# Installation et Utilisation : CIBLE 2440



## Sommaire

1.	I	ntroduction
2.	۵	Dialogue avec la cible 2440 4
1		Par Port Série4
2	2.	Par TFTP
3	8.	Par FTP14
4	ŀ.	par NFS
3.	C	Cross-Compilateur (Arm-linux)16
5	5.	Installation de Arm-linux
4.	I	nstallation de la TsLib
1		Installation
2	2.	Installation et configuration sur la cible17
5.	C	QT Embedded Pour La Cible 244019
1		Configuration et Installation19
2	2.	QTCreator – Linux
6.	C	Créer une application pour la cible 2440 22
1		Application C22
2	2.	Application Qt (console)
3	8.	Application Qt (gui)
7.	L	iens utiles
1		Sites officiels
2	2.	Tutoriaux
3	8.	Download

## **1. Introduction**

Bien que ce tutoriel décrive toutes les étapes à réaliser pour espérer (ENFIN) développer sereinement sur la MINI2440, il faut savoir qu'il peut y avoir des situations ou des problèmes différents, dues entre autre aux configurations matériels, logiciels qui peuvent varier entre les utilisateurs. Il est impossible de lister tous les cas, il faudra alors mener votre enquête d'investigation sur internet, pour voir comment régler ce problème. Je suis passé par là.

J'ai travaillé sur la SBC2440-III (voir photo en page de garde). Je ne connais pas les autres types de cartes, et donc je ne sais pas si ce tutoriel ou une partie de ce tutorial sera utile pour d'autres types de cartes.

Premier conseil : avoir un stock de shampoing qui facilitent la repousse des cheveux, car à certains moment, vous risquez de vous arracher les cheveux, tellement le chemin est long et fastidieux.

Voici ma configuration :

- → PC Windows XP (cmd en bleu)
- → PC Linux Fedora 12 (cmd en vert) kernel : 2.6.32.16-90
- → Cible SBC2440-III (cmd en rouge (minicom, HyperTerminal) kernel : 2.6.13-armlinux2440
- → Câble série reliant le PC à la cible
- → Câble Ethernet reliant le PC à la cible

Premièrement, il faut se connecter à la Cible, il y a différentes méthodes de connexion, j'en montre quelques unes (il y a également la connexion par USB, par JTAG). Ces connexions permettent de réinstaller le noyau, réinstaller le file-système de base, envoyer des fichiers (exe, libQt, tslib...) utiliser la console de la cible (minicom, HyperTerminal).

Ensuite nous allons configurer le PC Linux ainsi que la Cible2440 pour pouvoir développer des applications C, C++, QT pour la cible. Le pc compile les applications pour processeurs X86. Hors sur la cible nous disposons d'un ARM. Les applications compilées ainsi ne fonctionneront pas. C'est pourquoi, il faut installer un Cross-Compilateur (X86-Arm) sur le PC.

La TsLib est une librairie permettant d'utiliser le TouchScreen comme une souris. Il faut cross-Compiler cette TsLib, intégrer et configurer ces fichiers sur la cible.

Une fois ces étapes effectuer, on peut maintenant exécuter des applications C et C++ sur la cible, mais toujours pas des applications QT. Il faut d'abord cross-compiler QT et ensuite compiler les application QT avec le Qmake(cross-compilé pour ARM) et insérer les librairies sur la cible.

Enfin je vous montre comment <u>exécuter un Programme C, et Qt</u> sur la cible.

En <u>annexe</u>, les liens des tutos et aides qui m'ont permis d'avancer. Comme je me suis beaucoup inspiré de leurs tutos, c'est normal qu'en retour je leur fasse de la pub.

## 2. Dialogue avec la cible 2440

## 1. Par Port Série

### a. HyperTerminal (Windows)

→ Lancer l'HyperTerminal et choisir un nom : ConnectionCible2440 et taper OK

Description de la connexion					
Nouvelle connexion					
Entrez un nom et choisissez une icône pour la connexion :					
<u>N</u> om :					
ConnectionCible2440					
<u>l</u> cône :					
🙈 🤹 🧐 🧐 🦉					
OK Annuler					

 $\rightarrow$  Se connecter en tant que : COMx (x = n°port de connexion avec la cible)

➔ Propriété de COM

Bits par seconde : 115200, Bits de données : 8, Parité : aucun, Bit d'arrêt : 1, Flux : aucun

Paramètres du port		
Bit <u>s</u> par seconde :	115200	V
Bits de <u>d</u> onnées :	8	~
Parité :	Aucun	
<u>B</u> its d'arrêt :	1	
<u>C</u> ontrôle de flux :	Aucun	
L	Pa <u>r</u> amètres par de	éfaut

- ➔ Enregistrer la configuration : Fichier/Enregistrer sous
- → Pour tester, il suffit d'exécuter : ConnectionCible2440.ht et d'allumer la cible 2440.
- → Taper sur ENTREE quand il a fini de charger la cible pour faire apparaitre la console

ConnectionCible2440 - HyperTerminal \_ 🗆 🛛 Fichier Edition Affichage Appeler Transfert ? 06 🐨 🕈 👘 🖆 U-Boot 2008.10 (Jun 18 2009 - 17:34:23) DRAM: 64 MB Flash: 1 MB NAND: 128 MiB serial In: Out: serial Err: serial Hit any key to stop autoboot: 0 NAND read: device 0 offset 0x60000, size 0x200000 2097152 bytes read: OK ## Booting kernel from Legacy Image at 31000000 ... Image Name: linux-2.6.13 Created: 2009-06-19 9:37:46 UTC Image Type: ARM Linux Kernel Image (uncompressed) Data Size: 1504204 Bytes = 1.4 MB Load Address: 30008000 30008000 Entry Point: Verifying Checksum ... DÉFIL Maj Num Capturer 00:00:29 connecté VT100J 115200 8-N-1

5

### b. Minicom (Linux)

→ Si minicom n'existe pas, vous devez l'installer avec le gestionnaire de package de votre distribution Linux.



→ Aller dans Serial port setup et configurer comme suit Serial Device : /dev/ttySx (x=n°Port), Bps : 115200 8N1, Hardware... : No, Software... : No

🕞 🕂 🦳 🖻 be@fedoraBE~/Documents	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal <u>H</u> elp	
A - Serial Device : /dev/ttyS0   B - Lockfile Location : /var/lock   C - Callin Program :   D - Callout Program :	
E - Bps/Par/Bits : 115200 8N1   F - Hardware Flow Control : No   G - Software Flow Control : No   Change which setting?	
+   Screen and keyboard     Save setup as dfl     Save setup as     Exit     Exit from Minicom   +	

→ Taper Entrer, Aller dans Save setup as dfl, et dans Exit

→ Pour tester, il suffit de taper : minicom, et d'allumer la cible.

```
→ Taper sur ENTREE quand il a fini de charger la cible pour faire apparaitre la console
```

🗢 🕂 🔚 be@fedoraBE:/home/be/Documents 🔤	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal <u>H</u> elp	
U Poot 2009 10 (Jup 19 2000 17, 24, 22)	
DRAM: 64 MB Flash: 1 MB NAND: 128 MiB In: serial ur: serial	
<pre>Hit any key to stop autoboot: 0 NAND read: device 0 offset 0x60000, size 0x200000 2097152 bytes read: 0K ## Booting kernel from Legacy Image at 31000000 Image Name: Linux-2.6.13 Created: 2009-06-19 9:37:46 UTC Image Type: ARM Linux Kernel Image (uncompressed) Data Size: 1504204 Bytes = 1.4 MB Load Address: 30008000 Entry Point: 30008000 Verifying Checksum 0K</pre>	
Loading Kernel Image	<u> </u>

## 2. Par TFTP

### a. Configuration de l'adresse IP du PC Windows

- → Nouvelle adresse IP : **192.192.192.105**
- → Désactiver le pare-feu Windows

es paramètres IP peuvent être dé iseau le permet. Sinon, vous deve ppropriés à votre administrateur ré	terminés automatiquement si votre ez demander les paramètres IP éseau.
O Obtenir une adresse IP autor	natiquement
Adresse IP :	192 . 192 . 192 . 105
Masque de <u>s</u> ous-réseau :	255 . 255 . 255 . 0
Passerelle par <u>d</u> éfaut :	<u> </u>
O Obtenir les adresses des serv	eurs DNS automatiquement
💿 Utiliser l'adresse de serveur D	NS suivante :
Serveur DNS préféré :	
Serve <u>u</u> r DNS auxiliaire :	<u></u>
	Avancé

### b. Installation et configuration de TFTP

- → Installer avec les options par défaut « tftpt\_std\_personal.exe »
- ➔ Créer un dossier TFTPBOOT et y
- → copier le kernel (ulmage\_480272\_ts) dans le dossier, le renommer en ulmage
- → copier le fileSystem (nogui.yaffs2) dans le dossier, le renommer en filesystem.yaffs
- → Exécuter « TFTP TURBO »
- → File/New/Server
- → Dans Name taper : localhost et cliquer sur OK

New Server	? 🗙
Server name or address	<u>0</u> K
localhost	
🗖 Alternate port 69 🚖	<u>C</u> ancel

#### ➔ Double-Cliquer ensuite sur « localhost », une fenêtre « Login » apparait. Taper sur ENTREE

🛏 Login 🤶 🔀					
User: Admin	Login				
Password:	<u>C</u> ancel				

TFTP Turbo on localhosi	£6			
Edit View Loois Help			1 - 1	
2 4 X N		00	<b>G</b>	N?
vers ocalhost I	Server Description Platform W	CBENOV09 /indows NT 5.1		
	TFTP Turbo			
	TFTP Turbo	4.2		
	- TFTP Turbo-	4.2		
	TFTP Turbo	4.2 2 For non-commerc	sial use only.	

→ Ouvrir Edit/Properties

→ Indiquer le chemin complet du répertoire TFTPBOOT dans Virtual root et cliquer sur OK

Server Properties
General Security Option Extensions Logging
Virtual root: and Settings\agoupil\Bureau\Cible Linux Arm\TftpBoot Build
Automatically create virtual root folders
Operating mode
<ul> <li>Standard</li> </ul>
C NAT compatible This is the recom
Error handling
Timeout: 3 seconds
Retransmits: 5 🝨
<u>O</u> K <u>C</u> ancel

 Ouvrir Tool/Control Service, et vérifier que TFTP Turbo is running, sinon cliquez sur START

👫 Control Service	? 🔀
TFTP Turbo is running on localhost	Stop
	ОК

#### c. HyperTerminal

- → Lancer l'HyperTerminal avec la configuration « ConnectionCible2440.ht »
- ➔ Allumer la Cibler
- ➔ Dès le démarrage de la cible sur l'HyperTerminal, appuyez sur une touche pour aller dans la configuration u-boot



→ Si vous ne voyez pas « mini2440 # » et que vous ne pouvez pas taper des commandes, rebooter et refaire l'opération.

### Installation du Kernel

→ Taper « run install-kernel »	
🍣 ConnectionCible2440 - HyperTerminal 📃 🗖	
Eichier Edition Affichage Appeler Iransfert ?	
In: serial Out: serial Err: serial Hit any key to stop autoboot: 0 mini2440 # run install-kernel dm9000 i/o: 0x18000300, id: 0x90000a46 DM9000: running in 16 bit mode MAC: 08:00:3e:26:0a:5b TFTP from server 192.192.192.105; our IP address is 192.192.192.200 Filename 'uImage'. Load address: 0x30000000 Loading: I ###################################	
UU:27:07 connecte Detec, auto 115200 8-N-1 Dente mai Num Capacitar Cono	

→ Si tout se passe bien vous devriez voir Writing ... 100% Complete.

## Installation du File System

Taper maintenant « run instan-mesystem »	
🌯 ConnectionCible 2440 - Hyper Terminal	
Eichier Edition Affichage Appeler Iransfert 2	
done Bytes transferred = 29688384 (1c50240 hex)	
NAND erase: device 0 offset 0x260000, size 0x1c50240 Erasing at 0x1ea0000 100% complete. OK	
NAND write: device 0 offset 0x260000, size 0x1c50240 Writing at 0x1dc0000 100% is complete. 28788736 bytes written: OK mini2440 #	
00:31:17 connecté Détec, auto 115200 8-N-1 DÉFIL Maj Num Capturer Écho	

→ Taper maintenant « run install-filesystem »

- → Si tout se passe bien vous devriez voir Writing ... 100% Complete.
- → Taper « **Reset** » pour rebooter la cible

## 3. Par FTP

### a. Adresse IP du PC

- → Sous Linux : Taper ifconfig eth0 192.168.1.15
- → Sous Windows : Nouvelle adresse IP : **192.168.1.15**
- → Sur la cible 2440, taper : chmod 777 /var/ftp/pub

is paramètres in peuvent être de seau le permet. Sinon, vous dev ipropriés à votre administrateur re	èterminés automatiquement si votre ez demander les paramètres IP éseau.
O <u>O</u> btenir une adresse IP auton	natiquement
Utiliser l'adresse IP suivante :	
Adresse <u>I</u> P :	192 . 168 . 1 . 15
Masque de <u>s</u> ous-réseau :	255 . 255 . 255 . 0
Passerelle par <u>d</u> éfaut :	94 30 40
) Obtenir les adresses des serv	eurs DNS automatiquement
Utiliser l'adresse de serveur D	NS suivante :
Serveur DNS pré <u>f</u> éré :	· · · · ·
Serveur DNS auxiliaire :	

### **b.** Client FTP

#### **Firefox**

→ Dans la barre d'adresse taper : ftp://192.168.1.168/

Index de ftp://192.168.1.168/	
🎦 Vers un rép. de plus haut niveau	
Nom	Taille Dernière modification
Pub Contraction of the contracti	16/08/2007 00:00:00

#### FileZilla

- ➔ Cliquer sur Nouveau Site
- → Taper Cible Linux
- → Dans Hôte, taper 192.168.1.168
- → Cliquer sur Connexion

Gestionnaire de Sites	X
Sélectionnez une entrée :	Général       Avancé       Transfer Settings       Jeu de caractères         Hôte :       192.168.1.168       Port :
	Logon Type:       Anonyme         Identifiant :       anonymous         Mot de passe :       •••••••         Compte :       •••••••
Nouveau Site     Nouveau Dossier       Nouveau Fayori     Renommer       Supprimer     Copier	Comment <u>a</u> ires :

### 4. par NFS

#### a. Démarrer NFS server et configurer l'IP

- → Taper dans la console Linux en mode root : /etc/init.d/nfs start
- → Taper : ifconfig eth0 192.168.1.15

#### b. Copier et décompresser s3c2440\_recover.tar.gz

- → Dans la console Linux en mode root taper : mkdir /armLinux
- → Aller dans le dossier ou se trouve le fichier s3c2440\_recover.tar.bz2
- → Taper : tar -xjvf s3c2440\_recover.tar.bz2 -C /armLinux

#### c. Recharger le système graphique par défaut

- → Exécuter minicom
- → Pendant le boot, appuyer sur une touche pour aller dans la configuration u-boot
- → Taper :

setenv bootargs init=linuxrc console=ttySAC0 root=/dev/nfs

nfsroot=192.168.1.15:/armLinux/s3c2440\_recover

ip=192.168.1.168:192.168.1.1:192.168.1.1:255.255.255.0:www.embedinfo.com :eth0 :off

- ➔ Taper ensuite boot
- → Il est possible que l'adresse lp soit supprimé (ifconfig pour vérifier), si c'est le cas, retaper : ifconfig eth0 192.168.1.15
- → Attendre que la console demande de taper « Entree ».
- ➔ Taper : recover\_system\_no\_gui
- → Attendre que la console ait terminé son installation
- → Dès qu'il demande de rebooter, taper reboot

## 3. Cross-Compilateur (Arm-linux)

### 5. Installation de Arm-linux

- → Télécharger le compilateur croisé ARM-Linux-GCC 3.4.1
- ➔ Taper : tar zxvf arm-linux-gcc-3.4.1.tgz
- ➔ Taper : mkdir –p /usr/local/arm
- → Taper : mv usr/local/arm/3.4.1/ /usr/local/arm/

## 4. Installation de la TsLib

## 1. Installation

- → Télécharger TsLib
- → Taper dans la console linux : tar xvzf tslib.tar.gz
- → Taper : cd tslib
- → Taper : ./autogen.sh
- ➔ Taper : sudo kate configure &
- → Chercher (CTRI+F) : rpl\_malloc
- ➔ Remonter jusqu'à voir :

{ \$as\_echo " \$as\_me :\$LINENO : checking for GNU libc compatible malloc " >&5
\$as\_echo\_n "checking for GNU libc compatible malloc... " > &6 ; }
if test "\${ac\_cv\_func\_malloc\_0\_nonnull+set} " = set ; then
 \$as\_echo\_n "(cached) " > &6
else
 if test "\$cross\_compiling" = yes ; then
 ac\_cv\_func\_malloc\_0\_nonnull=no

- Remplacer no par yes sur la ligne : ac\_cv\_func\_malloc\_0\_nonnull=no
- → Sauvegarder et Quitter Kate
- → Sur la console, Taper : CXX= /usr/local/arm/3.4.1/bin/arm-linux -g++ CC= /usr/local/arm/3.4.1/bin/arm-linux-gcc ./configure --host=arm-linux --target=arm --prefix=/usr/local/tslib
- → Taper : make
- → Taper : make install
- → Copier : cp /usr/local/tslib /armLinux/s3c2440\_recover/

### 2. Installation et configuration sur la cible

- Se connecter par Minicom en utilisant NFS au dossier s3c2440\_recover (Etape 3 sans faire la commande recover\_system\_no\_gui)
- → Taper sur la console Minicom : mount -t yaffs /dev/mtdblock2 /mnt
- → Copier dans la librairie de la cible les libs Qt : cp -R / tslib/\* /mnt/
- → Taper : vi /etc/ts.conf
- Décommenter la ligne correspondant à votre entrée Ts.
   Par exemple, pour mon cas c'est H3600. Je le vois dans le dossier /dev, il y a une entrée « h3600\_tsraw »

Ts.conf devrait pour mon cas être définit comme suit :

#### module\_raw h3600

module pthres pmin=1 module variance delta=30 module dejitter delta=100 module linear → Taper : vi /etc/qt4\_env.conf

→	Editer le fichier :
	export TSLIB_TSEVENTTYPE=H3600
	export TSLIB_CONSOLEDEVICE=none
	export TSLIB_FBDEVICE=/dev/fb0
	export TSLIB_TSDEVICE=/dev/h3600_tsraw
	export TSLIB_CALIBFILE/etc/pointercal
	export TSLIB_CONFFILE=/etc/ts.conf
	export TSLIB_PLUGINDIR=/lib/ts
	export QWS_MOUSE_PROTO=tslib:/dev/h3600_tsraw
→	Calibrer le panel en tapant : ts calibrate

- Sur le panel, on devrait voir une séquence de calibration apparaitre
- ➔ Tester le panel en tapant : ts\_test
- → Editer le fichier profile : vi /etc/profile
- ➔ Ajouter la ligne

source /etc/qt4-env

## 5. QT Embedded Pour La Cible 2440

## 1. Configuration et Installation

### a. Téléchargement de QT Embedded (version 4.5.2)

- → Télécharger la dernière version de QT-Embedded
- → Taper : tar zxvf qt-embedded-linux-opensource-src-4.5.2.tar.gz

### b. Editer qmake.conf (Linux-arm-g++)

- → Taper : cd qt-embedded-linux-opensource-src-4.5.2
- → Taper : sudo Kate mkspecs/qws/linux-arm-g++/qmake.conf
- → Modifier ce fichier :

TOOLS_BIN = /usr/local/arm/3.4.1	/bin
#modifications to g++.conf	
QMAKE_CC	= \$\$TOOLS_BIN/arm-linux-gcc
QMAKE_CXX	= \$\$TOOLS_BIN/arm-linux-g++
QMAKE_LINK	= \$\$TOOLS_BIN/arm-linux-g++
QMAKE_LINK_SHLIB	= \$\$TOOLS_BIN/arm-linux-g++
#modifications to linux.conf	
QMAKE_AR	= \$\$TOOLS_BIN/arm-linux-ar cqs
QMAKE_OBJCOPY	= \$\$TOOLS_BIN/arm-linux-objcopy
QMAKE_STRIP	= \$\$TOOLS_BIN/arm-linux-strip

➔ Rajouter à ce fichier :

QMAKE_CFLAGS_RELEASE	~= s/-O[123s]/-O0/
QMAKE_CXXFLAGS_RELEASE	~= s/-O[123s]/-O0/

### c. Configuration et compilation

- → Taper : mkdir /usr/local/Qt
- → Taper : ./configuration -embedded arm -xplatform qws/linux-arm-g++ -prefix /usr/local/Qt -qt-mouse-tslib -L /usr/local/tslib/lib -l /usr/local/tslib/include
- → Si pas d'erreur, taper : make (étape très longue = quelques heures)
- ➔ Si pas d'erreur, taper : make install

### d. Installation sur la cible

- ➔ Privilégier l'envoi par NFS (cf <u>4.par NFS</u>)
- ➔ Créer un dossier : mkdir /armLinux/s3c2440\_recover/LibQt
- → Copier : cp -R /usr/local/Qt/lib/\* /armLinux/s3c2440\_recover/LibQt/
- → Copier : cp /ustr/local/ arm/3.4.1/arm-linux/lib/libgcc\_s.so.1
- → Se connecter par Minicom en utilisant NFS au dossier s3c2440\_recover (Etape 3 sans faire la commande recover\_system\_no\_gui)
- → Taper sur la console Minicom : mount –t yaffs /dev/mtdblock2 /mnt
- → Mettre les droits : chmod 777 /mnt/lib/libgcc\_s.so.1
- → Copier dans la librairie de la cible les libs Qt : cp –R / LibQt/\*.so.4 /mnt/lib/
- → Créer les dossier s: mkdir /usr/local/
- → mkdir /usr/local/Qt
- → mkdir /usr/local/Qt/lib
- → Déplacer le dossier fonts : mv /mnt/lib/fonts /usr/local/Qt/lib/

### 2. QTCreator – Linux

### a. Ajout du Qmake compilé pour Arm

Exécuter QT Creato	or and a second s	
Ouvrir : Tools/Optic	ons	
Ouvrir Qt4/Qt Vers	ions	
→ Appuyer sur 🔁		
😌 🕀 🌔 🥵 Options	3 🗆 🖿	×
+- Environment +- Locator	Qt versions	
	Name QMake Location	
+- C++ +- CMake	Auto-detected Qt in PATH /usr/bin/qmake-qt4 Manual	9
+ Code Pasting	4.5.2 Emb /usr/local/Qt/bin/qmake	
+ Debugger		
+ Designer		
+ Help		
- Qt4		
Qt Versions		
	Version Name: 4.5.2 Emb	
	QMake Location: /usr/local/Qt/bin/qmake Browse	
	Debugging Helper:	
	Found Qt version 4.5.2, using mkspec /usr/local/Qt/m	
	Default Ot Version: 4 5 2 Emb	0
		<u> </u>
	🛛 🔗 OK 🛛 🔗 Apply 🖉 🔗 Canc	el

- → Versions Name, taper : <versions> Emb
- → QMake Location : Aller chercher l'executable qmake, ici : /usr/local/Qt/bin/qmake
- → Cliquer sur **Rebuild**
- → Si tout se passe bien, Debbugging Helper : 🗹
- → Alors cliquer sur Apply et sur OK

Maintenant, les applications seront compilées pour ARM.

### b. Vérification que l'exécutable est compilé pour ARM

Vérification Après avoir créer et compilé (avec succès) une ApplicationX

- → Dans la console Linux : cd <Path ApplicationX>
- → Taper : file ApplicationX avec ApplicationX = le nom du fichier compilé

ApplicationX : ELF 32-bit LSB executable, ARM, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.14, not stripped

## 6. Créer une application pour la cible 2440

- → Dans la console Linux : mkdir /armLinux/s3c2440\_recover/home/Developpement
- ➔ mkdir /armLinux/s3c2440\_recover/home/Developpement/C
- → mkdir /armLinux/s3c2440\_recover/home/Developpement/Qt

## **1. Application C**

### a. Compilation

- → Dans la console Linux : cd /armLinux/s3c2440\_recover/home/Developpement/C/
- ➔ sudo kate Test1.c

```
#include <stdio .h>
int main(void)
{
    printf("Hello World !!\n ");
    return 0;
}
```

→ Compiler : /usr/local/arm/3.4.1/bin/arm-linux-gcc -o Test1 Test1.c

#### **b.** Exécution

- ➔ Privilégier l'envoi par NFS (cf <u>4.par NFS</u>)
- → Se connecter par Minicom en utilisant NFS au dossier s3c2440\_recover (Etape 3 sans faire la commande recover\_system\_no\_gui)
- → Taper sur la console Minicom : mount -t yaffs /dev/mtdblock2 /mnt
- → Copier l'exécutable : cp /homeDeveloppement/C/Test1 /mnt/home
- → Rebooter sur la cible: reboot
- → Exécuter Test1 : ./home/Test1 Si tout se passe bien, on devrait voir apparaitre : Hello World !!

PS : pour quitter le programme (s'il ne se quitte pas tout seul), Presser : CTRL+C)

## 2. Application Qt (console)

### a. Compilation

- Créer un projet console avec <u>QtCreator</u> dans /armLinux/s3c2440\_recover/home/Developpement/Qt
- ➔ Compiler l'application

#### **b.** Exécution

- → Privilégier l'envoi par NFS (cf <u>4.par NFS</u>)
- → Se connecter par Minicom en utilisant NFS au dossier s3c2440\_recover (Etape 3 sans faire la commande recover\_system\_no\_gui)
- → Taper sur la console Minicom : mount –t yaffs /dev/mtdblock2 /mnt
- → Copier l'exécutable : cp /home/Developpement/Qt/<nomApp> /mnt/home
- → Rebooter sur la cible: reboot
- Exécuter <nomApp> : ./home/ <nomApp>
   Si tout se passe bien, l'application devrait s'exécuter.

## 3. Application Qt (gui)

#### a. Compilation

- Créer un projet gui avec <u>QtCreator</u> dans /armLinux/s3c2440\_recover/home/Developpement/Qt
- ➔ Compiler l'application

#### **b.** Exécution

- ➔ Privilégier l'envoi par NFS (cf <u>4.par NFS</u>)
- Se connecter par Minicom en utilisant NFS au dossier s3c2440\_recover (Etape 3 sans faire la commande recover\_system\_no\_gui)
- → Taper sur la console Minicom : mount –t yaffs /dev/mtdblock2 /mnt
- → Copier l'exécutable : cp /home/Developpement/Qt/<nomAppGui> /mnt/home
- → Rebooter sur la cible: reboot
- ➔ Exécuter <nomAppGui> : ./home/ < nomAppGui > -qws Si tout se passe bien, l'application devrait s'exécuter.

## 7. Liens utiles

## 1. Sites officiels

- → Site officiel de la mini2440 : <u>http://www.friendlyarm.net/</u>
- → Site officiel de Qt : <u>http://qt.nokia.com/</u>
- → différentes versions de Qt-Embedded : <u>ftp://ftp.qt.nokia.com/qt/source/</u>

## 2. Tutoriaux

- → Wiki Fr très complet : <u>http://benoit.vince84.free.fr/wiki/pmwiki.php</u>
- ➔ tutorial Fr Qt-Embedded sur la mini2440 : <u>http://www.pobot.org/Qt4-6-embedded-sur-la-mini2440.html</u>
- → tutorial En Qt-Embedded sur la mini2440 :

  <u>http://blog.cor-net.org/embedded/mini2440/qt-45-on-mini2440/</u>
- ➔ Blog qui as pour ambition de regrouper toutes les infos sur la Mini2440 : <u>http://dlewin.free.fr/davblog/</u>

## 3. Download

- → S3c2440\_recover : <u>http://mini2440.agoupil.free.fr/Download/s3c2440\_recover.tar.gz</u>
- → arm-linux-gcc-3.4.1 : <u>http://mini2440.agoupil.free.fr/Download/arm-linux-gcc-3.4.1.tar.bz2</u>
- → tslib : <u>http://mini2440.agoupil.free.fr/Download/tslib.tar.gz</u>
- → qt-embedded-linux-opensource-src-4.5.2 : <u>http://mini2440.agoupil.free.fr/Download/qt-embedded-linux-opensource-src-4.5.2.tar.gz</u>