

Mini2440 の Android インストール 簡易マニュアル

株式会社日昇テクノロジー

<http://www.csun.co.jp>

info@csun.co.jp

2011/08/03



[copyright@2013](http://www.csun.co.jp)

修正履歴

NO	バージョン	修正内容	修正日
1	Ver1.0	新規作成	2010/2/23
2	Ver1.1	Android2.1にアップデート	2011/8/3

※ この文書の情報は、文書を改善するため、事前の通知なく変更されることがあります。最新版は弊社ホームページからご参照ください。

「<http://www.csun.co.jp>」

※ (株)日昇テクノロジーの書面による許可のない複製は、いかなる形態においても厳重に禁じられています。

※ Androidに関わるリソースはご参考までの実験レベルで確認済みのものですので、サポートは提供していません。

機能概要	4
第一章 クロスコンパイラーのバージョンの確認	5
第二章 カーネルとファイルシステムの解凍	11
第三章 カーネルの設定とコンパイル	12
第四章 イメージファイルシステム yaffs2 の作成	14
第五章 ターゲットボードに書き込む及び実行	16

開発環境 Fedora 9

クロスコンパイラー arm-linux-gcc-4.3.2 with EABI

ターゲット mini2440+3.5" LCD(Sony LCD3.5)

機能概要

- 1、タッチパネル LCD(Topply LCD3.5、NEC LCD3.5、Sony LCD3.5)
- 2、SD カード自動認識 (起動前に挿入する必要)
- 3、キー操作
 - K1: volume up
 - K2: middle
 - K3: volume down
 - K4: menu
 - K5: right
 - K6: return
- 4、DM9000ドライバ
- 5、busybox 1.13.3
- 6、Audio、MP3 play、録音可
- 7、snapshot

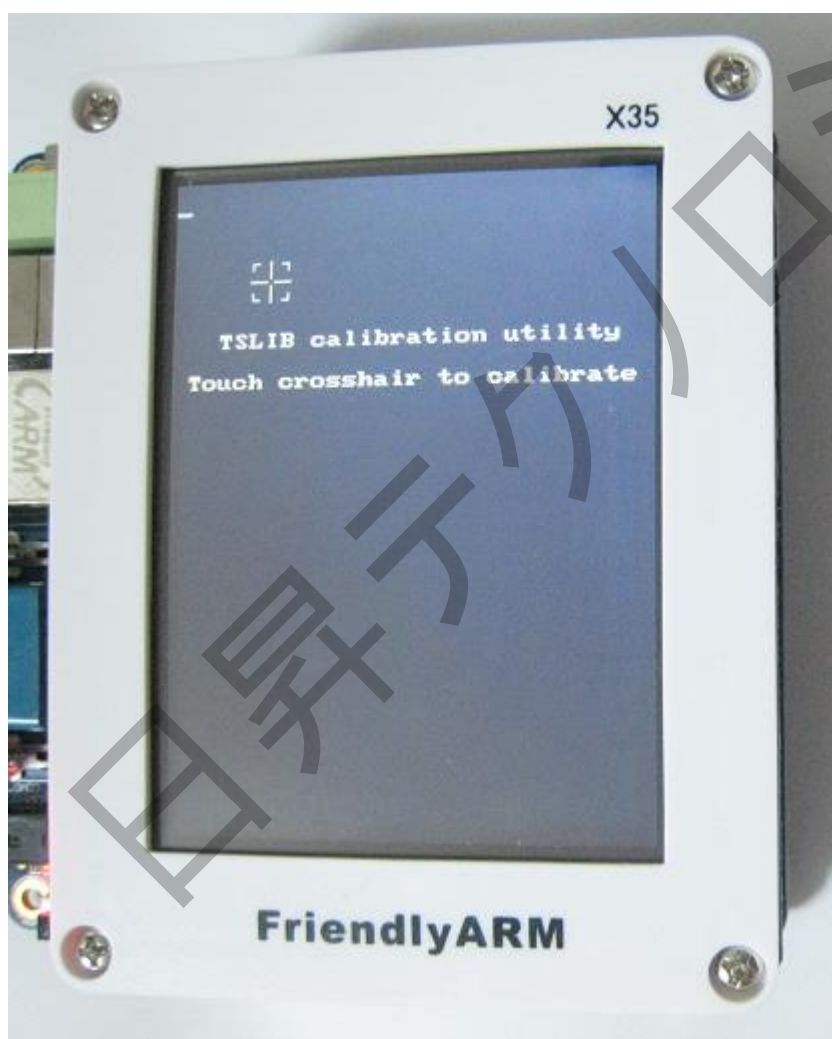
第一章 初体験

1.1 タッチパネルの校正

インストール後、初めて起動する場合、自動にタッチパネル校正画面に入ります。ペンで“十”字の中心をタッチします。四角と中心、すべて五つの“十”字が順番に出てきます。

次のコマンドで `/system/etc/shine/pointercal` というファイルを削除すれば、再起動の時、タッチパネルを再補正します。

```
# rm /system/etc/shine/pointercal
```



1.2 スリープモードにしないように設定

Android2.1 が起動した直後の様子。Android の起動時間が長い、ご了承ください。



起動後、ロックの状態になります。ロックを解除するため、以下のように操作してください。下記ロックの所にペンでクリックしてから右方向にドラッグしてください。



ロック解除



ロック解除後様子

この Android バージョンは一度スリープモードに入ると、戻れません、再起動しなければなりません。そのため、Android の液晶画面で「設定」→「アプリケーション」→「開発」→「スリープモードにしない」に設定してください。

※ボードには **64MB** メモリしかない、数多くのアプリケーションを実装すれば、とても遅いです。提供されたファイルシステムには多いアプリケーションがあるので、遅いです。ご了承ください。(1.3 節にスピードアップ方法を紹介)

次は操作の図解です。



ペンで上下、左右で移動し選択



1.3 動きをスピードアップ

1.2 節に説明した通り、ボードには **64MB** メモリしかないので、操作が遅いです。使わないアプリケーションを削除すれば、速くなります。

Android の **system/app** ディレクトリには最小限の 6 個ファイル：

LatinIME.apk

Launcher.apk

PackageInstaller.apk

Settings.apk

SettingsProvider.apk

UserDictionaryProvider.apk を残って、自分のアプリケーションだけをインストールすれば、とても速いです。

1.4 ネットワーク設定

ネットワークの設定のために、Android のコンソールで次のコマンドを入力してください。

```
# rm /system/bin/ifconfig
```

```
# rm /system/bin/route
```

```
# cd /bin
```

```
# ln -s busybox ifconfig
```

```
# ln -s busybox route
```

次のコマンドで DHCP 機能により IP アドレスを自動取得します。

```
# netcfg eth0 up
```

```
# netcfg eth0 dhcp
```

自分も IP アドレスを指定できます。

```
# ifconfig eth0 192.168.0.230
```

「192.168.0.230」は指定された IP アドレスです。

起動のとき、ネットワークを自動的に設定とすれば、「/system/etc/shine/net.conf」を編集してください。

/system/etc/shine/net.conf の中身

```
NET=wlan0 #有線 LAN なら eth0 に変更してください
```

```
IP=192.168.0.230
```

```
GW=192.168.0.1
```

```
DNS=192.168.0.1
```

Android のコンソールのコマンドの使い方はLinuxと同じです。

例：液晶画面をキャプチャするコマンド

```
# snapshot -d /dev/graphics/fb0 picture.png
```

※ 上記 Android の操作画面はこのコマンドで撮りました。

第二章 クロスコンパイラーのバージョンの確認

弊社で提供している android のカーネルをコンパイルする為には、EABI の arm-linux-gcc-4.3.2 を利用する。下記コマンドでバージョンを確認する。

```
#arm-linux-gcc -v
```



```
root@tom:~  
File Edit View Terminal Tabs Help  
[root@tom ~]# arm-linux-gcc -v  
Using built-in specs.  
Target: arm-none-linux-gnueabi  
Configured with: /scratch/julian/lite-respin/linux/src/gcc-4.3/configure --build=i686-pc-linux-gnu --host=i686-pc-linux-gnu --target=arm-none-linux-gnueabi --enable-threads --disable-libmudflap --disable-libssp --disable-libstdcxx-pch --with-gnu-as --with-gnu-ld --enable-languages=c,c++ --enable-shared --enable-symvers=gnu --enable-__cxa_atexit --with-pkgversion='Sourcery G++ Lite 2008q3-72' --with-bugurl=https://support.codesourcery.com/GNUToolchain/ --disable-nls --prefix=/opt/codesourcery --with-sysroot=/opt/codesourcery/arm-none-linux-gnueabi/libc --with-build-sysroot=/scratch/julian/lite-respin/linux/install/arm-none-linux-gnueabi/libc --with-gmp=/scratch/julian/lite-respin/linux/obj/host-libs-2008q3-72-arm-none-linux-gnueabi-i686-pc-linux-gnu/usr --with-mpfr=/scratch/julian/lite-respin/linux/obj/host-libs-2008q3-72-arm-none-linux-gnueabi-i686-pc-linux-gnu/usr --disable-libgomp --enable-poison-system-directories --with-build-time-tools=/scratch/julian/lite-respin/linux/install/arm-none-linux-gnueabi/bin --with-build-time-tools=/scratch/julian/lite-respin/linux/install/arm-none-linux-gnueabi/bin  
Thread model: posix  
gcc version 4.3.2 (Sourcery G++ Lite 2008q3-72)  
[root@tom ~]#
```

第三章 カーネルとファイルシステムの解凍

- 1、`/opt/FriendlyARM/android` のフォルダを作成する。
- 2、上記フォルダで下記コマンドを実施する。

```
#tar xvzf csun_android2.1-armv4t_kernel.tgz
```

```
#tar xvzf csun_android2.1-armv4t-rootfs.tgz
```

実施後の結果：



```
csun@fedora13:/home/csun/mini2440/android
ファイル(F) 編集(E) 表示(V) 端末(T) ヘルプ(H)
[csun@fedora13 ~]$ su
パスワード:
[root@fedora13 csun]# ls
kddi      ダウンロード デスクトップ ビデオ 画像
mini2440  テンプレート ドキュメント 音楽 公開
[root@fedora13 csun]# cd mini2440
[root@fedora13 mini2440]# ls
android  other
[root@fedora13 mini2440]# cd android
[root@fedora13 android]# ls
csun_android2.1-armv4t-rootfs      csun_android2.1-armv4t_kernel
csun_android2.1-armv4t-rootfs.tgz  csun_android2.1-armv4t_kernel.tgz
[root@fedora13 android]#
```

第三章 カーネルの設定とコンパイル

上記解凍後既に何種類の config ファイルがある。

使われる LCD により、該当の config ファイルを使ってください。

config_mini2440_X35 : Sony LCD3.5 用の config ファイル

config_mini2440_T35 : Toppo LCD3.5 用の config ファイル

config_mini2440_N35 : NEC LCD3.5 用の config ファイル

ここ Sony LCD3.5 を例として説明します。

下記コマンドを実行する。

```
#cp config_mini2440_X35 .config
#make menuconfig
```



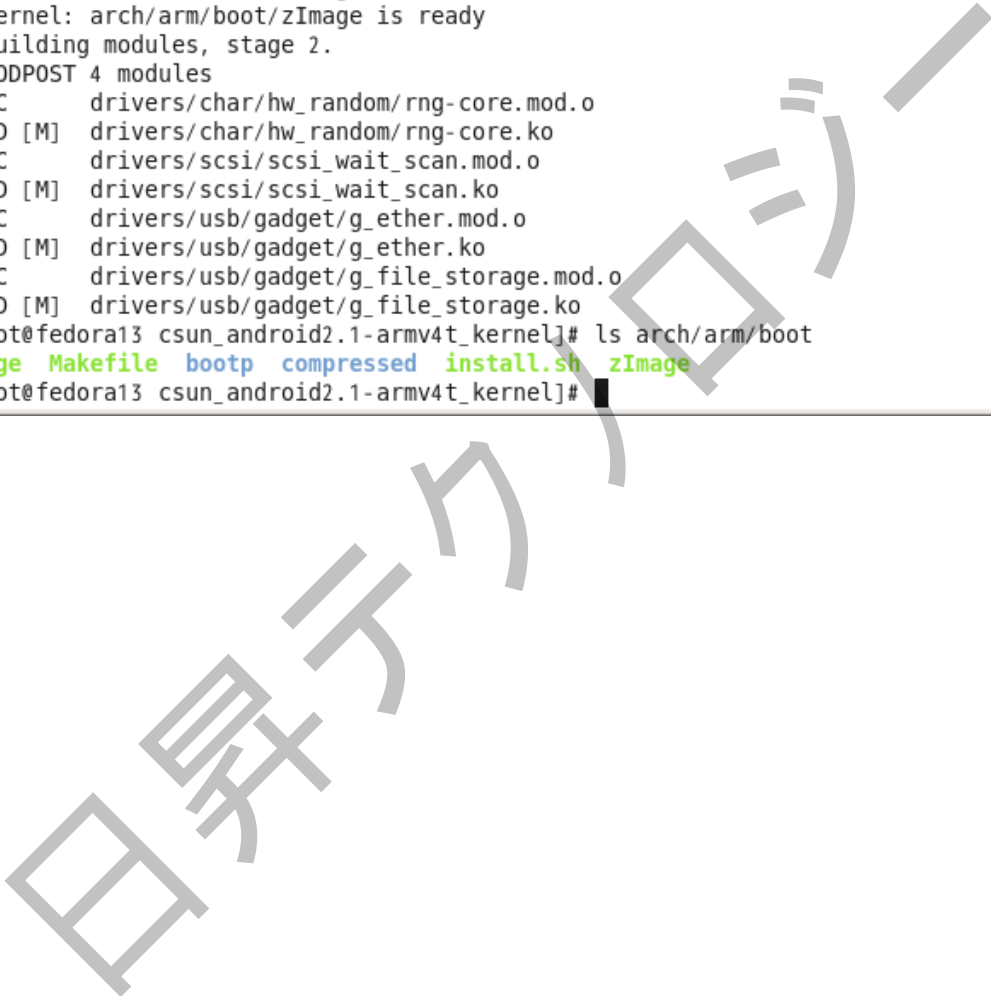
```
csun@fedora13:/home/csun/mini2440/android/csun_android2.1-armv4t_ke
ファイル(F) 編集(E) 表示(V) 端末(T) ヘルプ(H)
[root@fedora13 csun_android2.1-armv4t_kernel]# cd ..
[root@fedora13 android]# ls
csun_android2.1-armv4t-rootfs      csun_android2.1-armv4t_kernel
csun_android2.1-armv4t-rootfs.tgz  csun_android2.1-armv4t_kernel.tgz
[root@fedora13 android]# cd csun_android2.1-armv4t_kernel
[root@fedora13 csun_android2.1-armv4t_kernel]# ls
COPYING      README      crypto      ipc          security
CREDITS      REPORTING-BUGS  defConfig  kernel      sound
Documentation arch        drivers     lib          usr
Kbuild       block      firmware    mm           virt
MAINTAINERS  config_mini2440_N35  fs         net
Makefile     config_mini2440_T35  include    samples
Module.symvers config_mini2440_X35  init       scripts
[root@fedora13 csun_android2.1-armv4t_kernel]# cp config_mini2440_X35 .config
cp: '.config' を上書きしてもよろしいですか(yes/no)? y
[root@fedora13 csun_android2.1-armv4t_kernel]# make menuconfig
HOSTCC scripts/basic/fixdep
```

この状態で何も変更しないで、下記コマンドを実行する。実行後 `arch/arm/boot` フォルダに `zImage` ファイルが生成される。

```
#make zImage
```

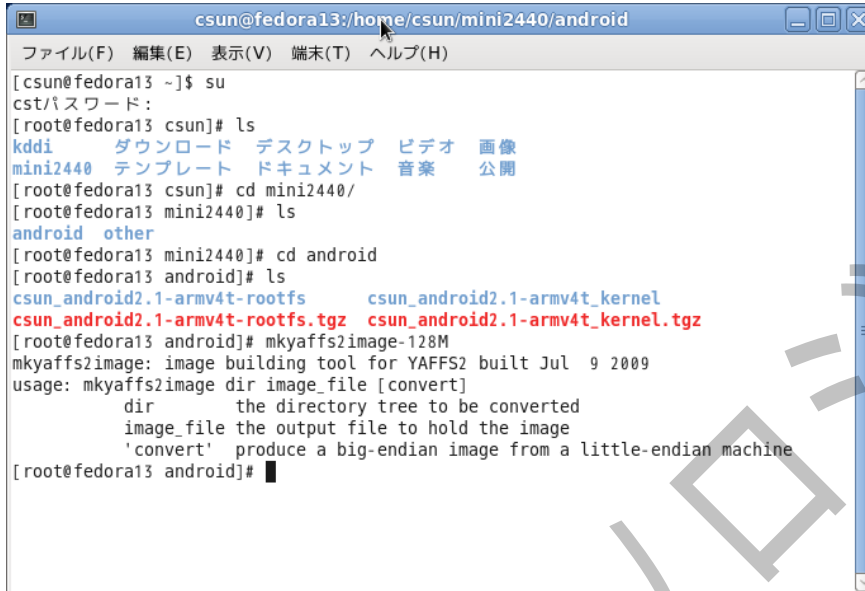


```
csun@fedora13:/home/csun/mini2440/android/csun_android2.1-armv4t_ke
ファイル(F) 編集(E) 表示(V) 端末(T) ヘルプ(H)
SYSMAP System.map
SYSMAP .tmp_System.map
OBJCOPY arch/arm/boot/Image
Kernel: arch/arm/boot/Image is ready
AS arch/arm/boot/compressed/head.o
GZIP arch/arm/boot/compressed/piggy.gz
AS arch/arm/boot/compressed/piggy.o
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 4 modules
CC drivers/char/hw_random/rng-core.mod.o
LD [M] drivers/char/hw_random/rng-core.ko
CC drivers/scsi/scsi_wait_scan.mod.o
LD [M] drivers/scsi/scsi_wait_scan.ko
CC drivers/usb/gadget/g_ether.mod.o
LD [M] drivers/usb/gadget/g_ether.ko
CC drivers/usb/gadget/g_file_storage.mod.o
LD [M] drivers/usb/gadget/g_file_storage.ko
[root@fedora13 csun_android2.1-armv4t_kernel]# ls arch/arm/boot
Image Makefile bootp compressed install.sh zImage
[root@fedora13 csun_android2.1-armv4t_kernel]#
```



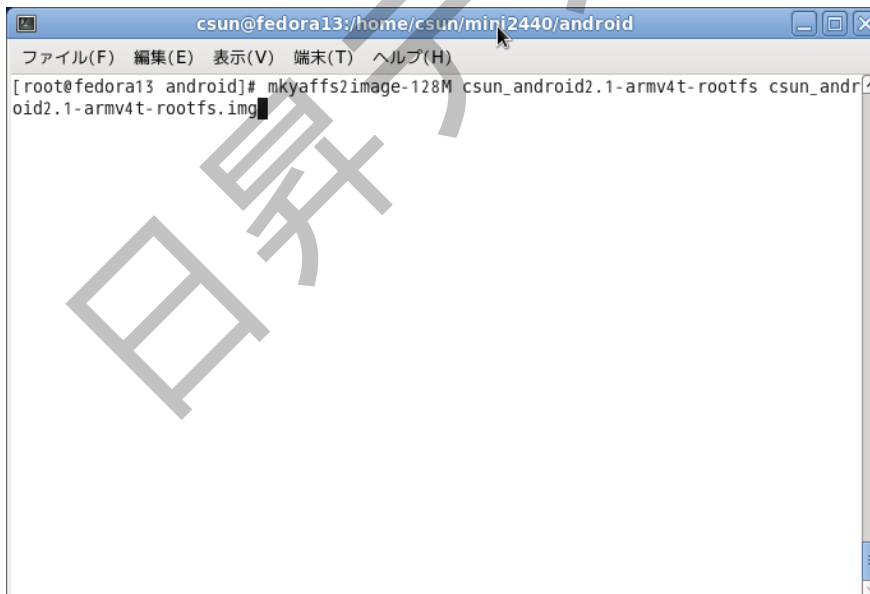
第四章 イメージファイルシステム yaffs2 の作成

1、書き込み用の yaffs2 イメージファイルシステムを作成するには [mkyaffs2image-128M](#) を実行する。



```
csun@fedora13:/home/csun/mini2440/android
ファイル(F) 編集(E) 表示(V) 端末(T) ヘルプ(H)
[csun@fedora13 ~]$ su
cst/パスワード:
[root@fedora13 csun]# ls
kddi   ダウンロード デスクトップ ビデオ 画像
mini2440 テンプレート ドキュメント 音楽 公開
[root@fedora13 csun]# cd mini2440/
[root@fedora13 mini2440]# ls
android other
[root@fedora13 mini2440]# cd android
[root@fedora13 android]# ls
csun_android2.1-armv4t-rootfs  csun_android2.1-armv4t_kernel
csun_android2.1-armv4t-rootfs.tgz csun_android2.1-armv4t_kernel.tgz
[root@fedora13 android]# mkyaffs2image-128M
mkyaffs2image: image building tool for YAFFS2 built Jul  9 2009
usage: mkyaffs2image dir image_file [convert]
       dir         the directory tree to be converted
       image_file  the output file to hold the image
       'convert'   produce a big-endian image from a little-endian machine
[root@fedora13 android]#
```

2、`/opt/FriendlyARM/android` フォルダで下記コマンドを実行して、`csun_android2.1-armv4t-rootfs.img` のターゲットに書き込む用のイメージファイルが作成される。
`#mkyaffs2image-128M csun_android2.1-armv4t-rootfs csun_android2.1-armv4t-rootfs.img`



```
csun@fedora13:/home/csun/mini2440/android
ファイル(F) 編集(E) 表示(V) 端末(T) ヘルプ(H)
[root@fedora13 android]# mkyaffs2image-128M csun_android2.1-armv4t-rootfs csun_andr
oid2.1-armv4t-rootfs.img
```

実行後の結果は下記：



```
csun@fedora13:/home/csun/mini2440/android
ファイル(F) 編集(E) 表示(V) 端末(T) ヘルプ(H)
Object 848, csun_android2.1-armv4t-rootfs/bin/true is a symlink to "busybox"
Object 849, csun_android2.1-armv4t-rootfs/bin/ln is a symlink to "busybox"
Object 850, csun_android2.1-armv4t-rootfs/bin/wpa_supplicant is a file, 1061 data c
hunks written
Object 851, csun_android2.1-armv4t-rootfs/bin/cpio is a symlink to "busybox"
Object 852, csun_android2.1-armv4t-rootfs/bin/cp is a symlink to "busybox"
Object 853, csun_android2.1-armv4t-rootfs/bin/ping is a symlink to "busybox"
Object 854, csun_android2.1-armv4t-rootfs/bin/watch is a symlink to "busybox"
Object 855, csun_android2.1-armv4t-rootfs/bin/date is a symlink to "busybox"
Object 856, csun_android2.1-armv4t-rootfs/bin/mkdir is a symlink to "busybox"
Object 857, csun_android2.1-armv4t-rootfs/bin/mktemp is a symlink to "busybox"
Object 858, csun_android2.1-armv4t-rootfs/bin/mountpoint is a symlink to "busybox"
Object 859, csun_android2.1-armv4t-rootfs/init.rc is a file, 7 data chunks written
Object 860, csun_android2.1-armv4t-rootfs/sdcard is a directory
Object 861, csun_android2.1-armv4t-rootfs/linuxrc is a file, 1 data chunks written
Object 862, csun_android2.1-armv4t-rootfs/sqlite_stmt_journals is a directory
Operation complete.
606 objects in 54 directories
38626 NAND pages
[root@fedora13 android]# ls
csun_android2.1-armv4t-rootfs          csun_android2.1-armv4t_kernel
csun_android2.1-armv4t-rootfs.img     csun_android2.1-armv4t_kernel.tgz
csun_android2.1-armv4t-rootfs.tgz
[root@fedora13 android]#
```

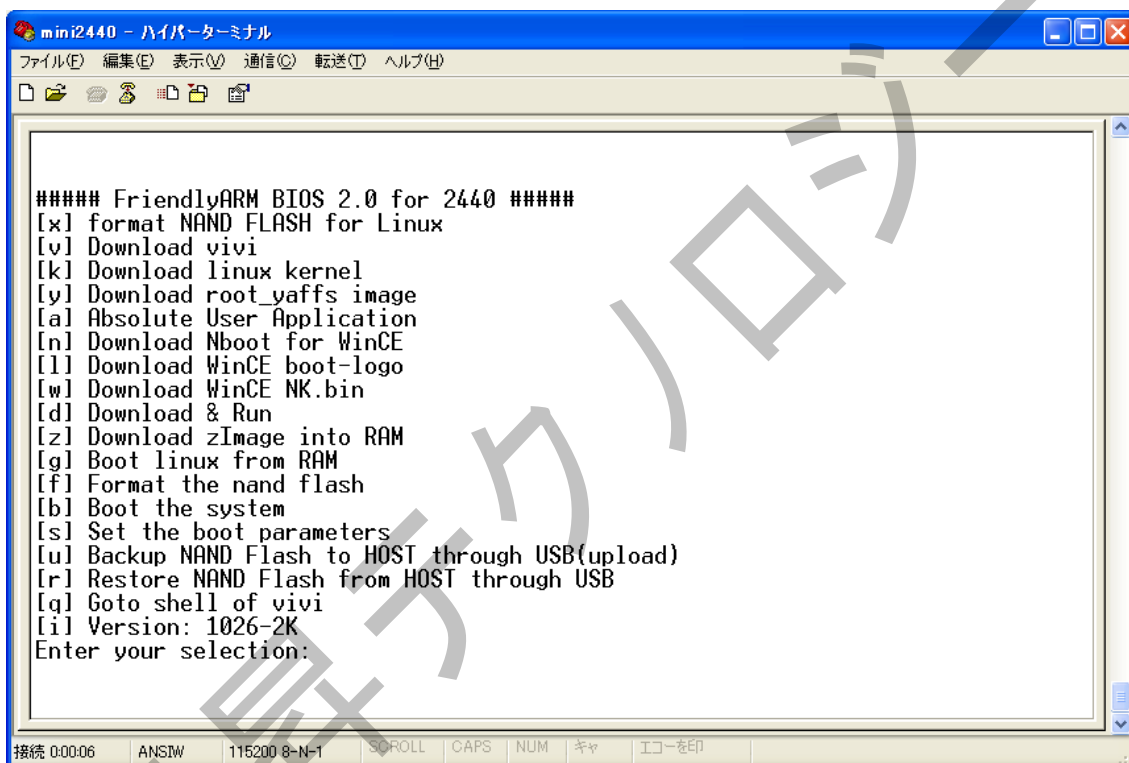


第五章 ターゲットボードに書き込む及び実行

ターゲットボード (mini2440) で内蔵している `supervivi` を利用して上記操作で作成したカーネルファイル `zImage` とイメージファイル `csun_android2.1-armv4t-rootfs.img` をボードに書き込んで実行する。

※USB ドライバのインストール、HyperTerminal の設定、USB 通じてダウンロード、書き込みについての手順は `mini2440` のマニュアルを参照する事。

1、ボードの S2 スイッチを NOR に切り替える。電源入れて BIOS モード (`supervivi` モード) で起動する。下記画面になる：



```

mini2440 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
##### FriendlyARM BIOS 2.0 for 2440 #####
[x] format NAND FLASH for Linux
[v] Download vivi
[k] Download linux kernel
[y] Download root_yaffs image
[a] Absolute User Application
[n] Download Nboot for WinCE
[l] Download WinCE boot-logo
[w] Download WinCE NK.bin
[d] Download & Run
[z] Download zImage into RAM
[g] Boot linux from RAM
[f] Format the nand flash
[b] Boot the system
[s] Set the boot parameters
[u] Backup NAND Flash to HOST through USB(upload)
[r] Restore NAND Flash from HOST through USB
[q] Goto shell of vivi
[i] Version: 1026-2K
Enter your selection:
接続 0:00:06  ANSW  115200 8-N-1  SCROLL  CAPS  NUM  キャ  エコーを印
  
```

このメニュー画面で下記操作を行う：

- ① 「x」を入力してフォーマットする。
- ② 「v」を入力する。Bootloader をダウンロードする (`vboot.bin` あるいは `supervivi-128M` どちらでも可)。
- ③ 「k」を入力する。Android のカーネルファイル `zImage_X35` をダウンロードする。
- ④ 「y」を入力する。`csun_android2.1-armv4t-rootfs.img` イメージファイルをダウンロードする。

2、上記操作後、「b」を入力するか、あるいは S2 を Nand Flash に切り替えて、再起動する。

初回起動時に、リカバリー画面が出ますが、“+”がなくなるまで中心部をクリックする。少し待っていれば、下記の画面が表示する：

