuite il convient de régénérer les 3 partitions MTD3, 4 et 5 de la flash NAND (filesystem de base, répertoires /etc et /usr).

Allez dans le répertoire « partitions » puis dans le répertoire « cramfs_bb-1.2.1 ». L'arborescence que vous y trouverez est celle de votre cible sans les répertoires etc et usr. Vous pouvez bien évidemment la modifier et y rajouter vos propres applications. Attention toutefois de ne pas dépasser la taille de 5Mo (taille de la partition) une fois l'image cramfs créée.

```
xavier@localhost:~/PRG_2410/partitions - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
[xavier@localhost partitions]$ cd cramfs_bb-1.2.1/
[xavier@localhost cramfs_bb-1.2.1]$ ll
total 104
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:27 bin
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:26 dev
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:26 etc
drwxr-xr-x  4 xavier xavier 4096 Oct 19 11:26 home
drwxr-xr-x  3 xavier xavier 4096 Oct 19 11:26 lib
-rwxr-xr-x  1 xavier xavier  468 Oct 19 11:27 linuxrc
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:26 opt
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:26 proc
lrwxrwxrwx  1 xavier xavier    8 Oct 19 11:27 ramdisk -> /dev/shm
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:27 sbin
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:26 sys
lrwxrwxrwx  1 xavier xavier    7 Oct 19 11:26 tmp -> var/tmp
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:27 usr
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:27 var
[xavier@localhost cramfs_bb-1.2.1]$ cd ..
[xavier@localhost partitions]$ ../tools/Linux/mkcramfs cramfs_bb-1.2.1/ rootfs
Directory data: 5812 bytes
Everything: 4756 kilobytes
Super block: 76 bytes
CRC: 51ed7cc9
warning: gids truncated to 8 bits (this may be a security concern)
[xavier@localhost partitions]$ ll
total 4796
drwxr-xr-x 13 xavier xavier  4096 Oct 19 11:27 cramfs_bb-1.2.1
drwxr-xr-x 12 xavier xavier  4096 Oct 19 11:26 etc
-rw-rw-r--  1 xavier xavier 4870144 Oct 19 11:54 rootfs
drwxr-xr-x 11 xavier xavier  4096 Oct 19 11:27 usr
[xavier@localhost partitions]$
```

Pour créer un cramfs à partir du répertoire cramfs_bb-1.2.1, tapez la commande depuis le répertoire des partitions : **../tools/Linux/mkcramfs cramfs_bb-1.2.1/ rootfs**

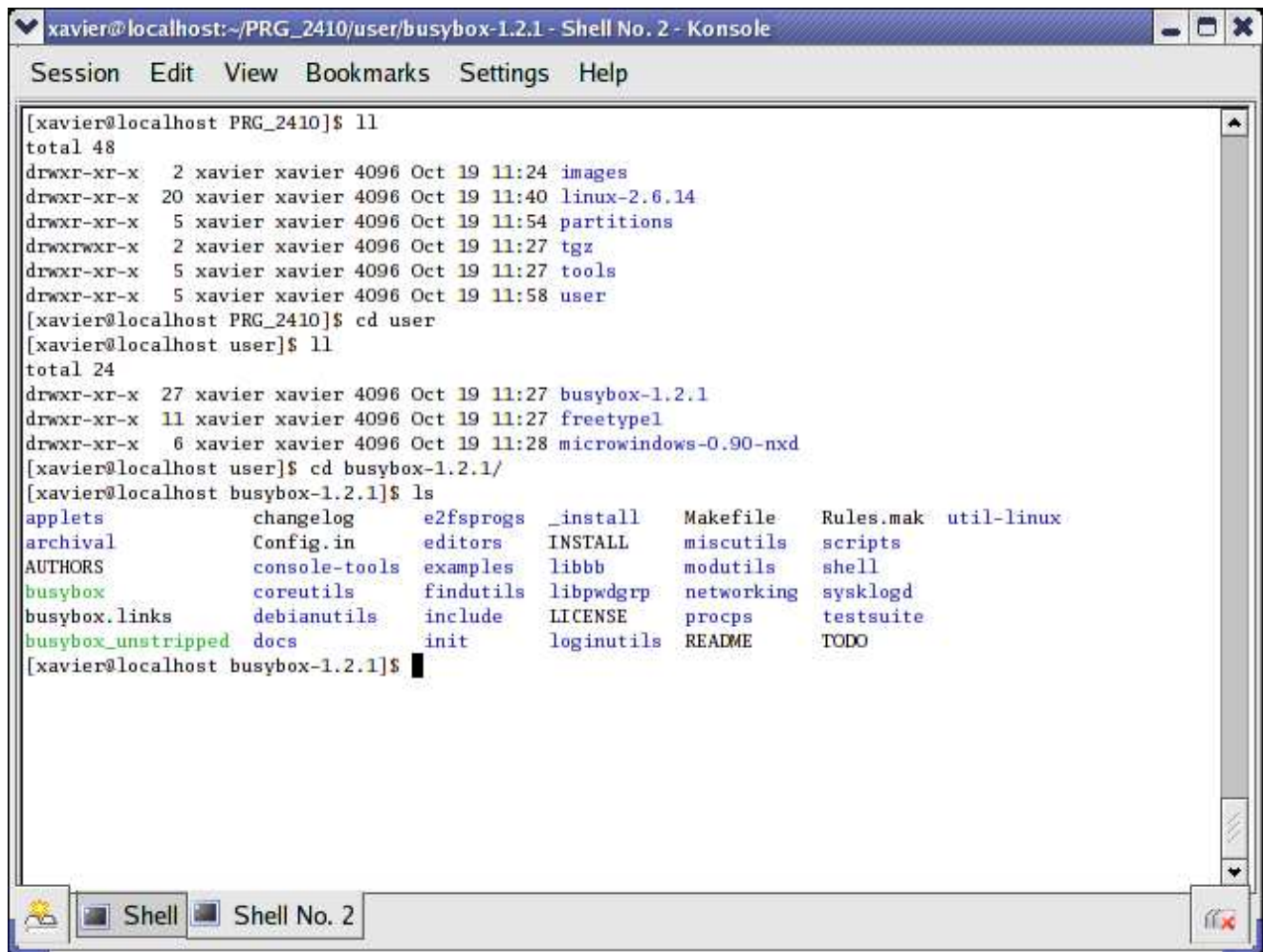
Copiez le fichier « rootfs » dans le répertoire « images » de la distribution.

Les répertoires « etc » et « usr » sont la copie conforme des répertoires qui doivent se trouver sur la cible. Vous pouvez donc directement modifier ces répertoires avant d'en transférer le contenu ultérieurement via NFS.

Linux2.6 - Mise en oeuvre

Busybox 1.2.1

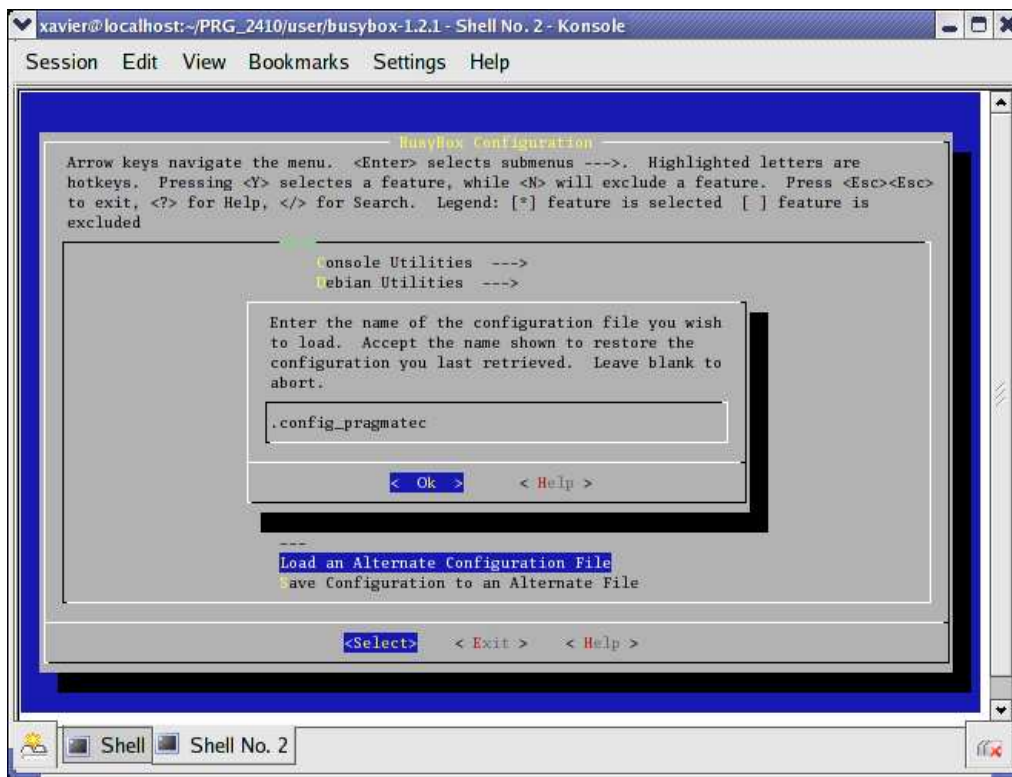
La busybox est une sorte de couteau suisse composé d'un shell basic et d'une multitude de programmes associés qui permettent d'utiliser la cible dans diverses applications.



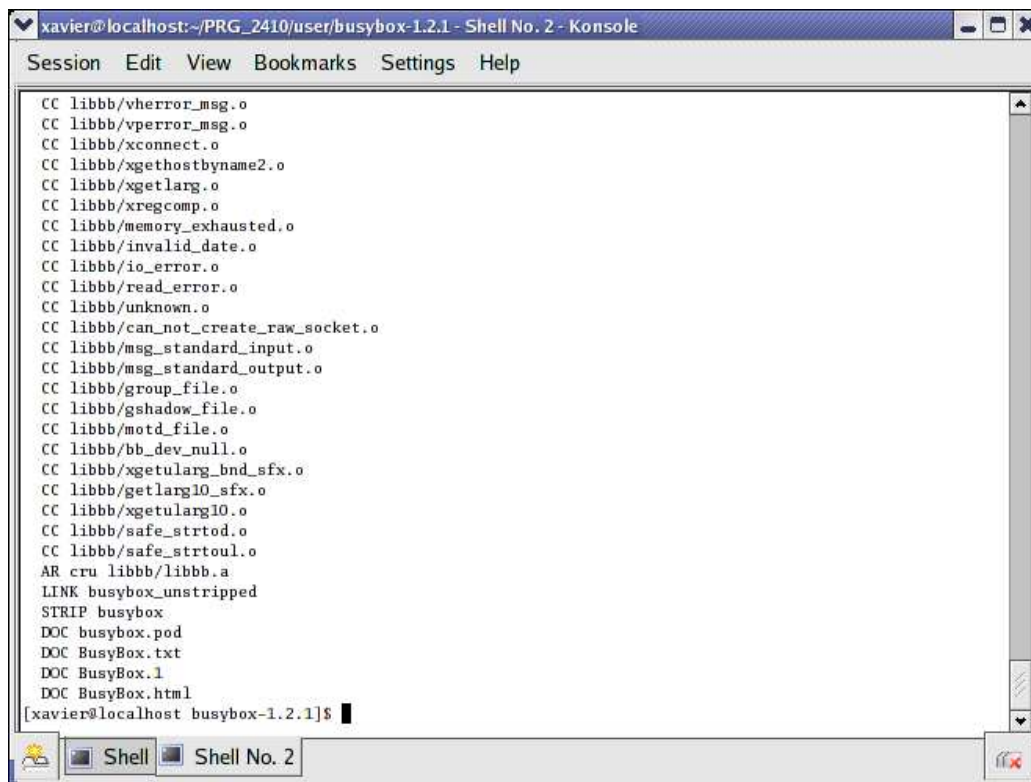
```
xavier@localhost:~/PRG_2410/user/busybox-1.2.1 - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
[xavier@localhost PRG_2410]$ ll
total 48
drwxr-xr-x  2 xavier xavier 4096 Oct 19 11:24 images
drwxr-xr-x 20 xavier xavier 4096 Oct 19 11:40 linux-2.6.14
drwxr-xr-x  5 xavier xavier 4096 Oct 19 11:54 partitions
drwxrwxr-x  2 xavier xavier 4096 Oct 19 11:27 tgz
drwxr-xr-x  5 xavier xavier 4096 Oct 19 11:27 tools
drwxr-xr-x  5 xavier xavier 4096 Oct 19 11:58 user
[xavier@localhost PRG_2410]$ cd user
[xavier@localhost user]$ ll
total 24
drwxr-xr-x 27 xavier xavier 4096 Oct 19 11:27 busybox-1.2.1
drwxr-xr-x 11 xavier xavier 4096 Oct 19 11:27 freetype1
drwxr-xr-x  6 xavier xavier 4096 Oct 19 11:28 microwindows-0.90-nxd
[xavier@localhost user]$ cd busybox-1.2.1/
[xavier@localhost busybox-1.2.1]$ ls
applets          changelog        e2fsprogs        _install         Makefile         Rules.mak        util-linux
archival         Config.in        editors          INSTALL          miscutils        scripts
AUTHORS         console-tools   examples         libbb            modutils         shell
busybox         coreutils       findutils        libpwdgrp        networking       syslogd
busybox.links   debianutils     include          LICENSE          procps           testsuite
busybox_unstripped docs             init             loginutils       README           TODO
[xavier@localhost busybox-1.2.1]$
```

Pour recompiler la busybox il est nécessaire de charger le fichier de configuration « .config_pragmatec ».

Linux2.6 - Mise en oeuvre

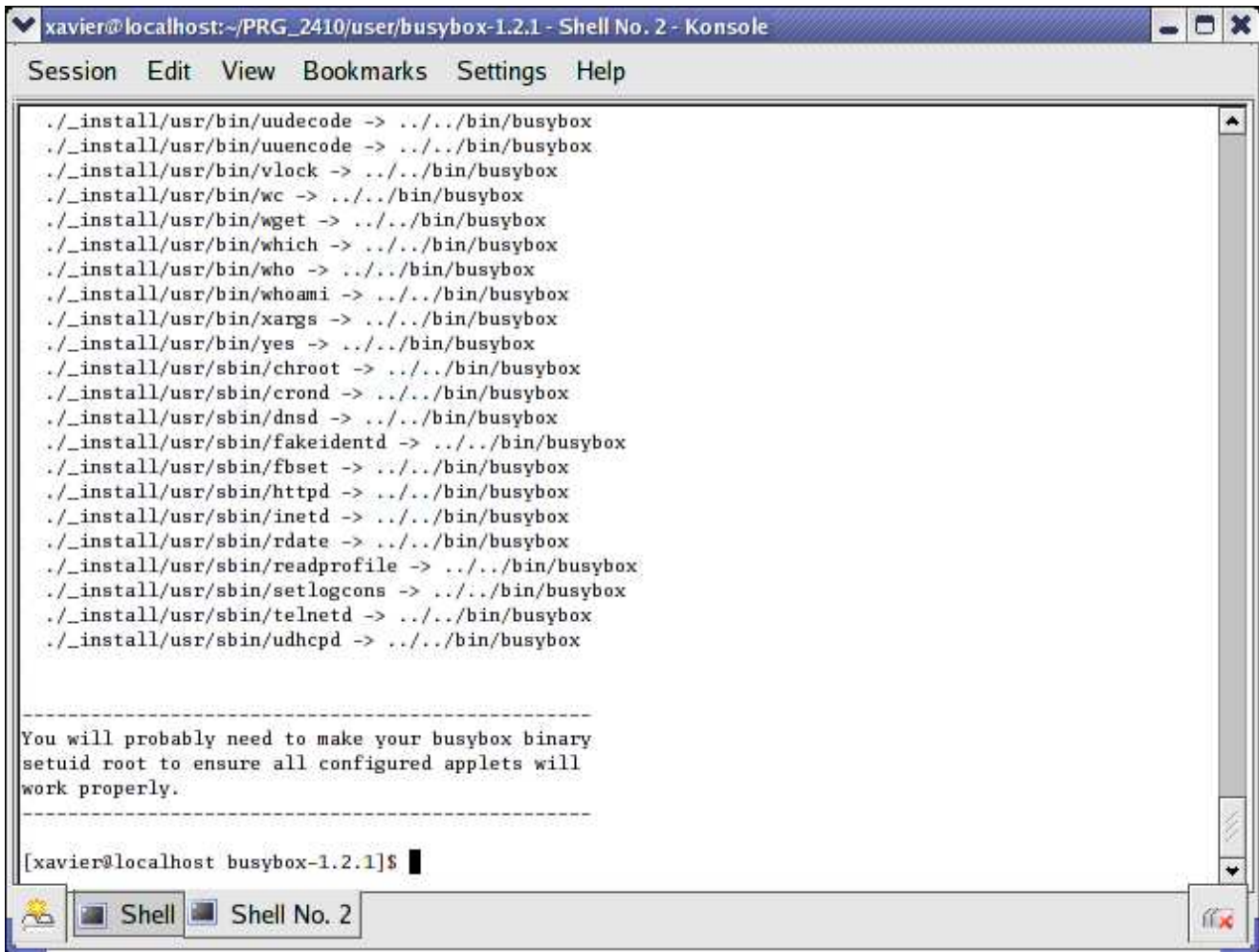


Quittez alors la configuration de busybox en sélectionnant le bouton EXIT puis en sauvegardant la configuration courante. Ensuite lancez la compilation à l'aide de la commande « make » :



Linux2.6 - Mise en oeuvre

La busybox est compilée mais les binaires ne sont pas encore créés et stockés dans un répertoire d'installation. Pour cela vous devez taper la commande « make install ». Tous les binaires se retrouveront alors dans le répertoire « _install ».



```
xavier@localhost:~/PRG_2410/user/busybox-1.2.1 - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help

./_install/usr/bin/uudecode -> ../../bin/busybox
./_install/usr/bin/uuencode -> ../../bin/busybox
./_install/usr/bin/vlock -> ../../bin/busybox
./_install/usr/bin/wc -> ../../bin/busybox
./_install/usr/bin/wget -> ../../bin/busybox
./_install/usr/bin/which -> ../../bin/busybox
./_install/usr/bin/who -> ../../bin/busybox
./_install/usr/bin/whoami -> ../../bin/busybox
./_install/usr/bin/xargs -> ../../bin/busybox
./_install/usr/bin/yes -> ../../bin/busybox
./_install/usr/sbin/chroot -> ../../bin/busybox
./_install/usr/sbin/crond -> ../../bin/busybox
./_install/usr/sbin/dnsd -> ../../bin/busybox
./_install/usr/sbin/fakeidentd -> ../../bin/busybox
./_install/usr/sbin/fbset -> ../../bin/busybox
./_install/usr/sbin/httpd -> ../../bin/busybox
./_install/usr/sbin/inetd -> ../../bin/busybox
./_install/usr/sbin/rdate -> ../../bin/busybox
./_install/usr/sbin/readprofile -> ../../bin/busybox
./_install/usr/sbin/setlogcons -> ../../bin/busybox
./_install/usr/sbin/telnetd -> ../../bin/busybox
./_install/usr/sbin/udhcpd -> ../../bin/busybox

-----
You will probably need to make your busybox binary
setuid root to ensure all configured applets will
work properly.
-----

[xavier@localhost busybox-1.2.1]$
```

Si vous avez modifié le contenu de la busybox vous devez alors transférer ce contenu dans les répertoires cramfs (/bin par exemple) et usr (/usr/sbin par exemple) afin de garder une cohérence entre la busybox et les répertoires des partitions NAND.

Linux2.6 - Mise en oeuvre

Microwindows

Microwindows est une application graphique client/serveur qui présente une API permettant de dessiner directement dans le Frame Buffer (fichier d'interface du LCD). Mieux encore, ceci peut se faire au travers du réseau Ethernet, c'est-à-dire que le serveur peut être lancé sur un PC distant alors que l'application est lancée en local sur la cible, ... dans le cas où vous ne disposeriez pas de LCD.

On peut aussi imaginer lancer le serveur sur la cible afin d'utiliser le LCD de la cible et de développer l'application cliente depuis la station en code natif i86 (pour un debug plus rapide et facilité).

```
xavier@localhost:~/PRG_2410/user/microwindows-0.90-nxd/src - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
[xavier@localhost user]$ ll
total 24
drwxr-xr-x 27 xavier xavier 4096 Oct 19 12:02 busybox-1.2.1
drwxr-xr-x 11 xavier xavier 4096 Oct 19 11:27 freetype1
drwxr-xr-x 6 xavier xavier 4096 Oct 19 11:28 microwindows-0.90-nxd
[xavier@localhost user]$ cd microwindows-0.90-nxd/src
[xavier@localhost src]$ ls
Arch.rules  Configs          Doxyfile-nanox  fontdemo.sh  lib             nanox           Version
bin         CONTENTS        drivers         fonts        LICENSE        nnterm.sh      vnc.sh
BUGS       contrib        ecos           grabdemo.sh  Makefile       nxview.sh      xconfigure
ChangeLog  CREDITS        engine         grdemo.sh   Makefile.rules rtems          tcmwin.mak
chess.sh   demo2.sh       FIXME         include     mcmwin.mak    tcnanox.mak
config     demos          fontdemo2.sh  indent.sh   mcnanox.mak  test
config.arm demo.sh         fontdemo3.sh  INSTALL    mouse.sh
config.linux Doxyfile-internal fontdemo4.sh  launcher.sh mwin          TODO
[xavier@localhost src]$ ll config*
-rw-r--r-- 1 xavier xavier 13561 Oct 19 11:28 config
-rw-r--r-- 1 xavier xavier 13561 Oct 19 11:28 config.arm
-rw-r--r-- 1 xavier xavier 13487 Oct 19 11:28 config.linux
[xavier@localhost src]$ vi config
```

Dans le répertoire « src » de Microwindows, il existe 3 fichiers de configuration :

- **config** : fichier utilisé à la compilation (copie du fichier config.arm par défaut)
- **config.arm** : fichier de configuration pour la cible
- **config.linux** : fichier de configuration pour la station

Linux2.6 - Mise en oeuvre

4 Programmation de la cible



Vous êtes parvenu à recompiler votre propre distribution Linux 2.6 pour votre cible ARM9. Si cette distribution est différente de celle livrée avec votre carte, il vous faut maintenant re-programmer la carte ARM9.

Pour cela il vous faudra un cordon USB pour connecter votre carte ARM9 (connecteur USB device) à un PC sous Windows™ (connecteur USB host), ainsi qu'un cordon RS232 croisé pour communiquer avec votre carte via HyperTerminal.

Préalable

Si vous souhaitez re-programmer la carte ARM9, démarrez la normalement jusqu'au prompt du shell. Là effacez le contenu des répertoires « /etc » et « /usr » :

```
#!/ rm -rf /etc/*  
#!/ rm -rf /usr/*  
#!/ rm -rf /home/*
```

Resetez la carte. Appuyez sur la touche Enter sous HyperTerminal pour stopper l'autoboot. Appuyez sur la touche « 0 » pour sélectionner « USB download file ».

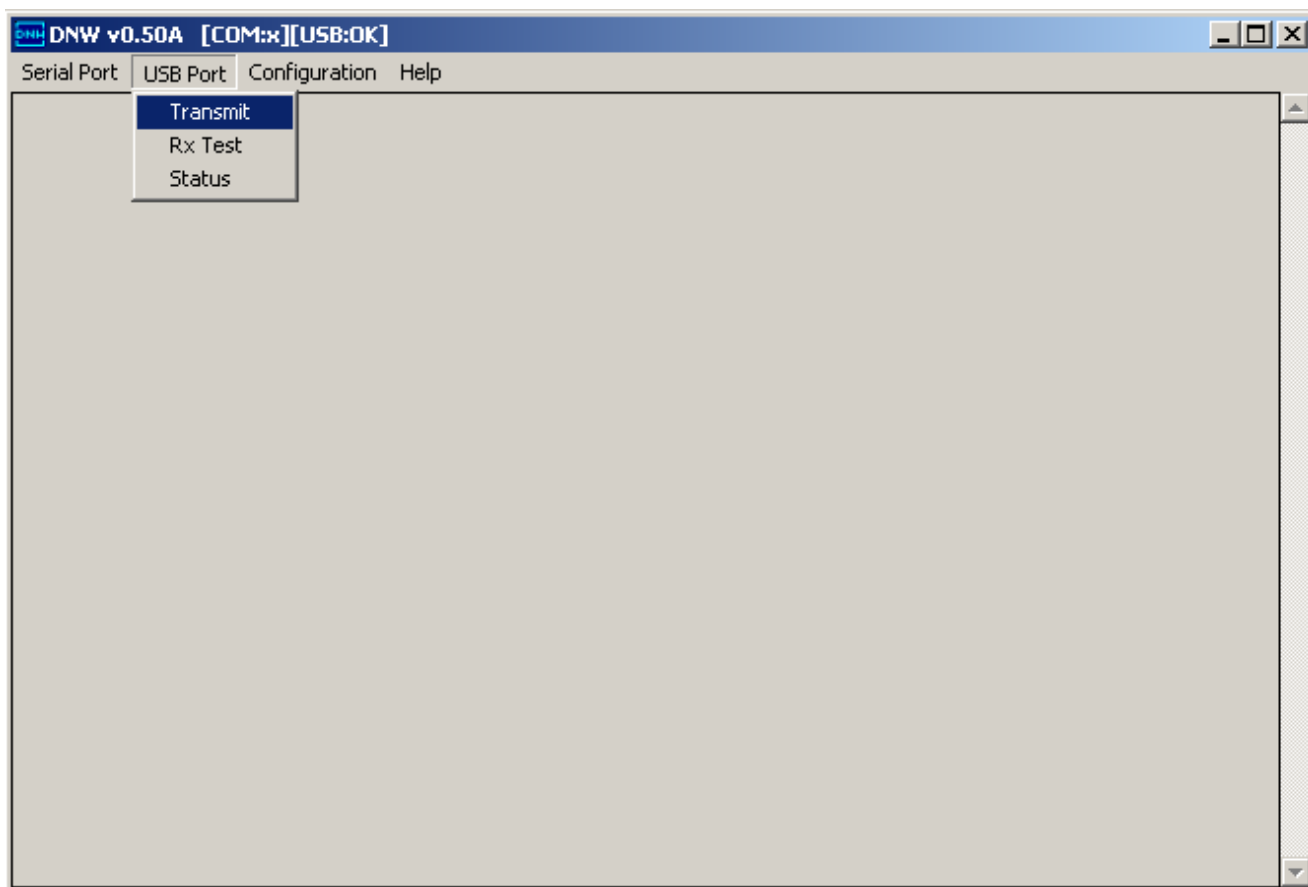
Connectez le cordon USB à la carte puis au PC sous Windows. Si un driver vous est demandé, sélectionnez « secbulk.inf » livré sur le CDROM (répertoire tools/usb). Au final, vous devriez obtenir le message « Now USB is connected » :

```
comusb - HyperTerminal  
Fichier Edition Affichage Appel Transfert ?  
Please select function :  
0 : USB download file  
1 : Uart download file  
2 : Write Nand flash with download file  
3 : Load Program from Nand flash and run  
4 : Erase Nand flash regions  
5 : Set AutoBoot parameter,1:linux 2:wince  
6 : Test Power off  
7 : test SDRAM Memory  
dd  
Please select function :  
0 : USB download file  
1 : Uart download file  
2 : Write Nand flash with download file  
3 : Load Program from Nand flash and run  
4 : Erase Nand flash regions  
5 : Set AutoBoot parameter,1:linux 2:wince  
6 : Test Power off  
7 : test SDRAM Memory  
0  
USB download file, press Esc key to exit  
Now USB is connected._  
00:17:23 connecté Détection auto 115200 8-N-1 DÉFIL Maj Num Capturer Écho
```

Linux2.6 - Mise en oeuvre

Programmation du noyau

Pour transférer le nouveau noyau Linux dans votre cible il vous faut utiliser le programme DNW.exe sous Windows™. Une fois connecté à votre cible via l'USB, vous pouvez sélectionner la commande « USB Port / Transmit » pour envoyer un fichier à l'ARM9 :



Sélectionnez le fichier zImage que vous avez généré précédemment.

Durant le transfert vous verrez s'afficher des caractères sous HyperTerminal témoignant de la progression du transfert. A l'issue du transfert la carte vous demande si vous souhaitez lancer le binaire téléchargé.

Répondez par non (touche « n ») afin de pouvoir le programmer dans une des partitions de la flash NAND.

Linux2.6 - Mise en oeuvre

Tapez alors sur la touche « 2 » afin de sélectionner l'écriture en flash NAND.

```
comusb - HyperTerminal
Fichier Edition Affichage Appel Transfert ?
Download O.K.
Do you want to run? [y/n] : n
Please select function :
0 : USB download file
1 : Uart download file
2 : Write Nand flash with download file
3 : Load Program from Nand flash and run
4 : Erase Nand flash regions
5 : Set AutoBoot parameter,1:linux 2:wince
6 : Test Power off
7 : test SDRAM Memory
2
Read chip id = ec76
Nand flash status = c0
Please select which region to write : Esc to abort
0 : offset 0x0      , size 0x20000  [boot]
1 : offset 0x20000 , size 0x300000 [kernel]
2 : offset 0x320000 , size 0x500000 [rootfs]
3 : offset 0x820000 , size 0x100000 [etc]
4 : offset 0x920000 , size 0xb00000 [user]
5 : offset 0x1420000, size 0x2b00000 [qt]
00:18:55 connecté  Détection auto  115200 8-N-1  DÉFIL  Maj  Num  Capturer  Écho
```

Enfin choisissez la partition « kernel » pour y stocker votre nouvelle image du noyau.

Cramfs

Nous allons utiliser le même procédé pour transférer le nouveau CRAMFS (appelé rootfs dans nos exemple précédent).

Tapez à nouveau la touche « 2 » pour pouvoir transférer le fichier « rootfs » puis « n » pour éviter toute exécution de code.

Enfin tapez la touche « 2 » afin de sélectionner la partition « rootfs ».

A l'issue de la programmation le BIOS a redémarré. Tapez sur la touche « 3 » pour lancer le noyau Linux.

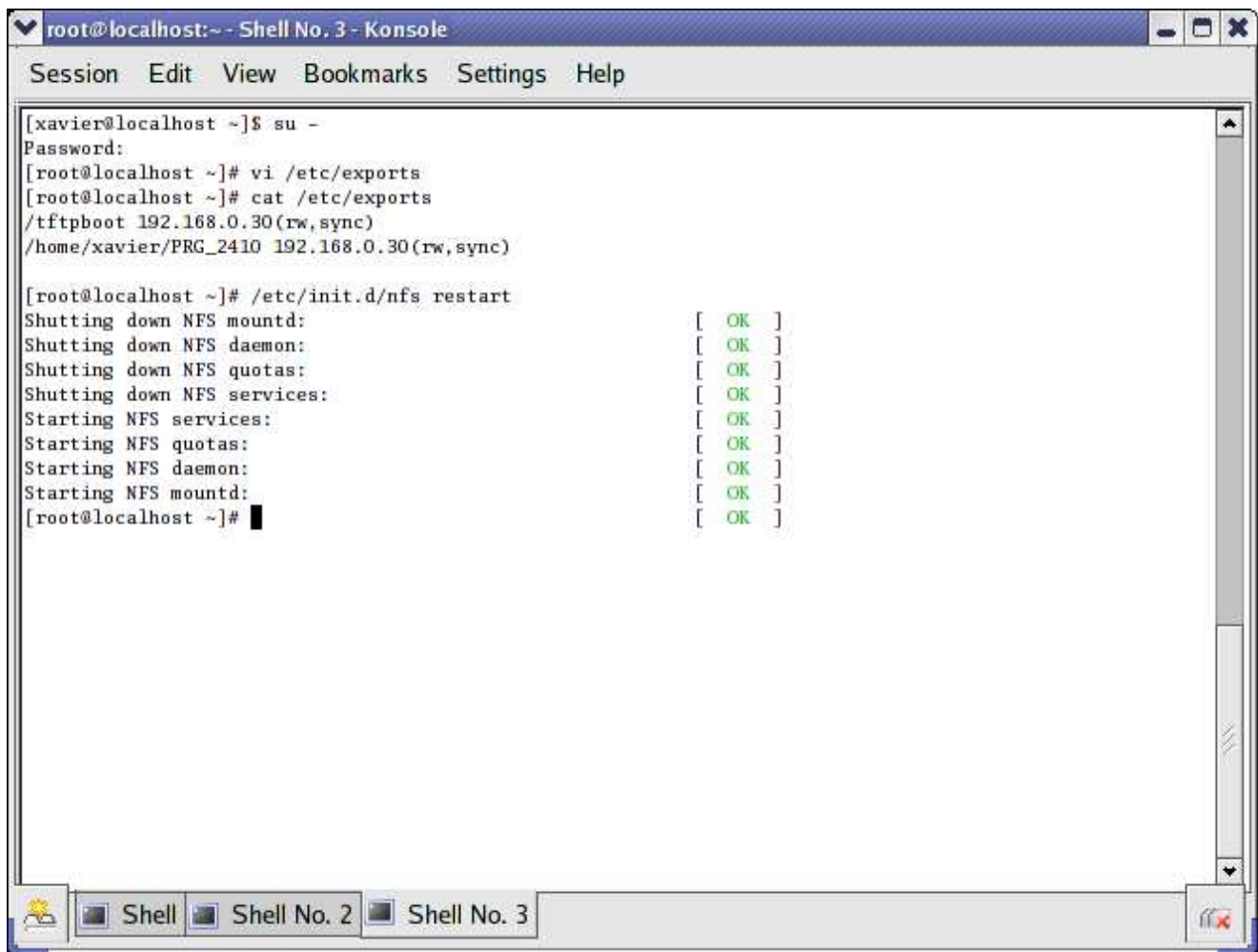
Linux2.6 - Mise en oeuvre

Répertoires etc, usr et home

Afin de mettre à jour les partitions « etc » et « usr » nous allons utiliser le noyau Linux plutôt que de transférer de nouvelles images à l'aide du BIOS.

La carte devrait être à présent démarrée.

Côté station Linux, vous devez tout d'abord modifier le fichier « /etc/exports » et relancer le service NFS comme suit (passez « root ») :



```
root@localhost:~ - Shell No. 3 - Konsole
Session Edit View Bookmarks Settings Help

[xavier@localhost ~]$ su -
Password:
[root@localhost ~]# vi /etc/exports
[root@localhost ~]# cat /etc/exports
/tftpboot 192.168.0.30(rw, sync)
/home/xavier/PRG_2410 192.168.0.30(rw, sync)

[root@localhost ~]# /etc/init.d/nfs restart
Shutting down NFS mountd:           [ OK ]
Shutting down NFS daemon:           [ OK ]
Shutting down NFS quotas:           [ OK ]
Shutting down NFS services:         [ OK ]
Starting NFS services:               [ OK ]
Starting NFS quotas:                 [ OK ]
Starting NFS daemon:                 [ OK ]
Starting NFS mountd:                 [ OK ]
[root@localhost ~]#
```

Le fichier « /etc/exports » contient la liste des répertoires accessibles par NFS. Nous avons spécifié le répertoire de la distribution Pragmatec afin de pouvoir accéder à tout son contenu.

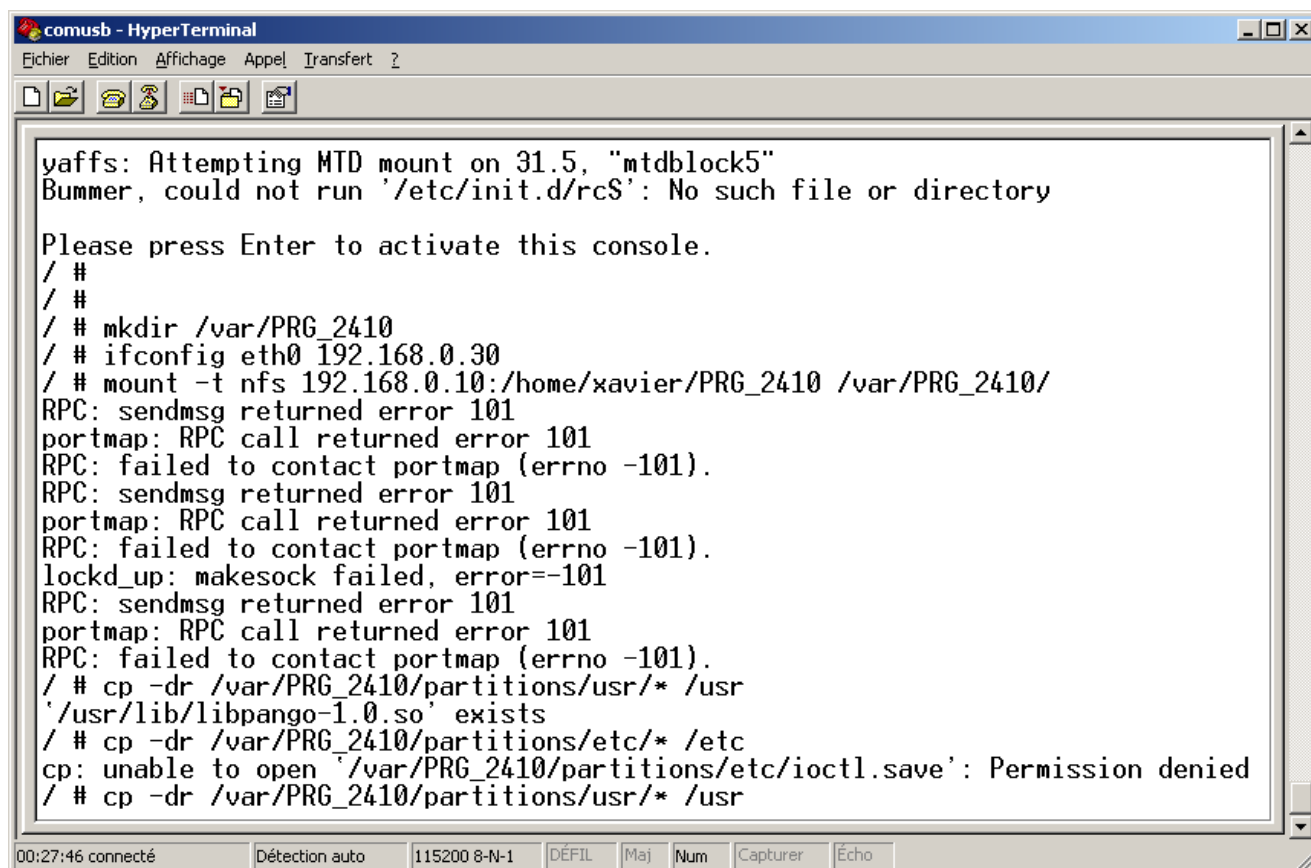
Le lancement du service NFS doit se terminer par un ensemble complet de « OK ». Si c'est la première fois que vous lancez le service NFS, il est possible d'obtenir des « FAILED ». Dans ce cas relancez à nouveau le service NFS.

Linux2.6 - Mise en oeuvre

Votre carte étant démarrée, vous pouvez alors procéder au montage NFS.

Attention toutefois, il n'y a plus de fichiers dans les répertoires /etc, /usr et /home. Vous avez donc perdu vos profile ainsi qu'une partie des scripts de démarrage. Il est donc nécessaire de paramétrer le réseau manuellement avant de monter le NFS.

Procédez comme suit :



```
comusb - HyperTerminal
Fichier Edition Affichage Appel Transfert ?

yaffs: Attempting MTD mount on 31.5, "mtdblock5"
Bummer, could not run '/etc/init.d/rcS': No such file or directory

Please press Enter to activate this console.
/ #
/ #
/ # mkdir /var/PRG_2410
/ # ifconfig eth0 192.168.0.30
/ # mount -t nfs 192.168.0.10:/home/xavier/PRG_2410 /var/PRG_2410/
RPC: sendmsg returned error 101
portmap: RPC call returned error 101
RPC: failed to contact portmap (errno -101).
RPC: sendmsg returned error 101
portmap: RPC call returned error 101
RPC: failed to contact portmap (errno -101).
lockd_up: makesock failed, error=-101
RPC: sendmsg returned error 101
portmap: RPC call returned error 101
RPC: failed to contact portmap (errno -101).
/ # cp -dr /var/PRG_2410/partitions/usr/* /usr
'/usr/lib/libpango-1.0.so' exists
/ # cp -dr /var/PRG_2410/partitions/etc/* /etc
cp: unable to open '/var/PRG_2410/partitions/etc/ioctl.save': Permission denied
/ # cp -dr /var/PRG_2410/partitions/usr/* /usr

00:27:46 connecté   Détection auto   115200 8-N-1   DÉFIL   Maj   Num   Capturer   Écho
```

Ne tenez pas compte des erreurs, elles disparaîtront au prochain redémarrage de la carte.

Le point de montage NFS choisi est « /var/PRG_2410 » pour rappeler le répertoire de la distribution sur la machine hôte. Il devient alors possible de copier les fichiers des partitions « etc », « usr » et « /home » du point de montage vers les répertoires « /etc », « /usr » et « /home » réels de la carte.

Si le montage NFS ne peut se faire, lancer la commande « **portmap** » préalablement.

Attention toutefois : il importe d'utiliser les options « -dr » afin de conserver les liens symboliques (/usr/sbin par exemple).

Redémarrez votre carte afin de contrôler que tout c'est bien passé.

Linux2.6 - Mise en oeuvre

5 Installation d'Eclipse



Eclipse est un environnement de développement intégré ouvert et gratuit. Il permet de gérer des projets couplets de façon graphique, et y compris de débiter le code développé.

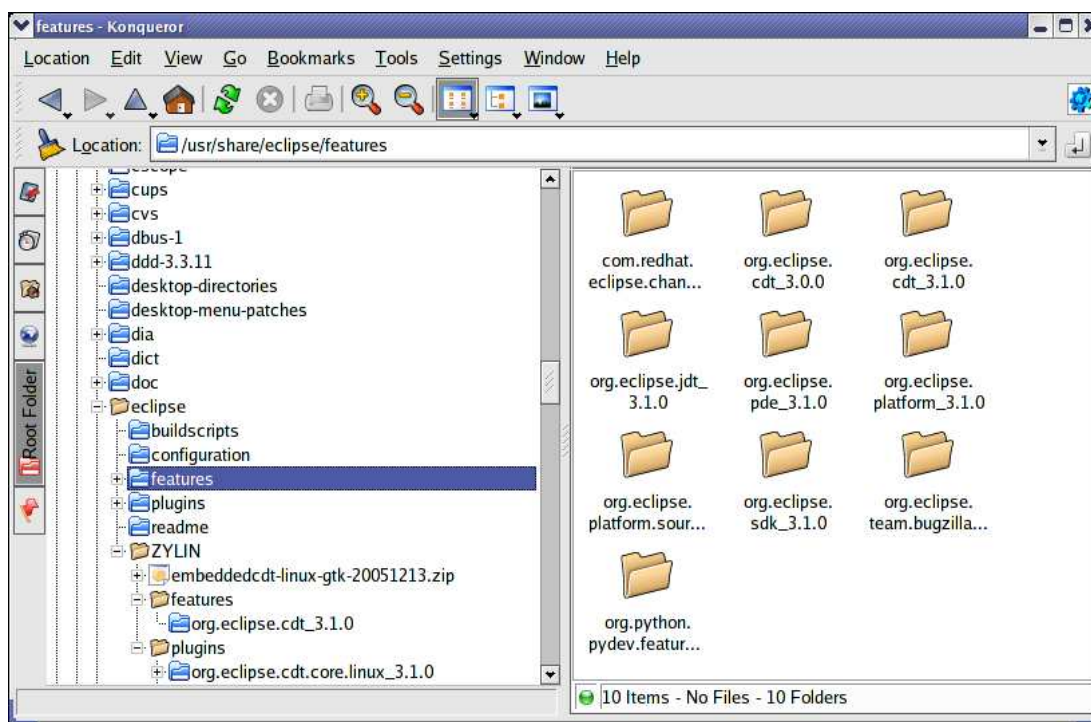


Nous allons utiliser Eclipse pour développer nos programmes et pour les débiter directement depuis la cible via le réseau Ethernet.

Pré-requis

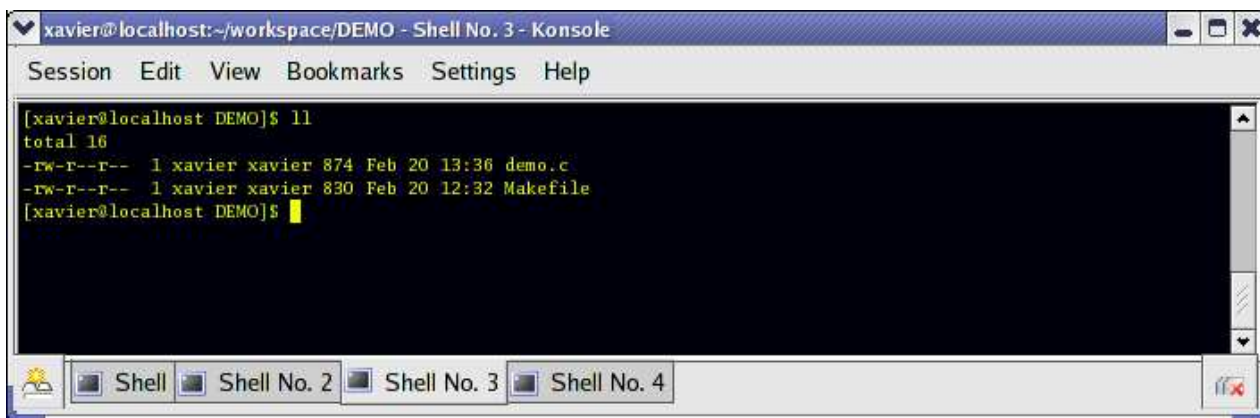
Le problème d'Eclipse est qu'il n'est pas conçu à la base pour effectuer un débiter distant via le réseau Ethernet, et pour corriger ce problème il est nécessaire d'y ajouter un correctif sous la forme d'un plug-in.

Pour cela utilisez le plug-in ZYLIN qui se trouve sur le CDROM (**embeddedcdt-linux-gtk-20051213**). Copiez les répertoires « features » et « plugins » sous les répertoires correspondant de Eclipse (présent sous **/usr/share/eclipse**).



Linux2.6 - Mise en oeuvre

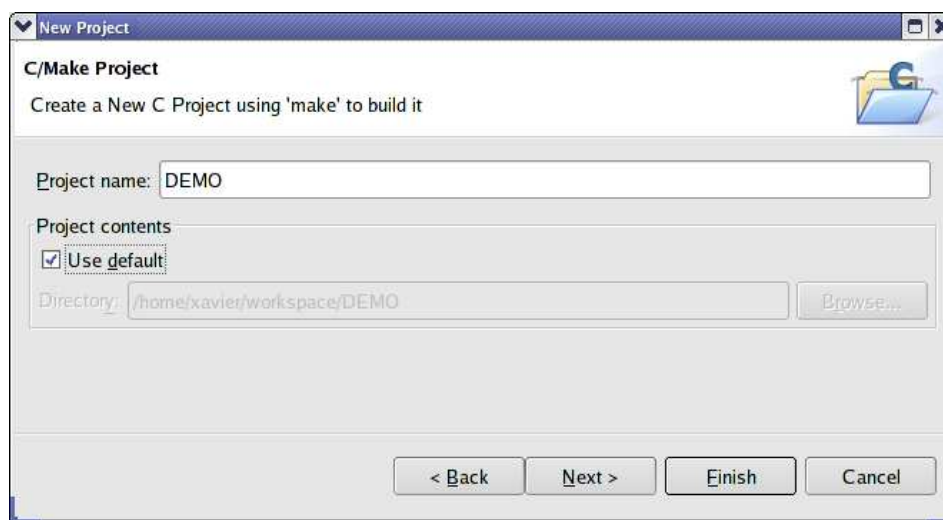
Ceci étant fait commencez par copier le projet DEMO sous votre workspace Eclipse (par exemple /home/xavier/workspace).



```
xavier@localhost:~/workspace/DEMO - Shell No. 3 - Konsole
Session Edit View Bookmarks Settings Help

[xavier@localhost DEMO]$ ll
total 16
-rw-r--r-- 1 xavier xavier 874 Feb 20 13:36 demo.c
-rw-r--r-- 1 xavier xavier 830 Feb 20 12:32 Makefile
[xavier@localhost DEMO]$
```

Vous allez à présent créer le projet sous Eclipse (**File/New/Project...Standart Make C project**) :



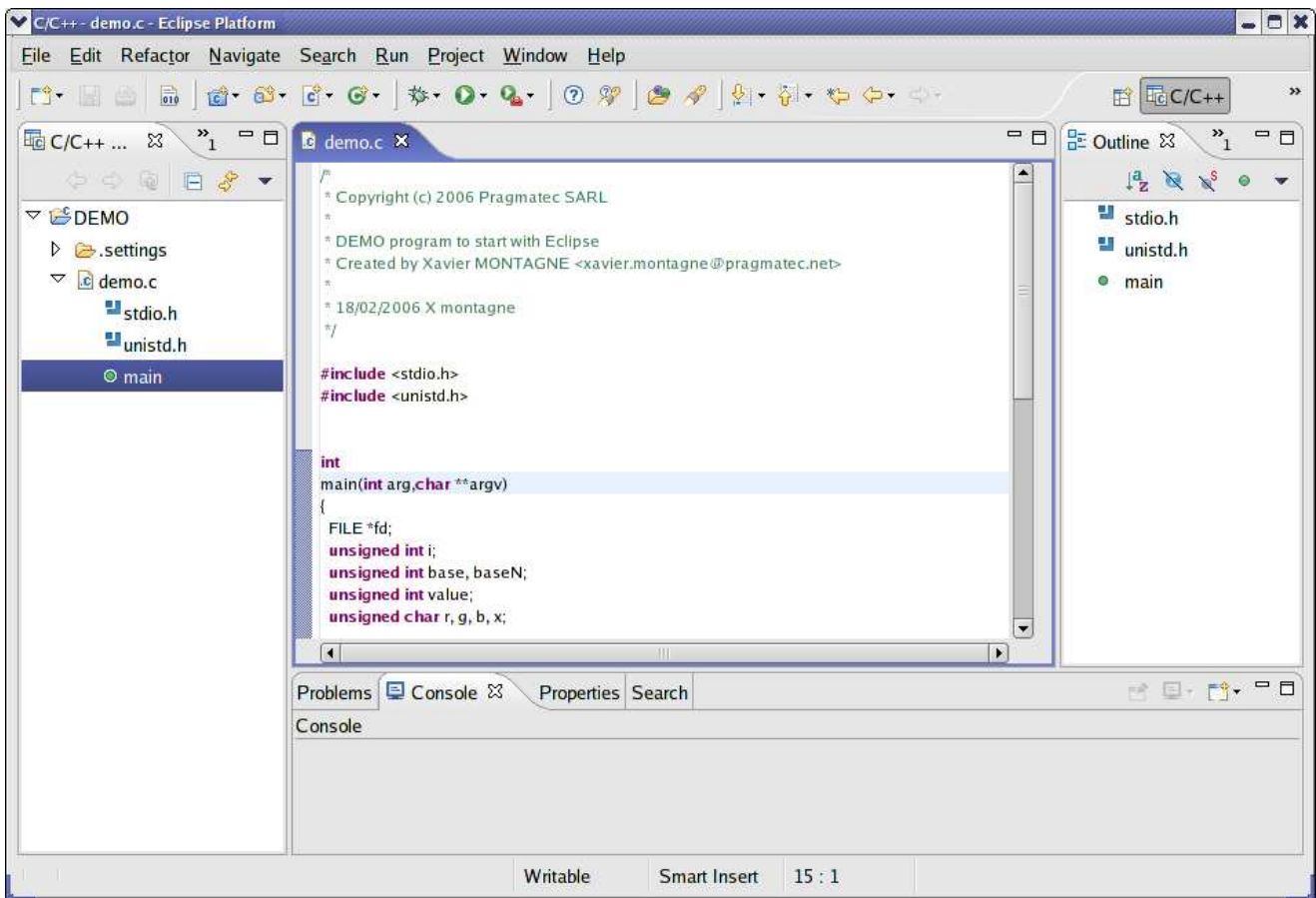
Puis cliquez sur « **Finished** ».

Importez les fichiers du projet fourni en exemple dans votre environnement Eclipse. Pour cela cliquez à l'aide du bouton droit sur le nom de votre projet et choisissez « Import... ».

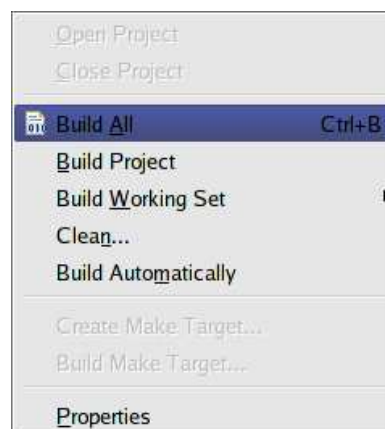
Faites de même pour activer le sous-menu « Build make target... » et sélectionnez les modes de compilation (all et clean devraient suffire).

Linux2.6 - Mise en oeuvre

Vous devez obtenir la vue suivante :



Enfin compilez votre projet à l'aide du menu « **Project** » :



L'affichage dans la fenêtre « Console » doit vous renseigner sur le résultat de la compilation.

Linux2.6 - Mise en oeuvre

Debug distant

Pour utiliser le débogueur il faut lancer d'une part gdbserver sur la cible, et d'autre part arm-linux-gdb sur la station via Eclipse.

Préparation de la cible :

Commencez par transférer le programme « gdbserver » sur la cible, puis le programme « demo ». Lancez la session de debug à l'aide de la commande suivante :

```
gdbserver 192.168.0.10 :2345 ./demo
```

Pour déboguer via le réseau, il est important de bénéficier d'un bon environnement de travail entre votre cible et votre station de développement.

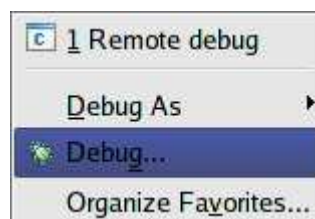
Afin de transférer le programme à déboguer sur la cible, vous pouvez vous servir d'une clef USB ou d'une Sdcard. Cela s'avère assez fastidieux....

Le mieux est encore d'utiliser le réseau Ethernet. Vous pouvez par exemple utiliser le protocole FTP. Mieux encore vous pouvez établir un montage NFS sur votre répertoire de développement (le « workspace » d'Eclipse par exemple) mais vous devrez à chaque fois recopier le fichier distant sur un répertoire local de la cible pour pouvoir l'exécuter.

Préparation d'Eclipse :

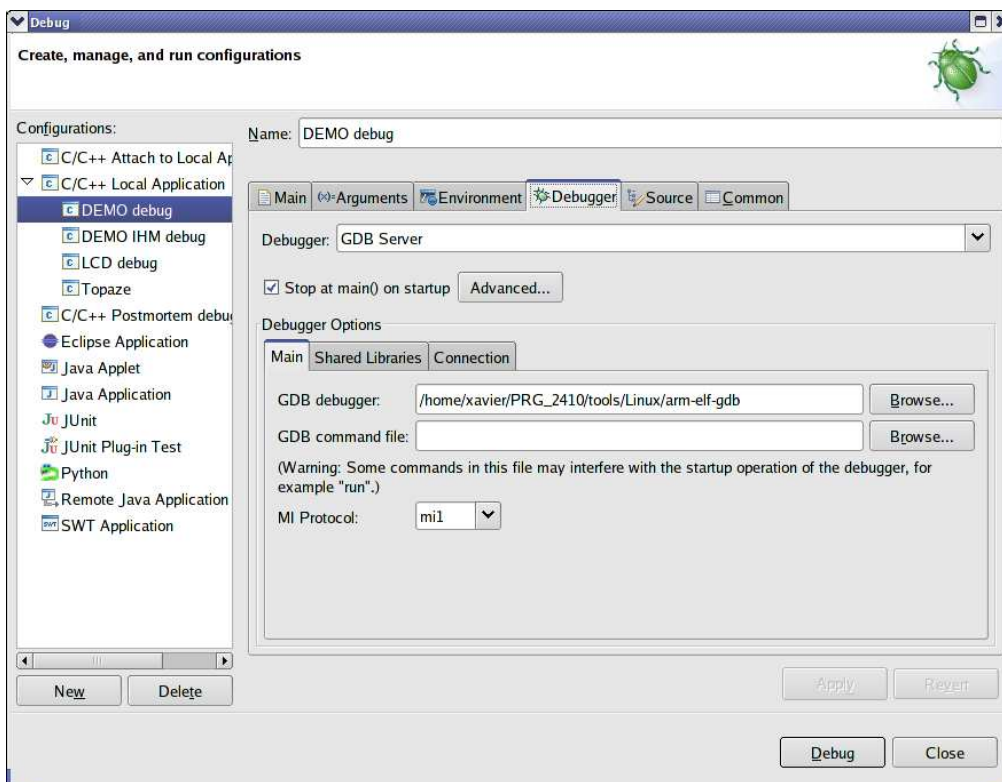
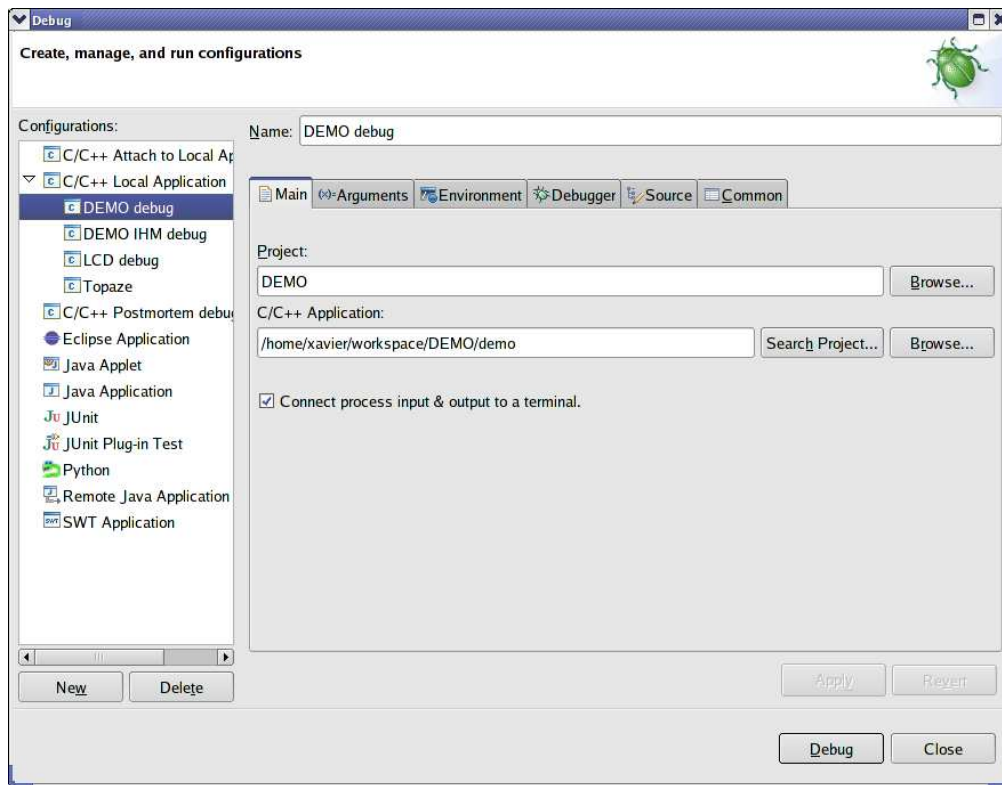
Pour pouvoir déboguer notre programme sous Eclipse il convient de créer une nouvelle perspective (une vue en quelques sortes) consacrée au debug.

Au milieu de votre barre d'outils Eclipse vous trouverez l'icone d'un cafard pour symboliser le debug. Cliquez sur la flèche descendante qui se situe à sa droite, puis sur « **Debug...** » :

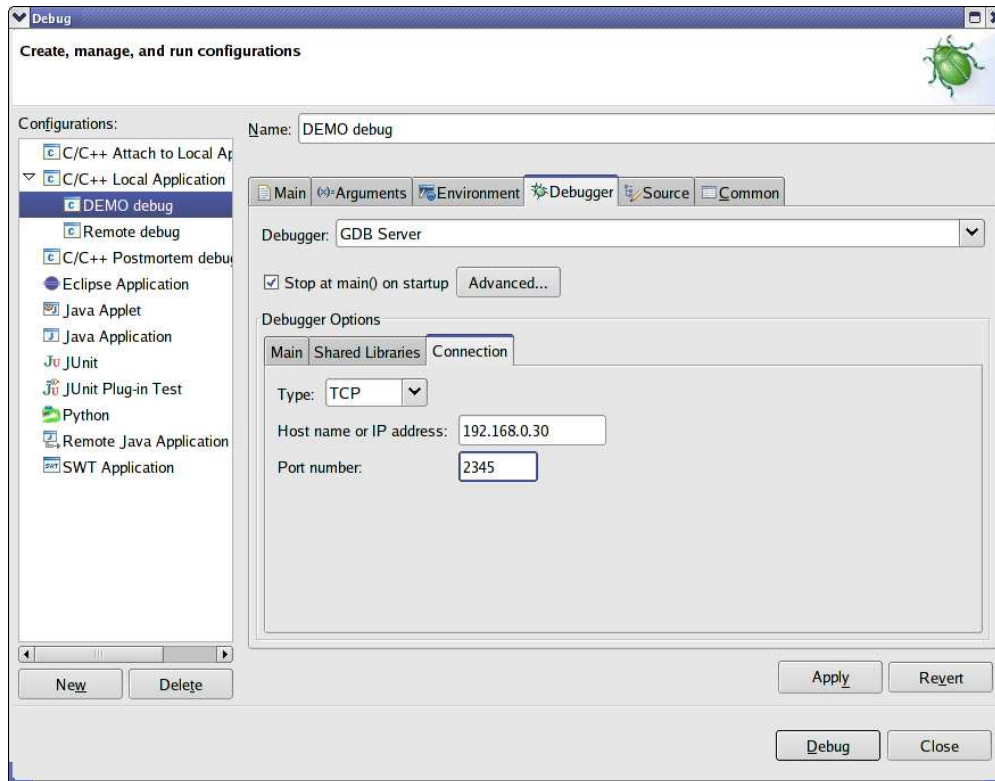


Linux2.6 - Mise en oeuvre

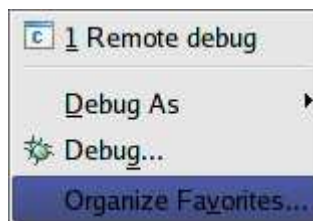
Dans la nouvelle fenêtre, sélectionnez « C/C++ Local application » et renseigner la nouvelle perspective comme suit :



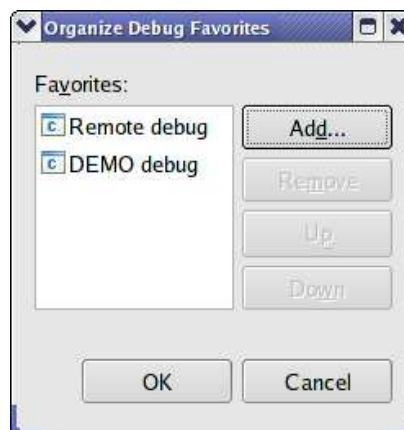
Linux2.6 - Mise en oeuvre



Il faut maintenant ajouter votre nouvelle perspective en cliquant à nouveau sur la flèche descendante de l'icône de debug :



Puis cliquez sur « Add... » :



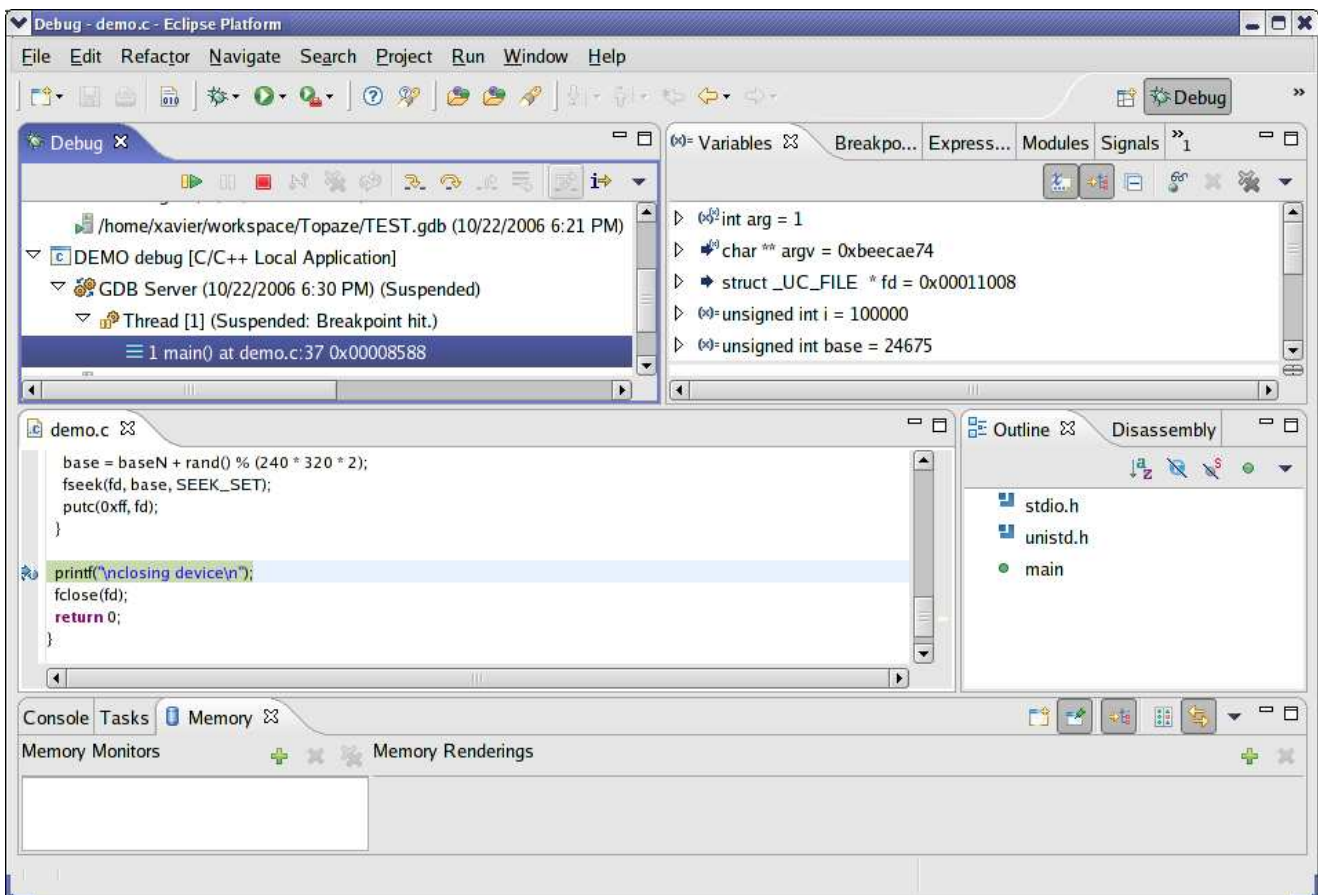
Linux2.6 - Mise en oeuvre

Vous pouvez maintenant changer de perspective en cliquant sur les doubles flèches qui se situent en haut et à droite d'Eclipse. Sélectionnez la perspective « **Debug** ».

Vous pouvez à présent tenter une connexion à la cible en mode debug en cliquant encore une fois sur la flèche descendante de l'icône debug, puis en sélectionnant le menu « **DEMO debug** » :



Vous devez obtenir la vue suivante sous Eclipse :



Vous pouvez à présent vous déplacer en pas à pas en cliquant sur la flèche du milieu de la fenêtre Debug, inspecter les variables locales à la fonction courante, placer des breakpoints... N'hésitez pas à manipuler le débogueur sous Eclipse, quitte à relancer le debug plusieurs fois, en prenant bien soin de lancer **D'ABORD** le débogueur sur la cible à l'aide du programme « **gdbserver** ».

Linux2.6 - Mise en oeuvre

Vous pouvez aussi directement modifier le code dans la perspective de debug et recompiler le code à la volée. Ensuite relancer « **gdbserver** » sur la cible pour transférer le nouveau programme automatiquement et lancer le mode debug.

L'application utilise le framebuffer qui est la mémoire vidéo primaire. Au démarrage de votre noyau Linux vous avez pu peut-être constater qu'il y avait un logo Pragmatec qui s'affichait sur le LCD 65000 couleurs. Le programme *demo* n'a d'autre but que d'essayer d'effacer le logo Pragmatec. Au bout d'un certain temps voici le résultat obtenu :



Avant



Après

Linux2.6 - Mise en oeuvre

Partage réseau par NFS

Si vous souhaitez échanger souvent des fichiers entre votre station de travail et votre cible, alors le système NFS (Network File System) est fait pour vous !

Il est possible grâce au NFS de faire qu'un répertoire quelconque soit un répertoire partagé à la fois par la cible et par la station. Ainsi votre PC et votre cible auront tous 2 accès à ce répertoire comme s'il s'agissait d'un répertoire local.

Le système NFS est natif dans le noyau Linux, si bien que votre station de travail Linux et votre cible peuvent gérer NFS. Toutefois il y a quelques préalables :

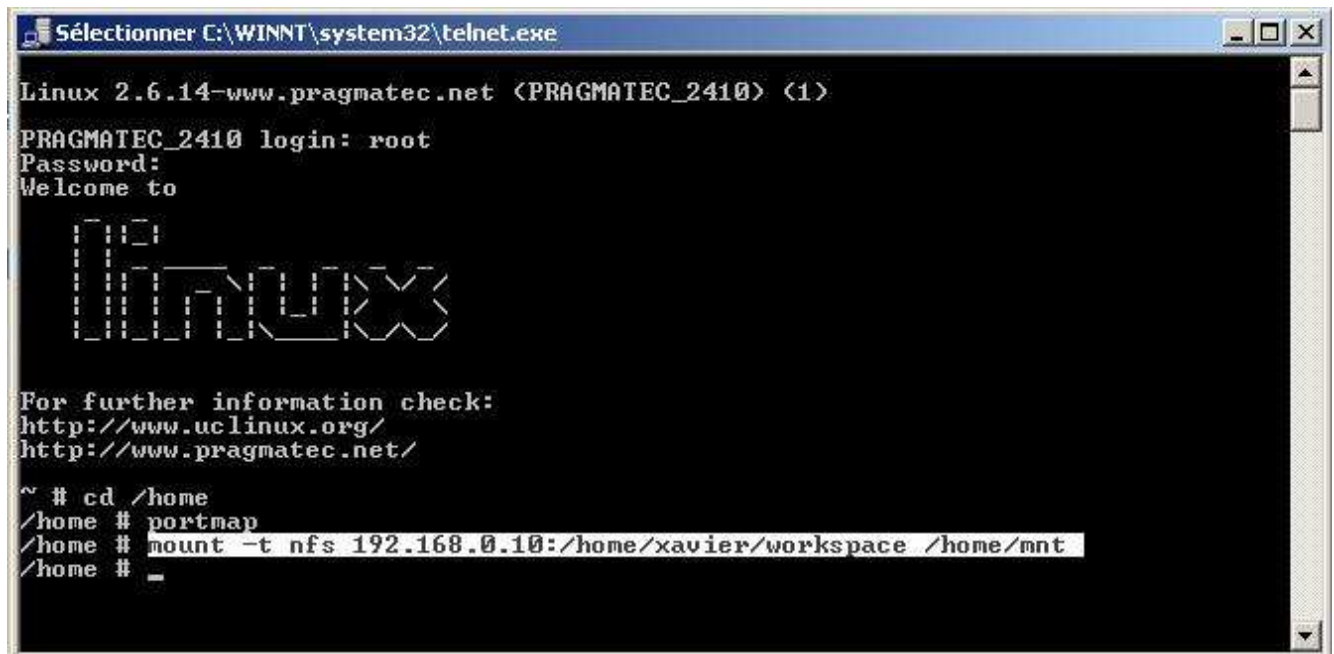
Sur la station, passez « root » et créez ou modifiez le fichier /etc/exports. Ajoutez :

```
/tftpboot 192.168.0.30(rw,sync)
```

Au prompt du shell tapez :

```
/etc/init.d/nfs restart
```

.. autant de fois que nécessaire afin d'avoir la totalité des services « OK ».



```
Sélectionner C:\WINNT\system32\telnet.exe
Linux 2.6.14-www.pragmatec.net <PRAGMATEC_2410> <1>
PRAGMATEC_2410 login: root
Password:
Welcome to

  Linux

For further information check:
http://www.uclinux.org/
http://www.pragmatec.net/

~ # cd /home
/home # portmap
/home # mount -t nfs 192.168.0.10:/home/xavier/workspace /home/mnt
/home # _
```

Linux2.6 - Mise en oeuvre

Afficher une image sur le LCD

Nous avons précédemment abordé l'utilisation du serveur graphique Nano-X. Pour afficher une image sur le LCD sans coder d'application, il est préférable d'utiliser « nxview » qui est l'un de utilitaire de démonstration fourni avec Nano-X.

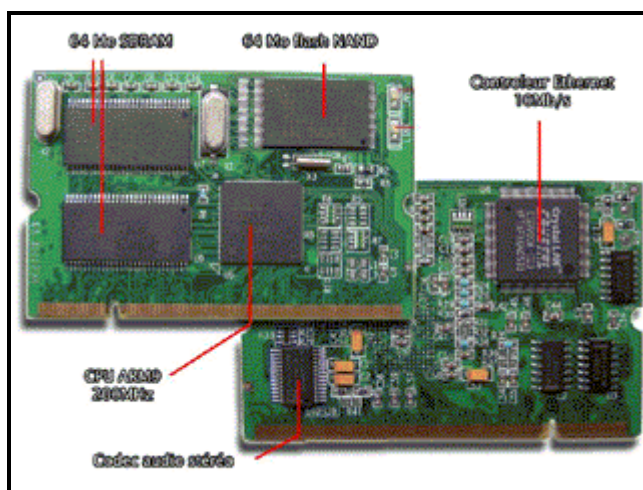
Commencez par lancer le serveur graphique en tâche de fond (avec l'option « -p » pour qu'il reste toujours actif malgré la déconnexion d'application cliente) puis l'application « nxview » :

```
cd /home/NANOX
```

```
./nano-X -p &
```

```
./nxview SODIMM.gif &
```

Vous verrez alors s'afficher l'image SODIMM.gif en 320 x 240 sur le LCD :



Enfin, nous rappelons que Nano-X est basée sur un concept client/serveur, avec un serveur graphique et la possibilité d'y connecter plusieurs clients graphiques.

Vous pouvez donc en parallèle lancer la montre « nxclock » qui s'affichera par dessus l'image SODIMM.gif :

```
./nxclock &
```

Lorsque vous supprimerez les processus nxclock et nxview de la liste des processus (commande kill), le serveur nano-X demeurera en mémoire grâce à l'option « -p » que nous avons spécifié lors de son lancement. Pour que le serveur se termine lorsque le client se déconnecte, retirer l'option « -p ».

Linux2.6 - Mise en oeuvre

Nano-X sans LCD

Dans les chapitres précédents nous avons abordé l'exemple DEMO, tous 2 basés sur l'utilisation d'un LCD graphique 240x320 pixels. Il manipule directement le frame buffer, mais il n'est pas très pratique de faire de la sorte pour dessiner sur le LCD. Le mieux est encore d'utiliser un programme qui servira d'interface entre l'espace utilisateur et le frame buffer.

Le principe est d'utiliser un serveur graphique appelé « nano-X » qui contrôle l'affichage des éléments graphiques sur le LCD et une application cliente qui dialogue avec le serveur graphique afin de lui indiquer les éléments à afficher ou rafraîchir.

Nano-X possède des fonctionnalités réseaux de sorte qu'il est possible d'avoir un client et un serveur graphique sur 2 machines Linux différentes. Il est donc possible de lancer le serveur graphique sur la station de développement et le client sur la cible.

Dans le répertoire nano-X de votre distribution (par exemple « /home/xavier/PRG_2410/user/microwindows-0.90-nxd ») nous avons ajouté le programme « **linux-bin/nano-X-i86** » qui est compilé pour un processeur i86. Vous pouvez donc le lancer sur votre station de développement. Auparavant il faut lui indiquer qu'il va devoir recevoir des consignes d'un client distant à l'aide de la variable d'environnement « REMOTE ». Tapez donc « **export REMOTE=1** ».

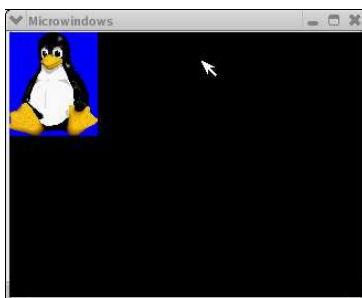
De plus si vous utilisez les font TRUETYPE avec votre projet il faut copier la font choisie dans votre homedirectory. Par exemple dans le cadre du projet DEMO_IHM qui utilise la font « arialb.ttf » vous devez copier ce fichier situé sous « « /home/xavier/PRG_2410/user/microwindows-0.90-nxd /src/fonts/truetype/ » dans le répertoire pointé par la variable FREETYPE_FONT_DIR du fichier « src/config ».

Sur la cible vous devez faire de même pour préciser au client qu'il va devoir envoyer ces consignes via le réseau. Tapez :

- **export REMOTE=1**
- **export NXDISPLAY=192.168.0.10**

Ensuite lancez votre programme client sur la cible.

Par exemple vous trouverez dans le répertoire « arm-bin » de nano-X (par exemple « /home/xavier/PRG_2410/user/microwindows-0.90-nxd/arm-bin ») le programme « nxview » qui permet d'afficher une image sur le LCD ainsi que l'image « tux.gif ».

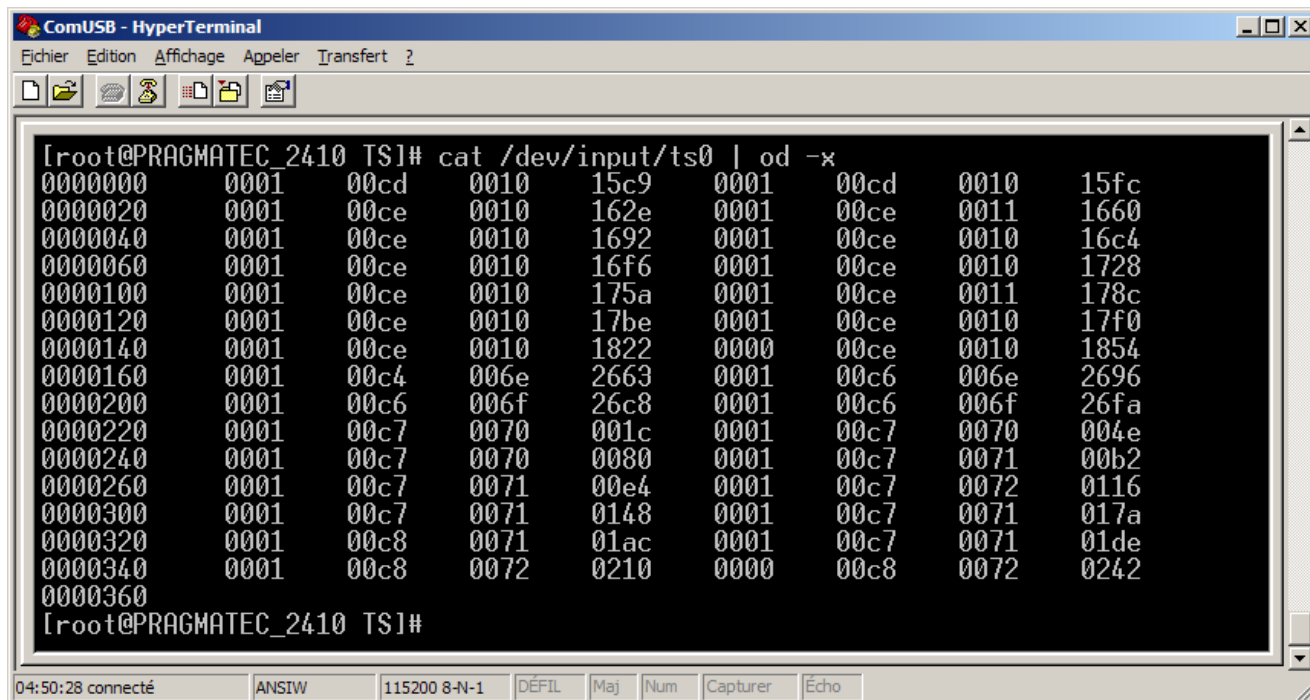


Transférez les sur la cible à l'aide de « FTP » et lancez le programme : **nxview tux.gif**

Linux2.6 - Mise en oeuvre

Utilisation de l'écran tactile

La gestion de l'écran tactile se fait par la lecture du périphérique /dev/input/ts0 :



```

ComUSB - HyperTerminal
Fichier Edition Affichage Appeler Transfert ?
[root@PRAGMATEC_2410 TS]# cat /dev/input/ts0 | od -x
0000000 0001 00cd 0010 15c9 0001 00cd 0010 15fc
0000020 0001 00ce 0010 162e 0001 00ce 0011 1660
0000040 0001 00ce 0010 1692 0001 00ce 0010 16c4
0000060 0001 00ce 0010 16f6 0001 00ce 0010 1728
0000100 0001 00ce 0010 175a 0001 00ce 0011 178c
0000120 0001 00ce 0010 17be 0001 00ce 0010 17f0
0000140 0001 00ce 0010 1822 0000 00ce 0010 1854
0000160 0001 00c4 006e 2663 0001 00c6 006e 2696
0000200 0001 00c6 006f 26c8 0001 00c6 006f 26fa
0000220 0001 00c7 0070 001c 0001 00c7 0070 004e
0000240 0001 00c7 0070 0080 0001 00c7 0071 00b2
0000260 0001 00c7 0071 00e4 0001 00c7 0072 0116
0000300 0001 00c7 0071 0148 0001 00c7 0071 017a
0000320 0001 00c8 0071 01ac 0001 00c7 0071 01de
0000340 0001 00c8 0072 0210 0000 00c8 0072 0242
0000360
[root@PRAGMATEC_2410 TS]#
04:50:28 connecté ANSIW 115200 8-N-1 DÉFIL Maj Num Capturer Écho

```

Sous le répertoire /home/TS, vous trouverez une application TS_test qui utilise cette interface pour afficher sur le terminal les coordonnées du *curseur* sur le LCD. Le code source de l'application se trouve aussi dans ce répertoire sous le nom « TS_test.c ».

Interfaçage de composant I2C

Sous Linux 2.6, le bus I2C est géré en 2 parties distinctes : le gestionnaire du bus lui-même et un ensemble de sous-driver venant s'enregistrer auprès de la couche de gestion du bus. Chaque sous-driver permet de piloter des périphériques spécifiques. Sous le répertoire /home/I2C vous trouverez 2 sous-drivers à charger à l'aide de la commande « insmod », permettant respectivement de gérer un IO Expander (PCF8574) et un convertisseur analogique numérique (PCF8591).

Nous vous conseillons de monter le « sysfs » au préalable afin de piloter directement les composants depuis l'interface /sys et ceci sans coder la moindre ligne de code (fort pratique lors des premiers essais) :

```
mount -t sysfs none /sys
```

```
cat /sys/bus/i2c/devices/0-0048/in0_input
```

Sous /sys vous trouverez les points d'entrée pour chacun des périphériques détectés. Utilisez les commandes « cat » et « echo » pour lire et écrire sur le périphérique sélectionné.

Linux2.6 - Mise en oeuvre

Montage d'une clef USB

L'USB host v1.1 est géré entièrement par le noyau sans nécessiter le chargement de drivers additionnels.



Insérez donc votre clef USB dans le slot inférieur de la prise USB host :

Vous obtenez le message suivant sur la COM1 de la carte (que vous pouvez visualiser avec HyperTerminal ou Minicom), vous indiquant qu'un media de type « USB mass storage » a été détecté :

```

comusb - HyperTerminal
Fichier Edition Affichage Appel Transfert ?
[Icons]

For further information check:
http://www.uclinux.org/
http://www.pragmatec.net/

Please press Enter to activate this console.
gethostbyname:: Resource temporarily unavailable
[root@PRAGMATEC_2410 /]# usb 1-1: new full speed USB device using s3c2410-ohci a
nd address 2
scsi0 : SCSI emulation for USB Mass Storage devices
  Vendor: USB      Model: FLASH DISK      Rev: 0437
  Type:   Direct-Access          ANSI SCSI revision: 02
SCSI device sda: 128000 512-byte hdwr sectors (66 MB)
sda: Write Protect is off
sda: assuming drive cache: write through
SCSI device sda: 128000 512-byte hdwr sectors (66 MB)
sda: Write Protect is off
sda: assuming drive cache: write through
/dev/scsi/host0/bus0/target0/lun0:<7>usb-storage: queuecommand called
p1
Attached scsi removable disk sda at scsi0, channel 0, id 0, lun 0
Attached scsi generic sg0 at scsi0, channel 0, id 0, lun 0, type 0

[root@PRAGMATEC_2410 /]# _
02:43:03 connecté  Détection auto  115200 8-N-1  DÉFIL  Maj  Num  Capturer  Écho

```

Procédez au montage comme suit :

```
mount -t vfat /dev/scsi/host0/bus0/target0/lun0/part1 /home/mnt
```

(cas où « /home/mnt » est votre répertoire de montage, et la clef USB possède une partition FAT)

Linux2.6 - Mise en oeuvre

Montage d'un disque IDE

Le driver IDE pour disque dur ou CDROM est déjà intégré au driver. Pour alimenter votre équipement IDE, utilisez les connecteurs d'extensions de la carte qui offrent des accès au +5V et à la masse.



Le disque IDE doit être insérer avant la mise sous tension de la carte. Ici nous utilisons un DiskOnModule de PQI alimenté sous 3.3V.

Pour connaître les points de connexion au +5V et à la masse, consulter le schéma électrique fourni sur votre CDROM.

La détection du disque IDE est faite durant le lancement du noyau (COM1) :

```

comusb - HyperTerminal
Fichier Edition Affichage Appel Transfert ?
[Icons]
eth0: CS8900A rev E at 0xe0000300 irq=52, no eeprom , addr: 08: 0:3E:26:0A:5B
Linux video capture interface: v1.00
ovcamchip: v2.27 for Linux 2.6 : 0V camera chip I2C driver
Uniform Multi-Platform E-IDE driver Revision: 7.00alpha2
ide: Assuming 50MHz system bus speed for PIO modes; override with idebus=xx
IDE driver for S3C2410, (c) 2006 Pragmatec
hda: PQI IDE DiskOnModule, ATA DISK drive
ide0 at 0xc4912020-0xc4912027,0xc491201c on irq 63
hda: max request size: 128KiB
hda: 64000 sectors (32 MB) w/1KiB Cache, CHS=500/8/16
hda: cache flushes not supported
/dev/ide/host0/bus0/target0/lun0: p1 p2
NFTL driver: nftlcore.c $Revision: 1.97 $, nftlmount.c $Revision: 1.40 $
S3C24XX NAND Driver, (c) 2004 Simtec Electronics
s3c2410-nand: mapped registers at c4b00000
s3c2410-nand: timing: Tacls 10ns, Twrph0 40ns, Twrph1 10ns
NAND device: Manufacturer ID: 0xec, Chip ID: 0x76 (Samsung NAND 64MiB 3,3V 8-bit
)
Scanning device for bad blocks
Creating 6 MTD partitions on "NAND 64MiB 3,3V 8-bit":
0x00000000-0x00040000 : "boot"
0x00020000-0x00320000 : "kernel"
0x00320000-0x00820000 : "rootfs"
0x00820000-0x009
02:57:46 connecté Détection auto 115200 8-N-1 DÉFIL Maj Num Capturer Écho

```

Procédez au montage comme suit :

```
mount -t ext3 /dev/ide/host0/bus0/target0/lun0/part1 /home/mnt
```

(cas où « /home/mnt » est votre répertoire de montage, et la disque possède une partition ext3)

Linux2.6 - Mise en oeuvre

Mise à jour du noyau depuis Linux

La version de Linux embarquée sur votre carte permet d'accéder à n'importe quelle partition YAFFS de votre flash NAND. Vous pouvez donc demander au noyau de mettre à jour la partition qui correspond au noyau lui-même et ceci depuis le shell.

Commencez par transférer votre image « zImage » sous /var et procédez comme suit :

```
cp /var/zImage /dev/tmdbContext1
```

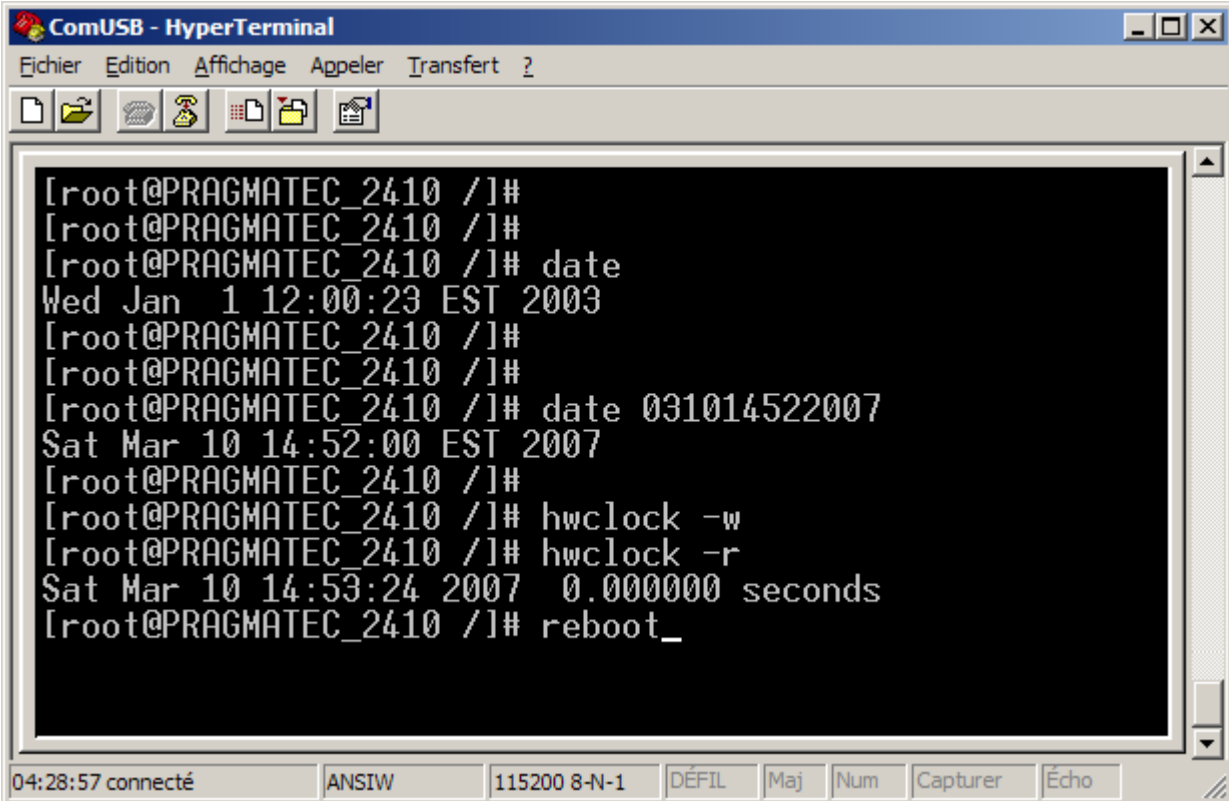
Au redémarrage de votre carte le nouveau noyau sera exécuté.

Mise à l'heure de la carte

Pour connaître à tout instant la date système il vous faut utiliser la commande « date », de même pour modifier cette date système.

```
date 031014522007 => signifie la date du 10 Mars 2007 à 14H52
```

Toutefois les valeurs entrées ou affichées ne seront pas celles issues de la RTC du processeur. Pour pouvoir synchroniser l'horloge temps réelle avec la date système de la carte, il faut utiliser la commande « hwclock » comme suit :



```
ComUSB - HyperTerminal
Fichier Edition Affichage Appeler Transfert ?
[root@PRAGMATEC_2410 /]#
[root@PRAGMATEC_2410 /]#
[root@PRAGMATEC_2410 /]# date
Wed Jan 1 12:00:23 EST 2003
[root@PRAGMATEC_2410 /]#
[root@PRAGMATEC_2410 /]#
[root@PRAGMATEC_2410 /]# date 031014522007
Sat Mar 10 14:52:00 EST 2007
[root@PRAGMATEC_2410 /]#
[root@PRAGMATEC_2410 /]# hwclock -w
[root@PRAGMATEC_2410 /]# hwclock -r
Sat Mar 10 14:53:24 2007 0.000000 seconds
[root@PRAGMATEC_2410 /]# reboot_
04:28:57 connecté ANSIW 115200 8-N-1 DÉFIL Maj Num Capturer Écho
```