

SH用 RTLinux クロス開発環境の構築

GPG-SH03R(SH-4 搭載 CF 型 CPU ボード用ソフトウェア(RT Linux))と DMG-ZSH01(CPZ-SH03 Linux 開発キット)を使用して、各種ディストリビューション上に CTP/CPZ-SH03 用の RTLinux クロス開発環境を構築する手順を説明します。

尚、各種ディストリビューションは、当社製 PentiumM 搭載 CPU モジュールを使用して行っています。

また、すべてのディストリビューションで、OS 起動後、root でログインして操作していません。

注) 記載している内容は、動作の保証はありません。使用される場合は、すべて自己責任でお願いします。

弊社 Web site からファイルをダウンロードする場合は、ユーザ ID を取得してください。

1. Debian Sarge 編

1.1 クロス開発環境に必要なコードを準備

まず、GPG-SH03R に収録している kernel 関連のファイル類を所定のディレクトリにコピーします。

GPG-SH03R を USB 接続 CD ドライブにセットし、CTP/CPZ-SH03(以降、CPU モジュール)に接続した後、mount /cdrom と実行します。

mkdir aaa ; cd aaa ; tar zxf /cdrom/deb-pac-cf-rt-linux-sh03-20051025.tar.gz と実行し、アーカイブファイルを展開します。(20051025 は、ご使用の GPG-SH03R に収録されたファイル名に合わせてください)

"aaa"は、ファイルを展開するためだけに作成した一時指定の任意のディレクトリ名です。他の名前を指定されても問題ありません。

ls usr/src/と実行すると、linux-2.4.17 と rtlinux-3.2-pre1 が存在する事がわかります。この2個のディレクトリを、/usr/src/に移動します。

```
mv usr/src/* /usr/src/
```

最後に、umount /cdrom ; eject /cdrom と実行し、GPG-SH03R を取り出します。

```
GNOME 端末
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(I) ヘルプ(H)
greenart:~# mount /cdrom
greenart:~# ls /cdrom/
bin                floppy             mnt               tmp
boot              gpgsh03r.ver     opt               usr
cdrom             home              proc              var
deb-pac-cf-rt-linux-sh03-20051025.tar.gz  initrd            root
dev               lib               rr_moved
etc              lost+found       sbin
greenart:~# mkdir aaa ; cd aaa
greenart:~/aaa# tar xzf /cdrom/deb-pac-cf-rt-linux-sh03-20051025.tar.gz
greenart:~/aaa# ls usr/src/
linux-2.4.17  rtlinux-3.2-pre1
greenart:~/aaa# mv usr/src/* /usr/src/
greenart:~/aaa# umount /cdrom
greenart:~/aaa# eject /cdrom
greenart:~/aaa#
```

(実行例)

1.2 クロスコンパイラ等の準備

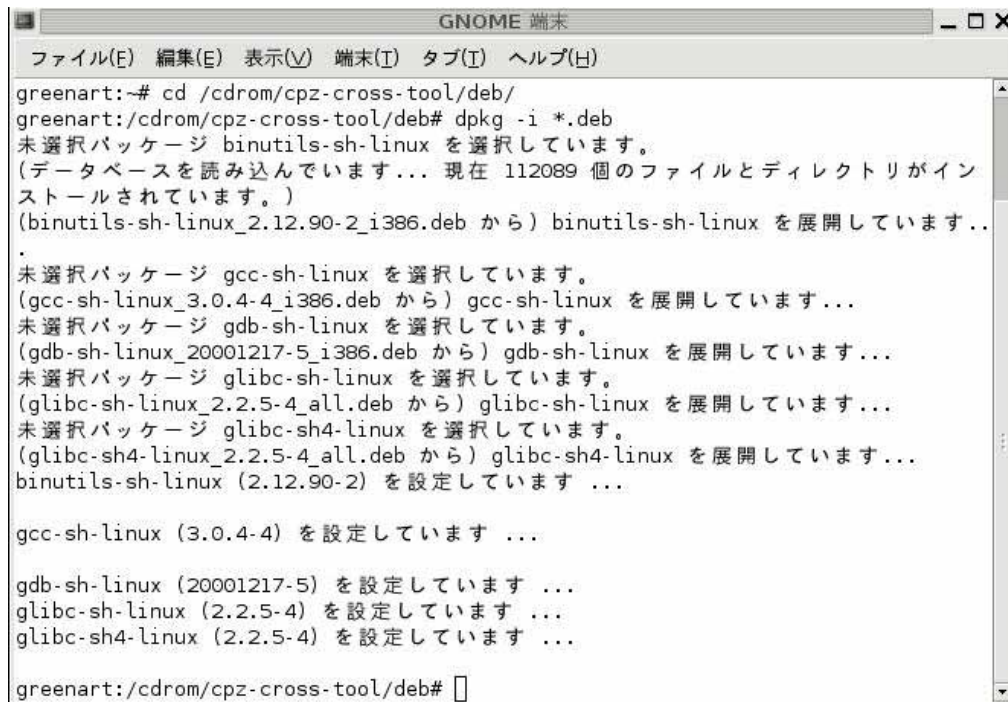
次に、DMG-ZSH01 に収録しているクロスコンパイラ等のパッケージをインストールします。

DMG-ZSH01 を USB 接続 CD ドライブにセットし、CPU モジュールに接続した後、`mount /cdrom` と実行します。

`ls /cdrom/cpz-cross-tool/deb/`で、*.deb ファイルを確認できます。

```
GNOME 端末
greenart:~# mount /cdrom
greenart:~# ls /cdrom/cpz-cross-tool/deb/
binutils-sh-linux_2.12.90-2_i386.deb  glibc-sh-linux_2.2.5-4_all.deb
gcc-sh-linux_3.0.4-4_i386.deb         glibc-sh4-linux_2.2.5-4_all.deb
gdb-sh-linux_20001217-5_i386.deb
greenart:~#
```

cd /cdrom/cpz-cross-tool/deb/ ; dpkg -i *.deb と実行し、これらのパッケージをインストールします。



```
GNOME 端末
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(T) ヘルプ(H)
greenart:~# cd /cdrom/cpz-cross-tool/deb/
greenart:/cdrom/cpz-cross-tool/deb# dpkg -i *.deb
未選択パッケージ binutils-sh-linux を選択しています。
(データベースを読み込んでいます... 現在 112089 個のファイルとディレクトリがインストールされています。)
(binutils-sh-linux_2.12.90-2_i386.deb から) binutils-sh-linux を展開しています...
.
未選択パッケージ gcc-sh-linux を選択しています。
(gcc-sh-linux_3.0.4-4_i386.deb から) gcc-sh-linux を展開しています...
未選択パッケージ gdb-sh-linux を選択しています。
(gdb-sh-linux_20001217-5_i386.deb から) gdb-sh-linux を展開しています...
未選択パッケージ glibc-sh-linux を選択しています。
(glibc-sh-linux_2.2.5-4_all.deb から) glibc-sh-linux を展開しています...
未選択パッケージ glibc-sh4-linux を選択しています。
(glibc-sh4-linux_2.2.5-4_all.deb から) glibc-sh4-linux を展開しています...
binutils-sh-linux (2.12.90-2) を設定しています ...

gcc-sh-linux (3.0.4-4) を設定しています ...

gdb-sh-linux (20001217-5) を設定しています ...
glibc-sh-linux (2.2.5-4) を設定しています ...
glibc-sh4-linux (2.2.5-4) を設定しています ...

greenart:/cdrom/cpz-cross-tool/deb#
```

/usr/sh-linux/include/asm と linux のシンボリックリンクを、GPG-SH03R からコピーした linux-2.4.17 の include に変更します。実行例は、以下の通りです。



```
GNOME 端末
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(T) ヘルプ(H)
greenart:~# ls -l /usr/sh-linux/include/asm
lrwxrwxrwx 1 10001 10001 33 2006-06-20 13:49 /usr/sh-linux/include/asm -> /usr/src/linux-sh-2.4/include/asm
greenart:~# ls -l /usr/sh-linux/include/linux
lrwxrwxrwx 1 10001 10001 35 2006-06-20 13:49 /usr/sh-linux/include/linux -> /usr/src/linux-sh-2.4/include/linux
greenart:~# rm /usr/sh-linux/include/asm
greenart:~# rm /usr/sh-linux/include/linux
greenart:~# ln -s /usr/src/linux-2.4.17/include/asm /usr/sh-linux/include/asm
greenart:~# ln -s /usr/src/linux-2.4.17/include/linux /usr/sh-linux/include/linux
greenart:~# ls -l /usr/sh-linux/include/asm
lrwxrwxrwx 1 root root 33 2006-06-20 13:51 /usr/sh-linux/include/asm -> /usr/src/linux-2.4.17/include/asm
greenart:~# ls -l /usr/sh-linux/include/linux
lrwxrwxrwx 1 root root 35 2006-06-20 13:51 /usr/sh-linux/include/linux -> /usr/src/linux-2.4.17/include/linux
greenart:~# umount /cdrom
greenart:~# eject /cdrom
greenart:~#
```

現在のリンク先を確認。

リンク先を/usr/src/linux-2.4.17/include/asm と linux に、それぞれ変更。

変更後のリンク先を確認。

最後に、umount /cdrom ; eject /cdrom と実行し、DMG-ZSH01 を取り出します。

2. Vine v3.2, Red Hat Linux v9, Red Hat Enterprise Linux ES v4, Fedora Core 5 編

これらのディストリビューションは、ほとんど同じ手順で行います。主な違いは、USB 接続 CD を自動認識するかどうかと、自動認識した場合のマウント先です。

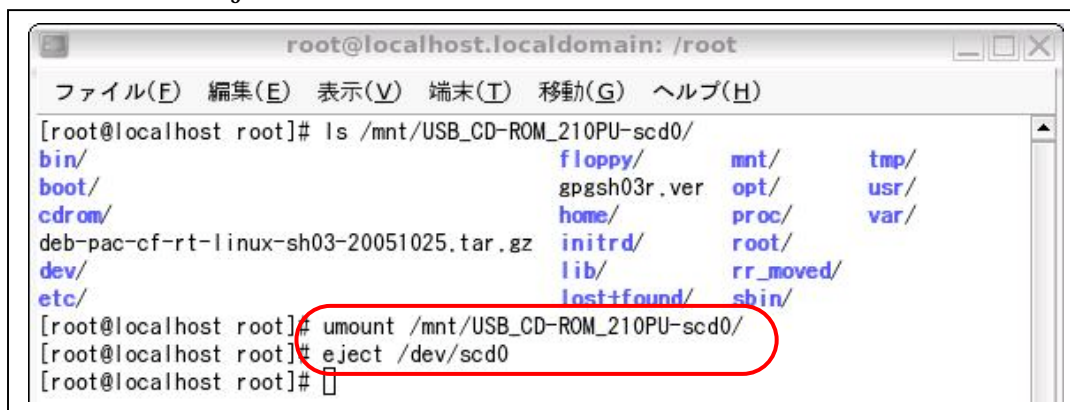
環境によっては、**Fedora Core 5** の場合、USB 接続 CD をセットしても自動マウントしないので、mount -t iso9660 /dev/scd0 /mnt/でマウントしています。

eject は、/dev/scd0 で指定しました。

Vine v3.2 の場合、USB 接続 CD をセットすると、デスクトップ上に USB-CD のアイコンが現れます。当社確認環境では、自動的に/mnt/USB_CD-ROM_210PU-scd0/にマウントされました。



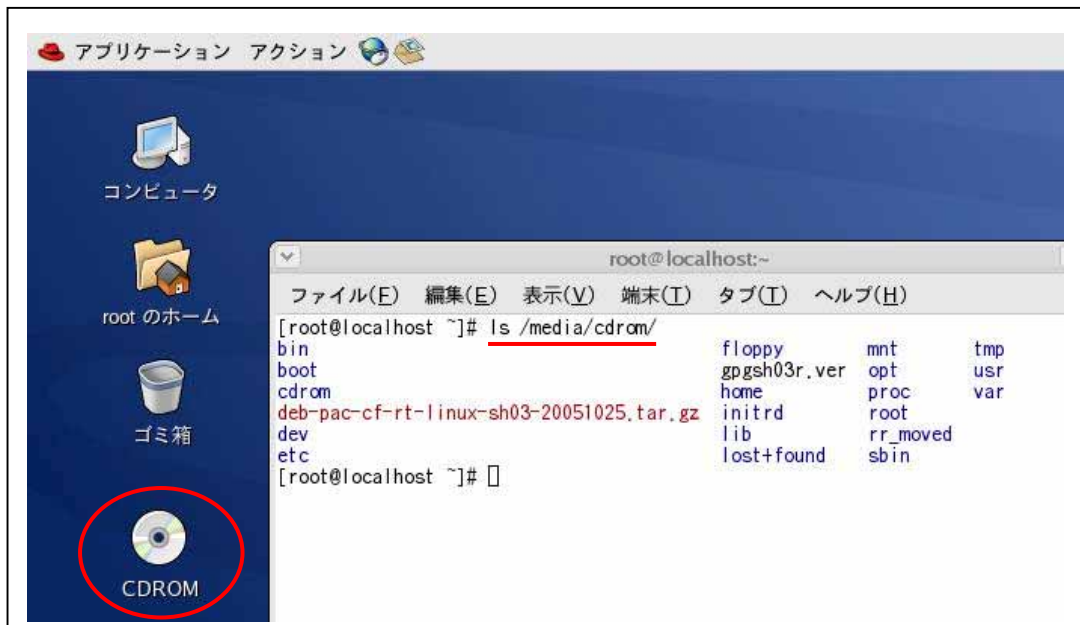
以下は、umount, eject の実行例です。



Red Hat Linux v9 の場合、USB 接続 CD をセットした後、mount /mnt/cdrom でマウントします。マウントすると、デスクトップ上に CDROM のアイコンが現れ、CDROM 内のファイル一覧を表示します。

umount は umount /mnt/cdrom/で、eject は eject /mnt/cdrom/で行います。

Red Hat Enterprise ES v4 の場合、USB 接続 CD をセットすると、デスクトップ上に CDROM のアイコンが現れます。当社確認環境では、自動的に/media/CDROM/にマウントされました。



以下は、umount, eject の実行例です。



ここからは、Fedora Core 5 を使用して説明します。mount, eject, マウント先のディレクトリは、ご使用のディストリビューションに合わせて、読み替えてください。

2.1 クロス開発環境に必要なコードを準備

まず、GPG-SH03R に収録している kernel 関連のファイル類を所定のディレクトリにコピーします。

GPG-SH03R を USB 接続 CD ドライブにセットし、CTP/CPZ-SH03(以降、CPU モジュール)に接続した後、mount します。

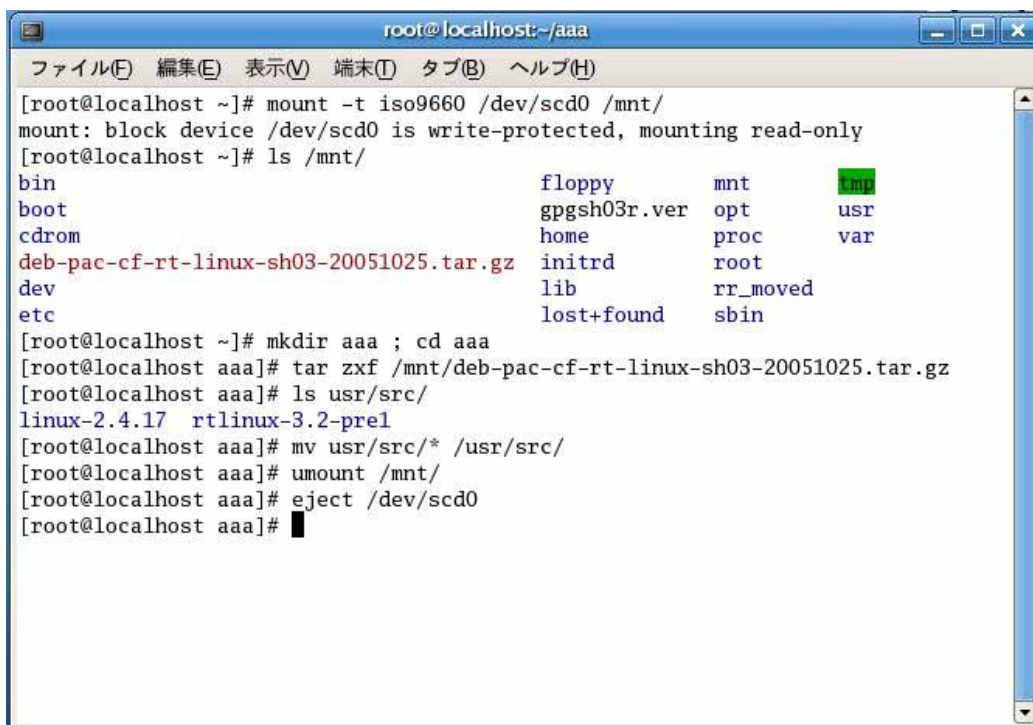
`mkdir aaa ; cd aaa ; tar zxf /cdrom/deb-pac-cf-rt-linux-sh03-20051025.tar.gz` と実行し、アーカイブファイルを展開します。(20051025 は、ご使用の GPG-SH03R に収録されたファイル名に合わせてください)

”aaa”は、ファイルを展開するためだけに作成した一時指定の任意のディレクトリ名です。他の名前を指定されても問題ありません。

`ls usr/src/`と実行すると、`linux-2.4.17` と `rtlinux-3.2-pre1` が存在する事がわかります。この2個のディレクトリを、`/usr/src/`に移動します。

```
mv usr/src/* /usr/src/
```

最後に、`umount /mnt` を実行し、GPG-SH03R を取り出します。



```
root@localhost:~/aaa
ファイル(F) 編集(E) 表示(V) 端末(T) タブ(B) ヘルプ(H)
[root@localhost ~]# mount -t iso9660 /dev/scd0 /mnt/
mount: block device /dev/scd0 is write-protected, mounting read-only
[root@localhost ~]# ls /mnt/
bin                floppy            mnt               tmp
boot               gpgsh03r.ver    opt               usr
cdrom              home             proc              var
deb-pac-cf-rt-linux-sh03-20051025.tar.gz  initrd           root
dev                lib               rr_moved
etc                lost+found       sbin
[root@localhost ~]# mkdir aaa ; cd aaa
[root@localhost aaa]# tar zxf /mnt/deb-pac-cf-rt-linux-sh03-20051025.tar.gz
[root@localhost aaa]# ls usr/src/
linux-2.4.17  rtlinux-3.2-pre1
[root@localhost aaa]# mv usr/src/* /usr/src/
[root@localhost aaa]# umount /mnt/
[root@localhost aaa]# eject /dev/scd0
[root@localhost aaa]#
```

(実行例)

2.2 クロスコンパイラ等の準備

次に、DMG-ZSH01 に収録しているクロスコンパイラ等のパッケージをインストールします。

DMG-ZSH01 を USB 接続 CD ドライブにセットし、CPU モジュールに接続した後、mount します。

ls /mnt/cpz-cross-tool/rpm/で、*.rpm ファイルを確認できます。

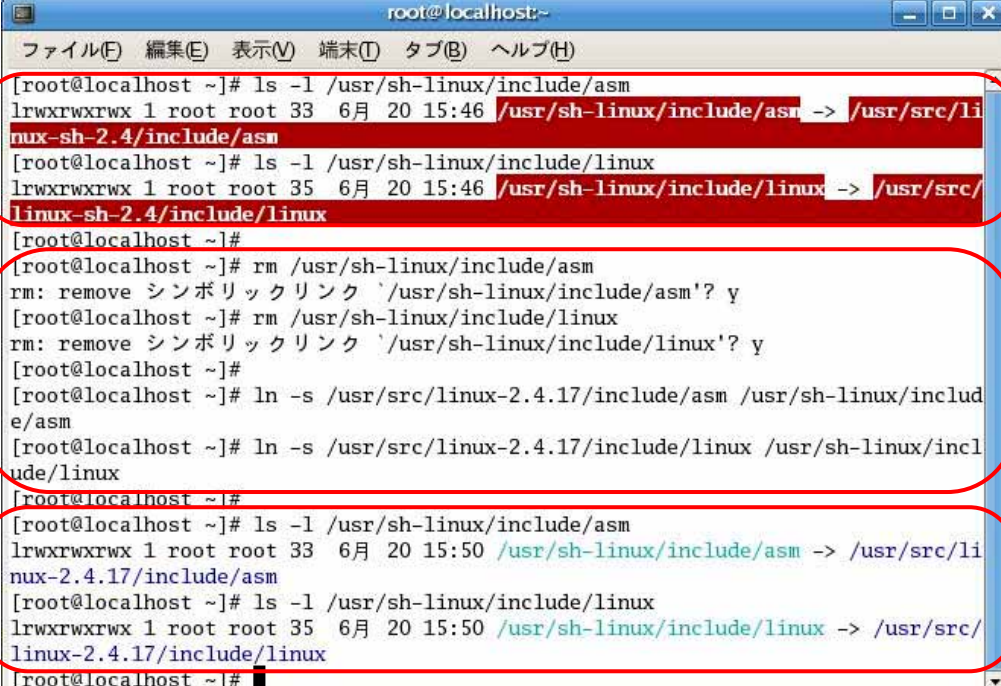
```
[root@localhost ~]# mount -t iso9660 /dev/scd0 /mnt/
mount: block device /dev/scd0 is write-protected, mounting read-only
[root@localhost ~]# ls /mnt/cpz-cross-tool/rpm/
binutils-sh-linux-2.12.90-1.i386.rpm  glibc-sh-linux-2.2.5-3.noarch.rpm
gcc-sh-linux-3.0.4-3.i386.rpm        glibc-sh4-linux-2.2.5-3.noarch.rpm
gdb-sh-linux-20001217-4.i386.rpm
[root@localhost ~]#
```

cd /mnt/cpz-cross-tool/rpm/ ; rpm -ivh *.rpm と実行し、これらのパッケージをインストールします。



```
root@localhost:/mnt/cpz-cross-tool/rpm
ファイル(F) 編集(E) 表示(V) 端末(T) タブ(B) ヘルプ(H)
[root@localhost ~]# mount -t iso9660 /dev/scd0 /mnt/
mount: block device /dev/scd0 is write-protected, mounting read-only
[root@localhost ~]# ls /mnt/cpz-cross-tool/rpm/
binutils-sh-linux-2.12.90-1.i386.rpm  glibc-sh-linux-2.2.5-3.noarch.rpm
gcc-sh-linux-3.0.4-3.i386.rpm        glibc-sh4-linux-2.2.5-3.noarch.rpm
gdb-sh-linux-20001217-4.i386.rpm
[root@localhost ~]# cd /mnt/cpz-cross-tool/rpm
[root@localhost rpm]# rpm -ivh *.rpm
準備中... ##### [100%]
 1:glibc-sh-linux          ##### [ 20%]
 2:binutils-sh-linux      ##### [ 40%]
 3:gcc-sh-linux           ##### [ 60%]
 4:gdb-sh-linux           ##### [ 80%]
 5:glibc-sh4-linux        ##### [100%]
[root@localhost rpm]#
```

/usr/sh-linux/include/asm と linux のシンボリックリンクを、GPG-SH03R からコピーした linux-2.4.17 の include に変更します。実行例は、以下の通りです。



```
root@localhost:~  
ファイル(F) 編集(E) 表示(V) 端末(T) タブ(B) ヘルプ(H)  
[root@localhost ~]# ls -l /usr/sh-linux/include/asm  
lrwxrwxrwx 1 root root 33 6月 20 15:46 /usr/sh-linux/include/asm -> /usr/src/linux-sh-2.4/include/asm  
[root@localhost ~]# ls -l /usr/sh-linux/include/linux  
lrwxrwxrwx 1 root root 35 6月 20 15:46 /usr/sh-linux/include/linux -> /usr/src/linux-sh-2.4/include/linux  
[root@localhost ~]#  
[root@localhost ~]# rm /usr/sh-linux/include/asm  
rm: remove シンボリックリンク `/usr/sh-linux/include/asm'? y  
[root@localhost ~]# rm /usr/sh-linux/include/linux  
rm: remove シンボリックリンク `/usr/sh-linux/include/linux'? y  
[root@localhost ~]#  
[root@localhost ~]# ln -s /usr/src/linux-2.4.17/include/asm /usr/sh-linux/include/asm  
[root@localhost ~]# ln -s /usr/src/linux-2.4.17/include/linux /usr/sh-linux/include/linux  
[root@localhost ~]#  
[root@localhost ~]# ls -l /usr/sh-linux/include/asm  
lrwxrwxrwx 1 root root 33 6月 20 15:50 /usr/sh-linux/include/asm -> /usr/src/linux-2.4.17/include/asm  
[root@localhost ~]# ls -l /usr/sh-linux/include/linux  
lrwxrwxrwx 1 root root 35 6月 20 15:50 /usr/sh-linux/include/linux -> /usr/src/linux-2.4.17/include/linux  
[root@localhost ~]#
```

現在のリンク先を確認。

リンク先を/usr/src/linux-2.4.17/include/asm と linux に、それぞれ変更。

変更後のリンク先を確認。

最後に、umount /mnt/ ; eject /dev/scd0 と実行し、DMG-ZSH01 を取り出します。



```
[root@localhost ~]# umount /mnt/  
[root@localhost ~]# eject /dev/scd0  
[root@localhost ~]#
```

3. RTLinux のサンプルプログラムを使用してクロスコンパイルする

クロス開発環境の確認用に、RTLinux のコードをダウンロードし、クロスコンパイル～実機での動作確認を説明します。

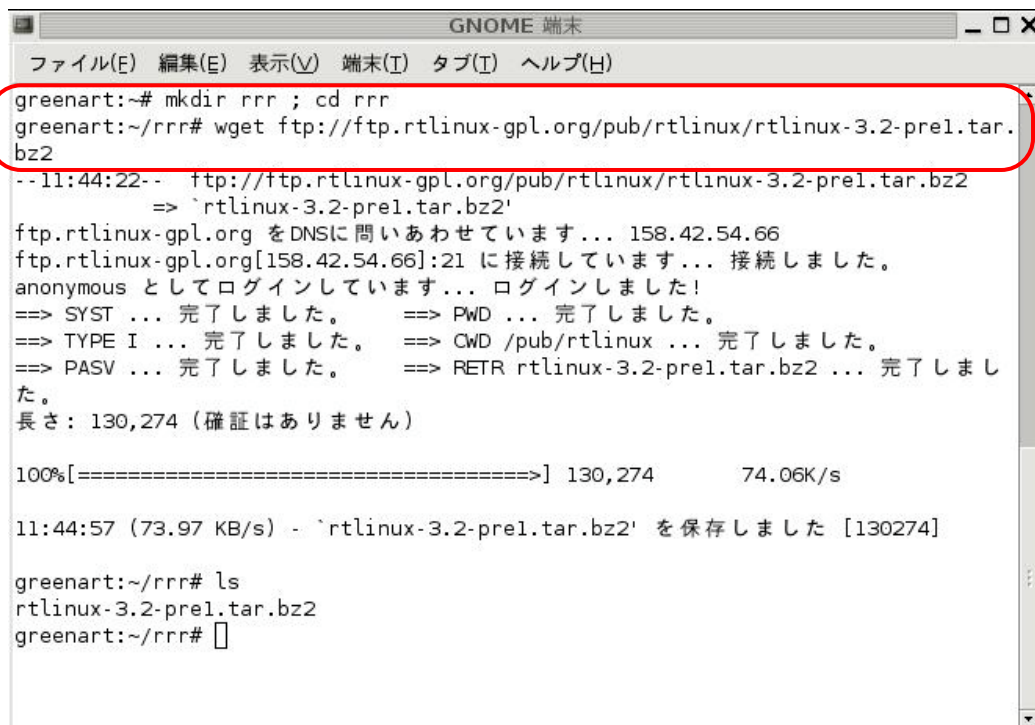
この操作は、各種ディストリビューションで違いが無いので、ここでは、Debian を使用して説明します。

3.1 サンプルプログラムをダウンロードする

平成 18 年 6 月 時点では、<ftp://ftp.rtlinux-gpl.org/pub/rtlinux/> からサンプルプログラムがダウンロードできます。GPG-SH03R は、RTLinux v3.2-pre1 なので、`rtlinux-3.2-pre1.tar.bz2` をダウンロードします。

以下は、実行例です。この実行例では、`rrr` ディレクトリを作り、そのディレクトリに `wget` コマンドを使用してダウンロードしています。

”`rrr`”は、ファイルを展開するためだけに作成した一時指定の任意のディレクトリ名です。他の名前を指定されても問題ありません。



```
GNOME 端末
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(T) ヘルプ(H)
greenart:~# mkdir rrr ; cd rrr
greenart:~/rrr# wget ftp://ftp.rtlinux-gpl.org/pub/rtlinux/rtlinux-3.2-pre1.tar.bz2
--11:44:22--  ftp://ftp.rtlinux-gpl.org/pub/rtlinux/rtlinux-3.2-pre1.tar.bz2
=> `rtlinux-3.2-pre1.tar.bz2'
ftp.rtlinux-gpl.org をDNSに問いあわせています... 158.42.54.66
ftp.rtlinux-gpl.org[158.42.54.66]:21 に接続しています... 接続しました。
anonymous としてログインしています... ログインしました!
==> SYST ... 完了しました。      ==> PWD ... 完了しました。
==> TYPE I ... 完了しました。    ==> CWD /pub/rtlinux ... 完了しました。
==> PASV ... 完了しました。     ==> RETR rtlinux-3.2-pre1.tar.bz2 ... 完了しまし
た。
長さ: 130,274 (確証はありません)

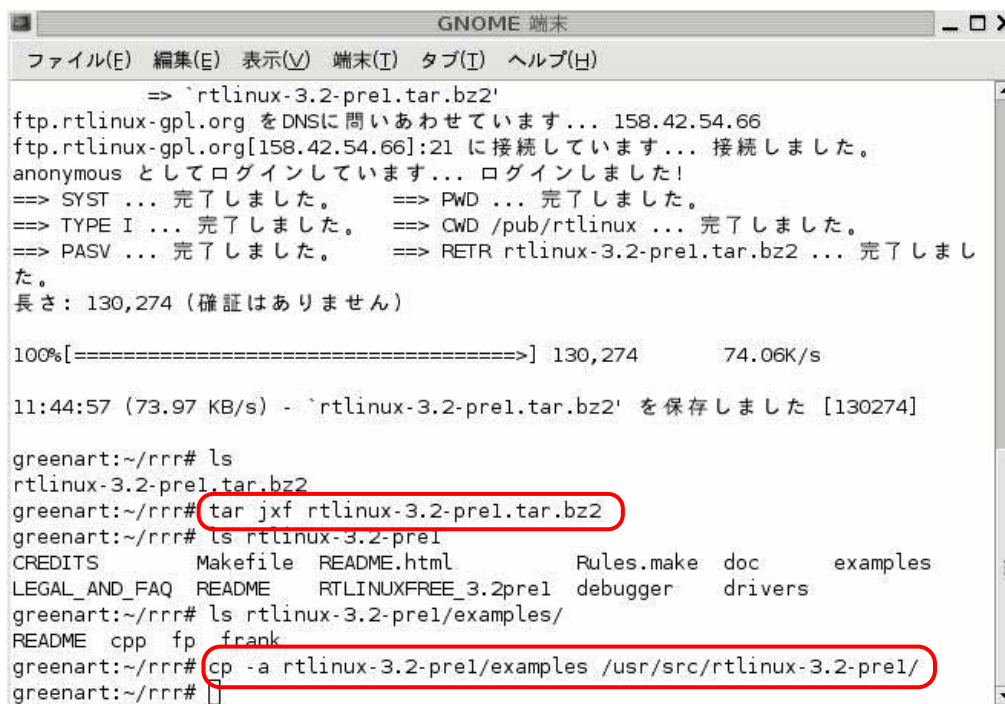
100%[=====] 130,274      74.06K/s

11:44:57 (73.97 KB/s) - `rtlinux-3.2-pre1.tar.bz2' を保存しました [130274]

greenart:~/rrr# ls
rtlinux-3.2-pre1.tar.bz2
greenart:~/rrr#
```

次に、アーカイブファイルを展開します。

クロスコンパイルの実験は、`./examples/frank` で行います。`./examples/frank/Makefile` の関係で、`/usr/src/rtnlinux-3.2-pre1` ディレクトリにコピーします。



```
GNOME 端末
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(T) ヘルプ(H)
=> `rtnlinux-3.2-pre1.tar.bz2'
ftp.rtnlinux-gpl.org をDNSに問いあわせています... 158.42.54.66
ftp.rtnlinux-gpl.org[158.42.54.66]:21 に接続しています... 接続しました。
anonymous としてログインしています... ログインしました!
==> SYST ... 完了しました。    ==> PWD ... 完了しました。
==> TYPE I ... 完了しました。  ==> CWD /pub/rtnlinux ... 完了しました。
==> PASV ... 完了しました。    ==> RETR rtnlinux-3.2-pre1.tar.bz2 ... 完了しまし
た。
長さ: 130,274 (確認はありません)

100%[=====>] 130,274      74.06K/s

11:44:57 (73.97 KB/s) - `rtnlinux-3.2-pre1.tar.bz2' を保存しました [130274]

greenart:~/rrr# ls
rtnlinux-3.2-pre1.tar.bz2
greenart:~/rrr# tar jxf rtnlinux-3.2-pre1.tar.bz2
greenart:~/rrr# ls rtnlinux-3.2-pre1
CREDITS      Makefile  README.html      Rules.make  doc        examples
LEGAL_AND_FAQ README    RTLINUXFREE_3.2pre1 debugger    drivers
greenart:~/rrr# ls rtnlinux-3.2-pre1/examples/
README cpp fp frank
greenart:~/rrr# cp -a rtnlinux-3.2-pre1/examples /usr/src/rtnlinux-3.2-pre1/
greenart:~/rrr#
```

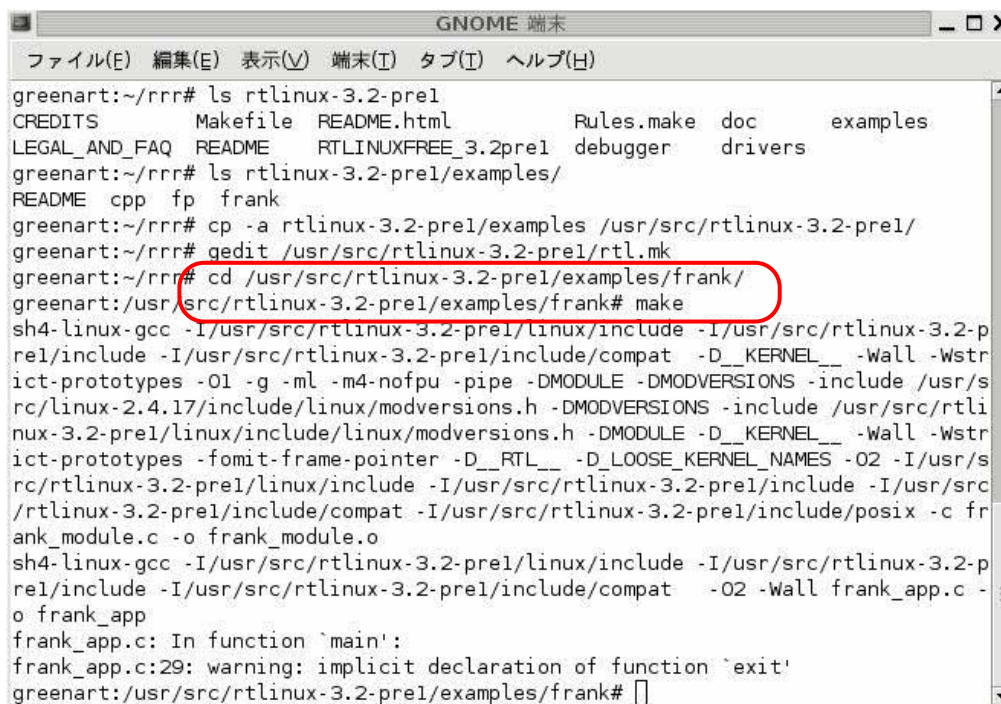
`./examples/frank/Makefile` は、`rtl.mk` を使用しています。

`/usr/src/rtnlinux-3.2-pre1/rtl.mk` は、SH のセルフコンパイル環境の設定になっているので、`/usr/src/rtnlinux-3.2-pre1/rtl.mk` の真ん中あたりの `CC = gcc` を `CC = sh4-linux-gcc` に、終わり部分の `LD = ld -EL` を `LD = sh4-linux-ld -EL`、`AR = ar` を `AR = sh4-linux-ar` に変更します。(以下は、変更後の内容です)

```
include -I/usr/src/rtnlinux-3.2-pre1/include/compat -I/usr/src/
rtnlinux-3.2-pre1/include/posix
ARCH = sh
CC = sh4-linux-gcc
CXXFLAGS = -D __KERNEL__ -Wall -Wstrict-prototypes -O1 -g -ml -m4-nofpu
-pipe -DMODULE -DMODVERSIONS -include /usr/src/linux-2.4.17/include/
linux/modversions.h -DMODVERSIONS -include /usr/src/rtnlinux-3.2-pre1/
linux/include/linux/modversions.h -DMODULE -D __KERNEL__ -Wall -Wstrict-
prototypes -fomit-frame-pointer -D __RTL__ -D LOOSE_KERNEL_NAMES -I/usr/
src/rtnlinux-3.2-pre1/linux/include -I/usr/src/rtnlinux-3.2-pre1/include
-I/usr/src/rtnlinux-3.2-pre1/include/compat -I/usr/src/rtnlinux-3.2-pre1/
include/posix -fno-exceptions -fno-rtti
LD = sh4-linux-ld -EL
AR = sh4-linux-ar
```

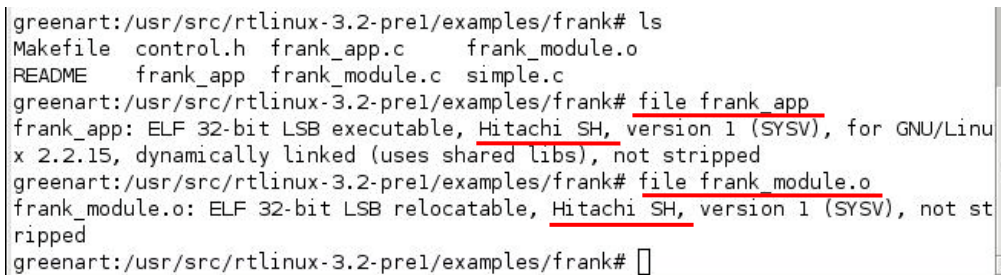
準備が出来たので、RTLinux のサンプルプログラム(`./example/frank`)を、クロスコンパイルしてみましょう。

cd で frank のディレクトリに移動し、make を実行します。



```
GNOME 端末
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(T) ヘルプ(H)
greenart:~/rrr# ls rtlinux-3.2-pre1
CREDITS      Makefile  README.html      Rules.make  doc        examples
LEGAL_AND_FAQ  README  RTLINUXFREE_3.2pre1  debugger    drivers
greenart:~/rrr# ls rtlinux-3.2-pre1/examples/
README  cpp  fp  frank
greenart:~/rrr# cp -a rtlinux-3.2-pre1/examples /usr/src/rtlinux-3.2-pre1/
greenart:~/rrr# gedit /usr/src/rtlinux-3.2-pre1/rtl.mk
greenart:~/rrr# cd /usr/src/rtlinux-3.2-pre1/examples/frank/
greenart:~/src/rtlinux-3.2-pre1/examples/frank# make
sh4-linux-gcc -I/usr/src/rtlinux-3.2-pre1/linux/include -I/usr/src/rtlinux-3.2-pre1/include -I/usr/src/rtlinux-3.2-pre1/include/compat -D__KERNEL__ -Wall -Wstrict-prototypes -O1 -g -ml -m4-nofpu -pipe -DMODULE -DMODVERSIONS -include /usr/src/linux-2.4.17/include/linux/modversions.h -DMODVERSIONS -include /usr/src/rtlinux-3.2-pre1/linux/include/linux/modversions.h -DMODULE -D__KERNEL__ -Wall -Wstrict-prototypes -fomit-frame-pointer -D__RTL__ -D_LOOSE_KERNEL_NAMES -O2 -I/usr/src/rtlinux-3.2-pre1/linux/include -I/usr/src/rtlinux-3.2-pre1/include -I/usr/src/rtlinux-3.2-pre1/include/compat -I/usr/src/rtlinux-3.2-pre1/include/posix -c frank_module.c -o frank_module.o
sh4-linux-gcc -I/usr/src/rtlinux-3.2-pre1/linux/include -I/usr/src/rtlinux-3.2-pre1/include -I/usr/src/rtlinux-3.2-pre1/include/compat -O2 -Wall frank_app.c -o frank_app
frank_app.c: In function `main':
frank_app.c:29: warning: implicit declaration of function `exit'
greenart:~/src/rtlinux-3.2-pre1/examples/frank#
```

make の実行で、frank_app と frank_module.o の 2 つのファイルが出来ています。file コマンドで、それぞれのファイルのタイプを表示し、“Hitachi SH”の表示があれば、SH 用のクロスコンパイルは成功です。



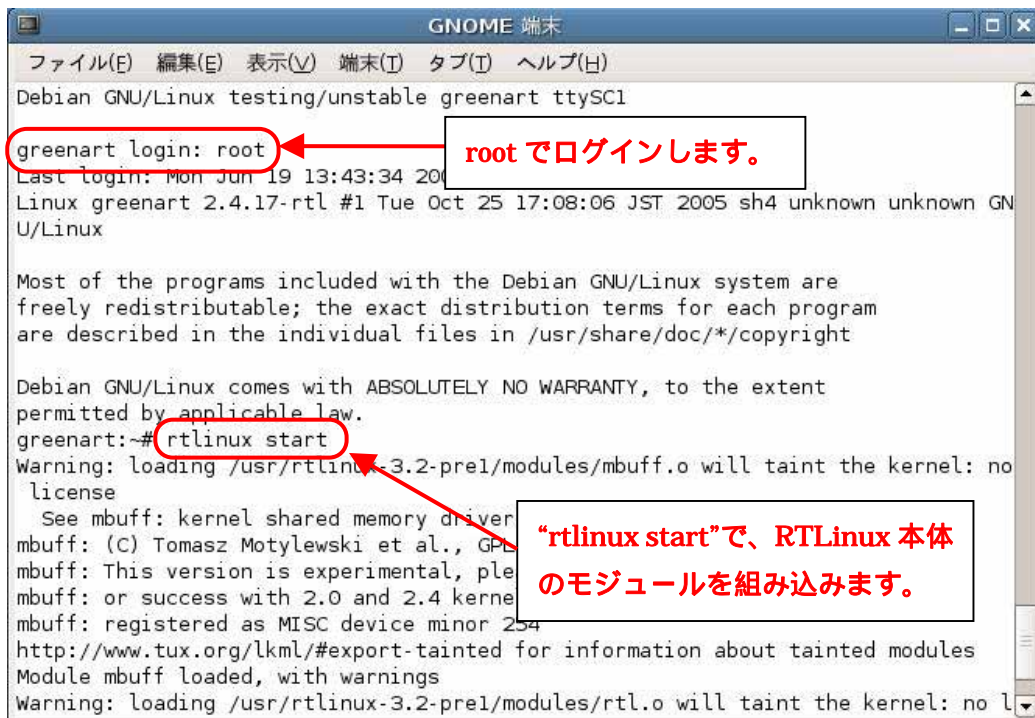
```
greenart:~/src/rtlinux-3.2-pre1/examples/frank# ls
Makefile  control.h  frank_app.c  frank_module.o
README    frank_app  frank_module.c  simple.c
greenart:~/src/rtlinux-3.2-pre1/examples/frank# file frank_app
frank_app: ELF 32-bit LSB executable, Hitachi SH, version 1 (SYSV), for GNU/Linux 2.2.15, dynamically linked (uses shared libs), not stripped
greenart:~/src/rtlinux-3.2-pre1/examples/frank# file frank_module.o
frank_module.o: ELF 32-bit LSB relocatable, Hitachi SH, version 1 (SYSV), not stripped
greenart:~/src/rtlinux-3.2-pre1/examples/frank#
```

3.2 クロスコンパイルした RTLinux のサンプルプログラムを実機で動かす

CTP/CPZ-SH03 にセットしている CF に、USB 接続の CF リーダ/ライタ等を使用し、上で作った frank_app と frank_module.o をコピーします。ここでは、/root/ディレクトリにコピーしたものとして説明します。

コピーした CF を使って CTP/CPZ-SH03 を起動します。

root でログイン後、RTLinux 本体のモジュールを組み込みます。



The image shows a terminal window titled "GNOME 端末" (GNOME Terminal) with a menu bar (File, Edit, View, Shell, Tabs, Help). The terminal output is as follows:

```
Debian GNU/Linux testing/unstable greenart ttySC1
greenart login: root
Last login: Mon Jun 19 13:43:34 2006
Linux greenart 2.4.17-rtl #1 Tue Oct 25 17:08:06 JST 2005 sh4 unknown unknown GNU/Linux

Most of the programs included with the Debian GNU/Linux system are
freely redistributable; the exact distribution terms for each program
are described in the individual files in /usr/share/doc/*/copyright

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
greenart:~# rtlinux start
Warning: loading /usr/rtlinux-3.2-pre1/modules/mbuff.o will taint the kernel: no
license
See mbuff: kernel shared memory driver
mbuff: (C) Tomasz Motylewski et al., GPL
mbuff: This version is experimental, please
mbuff: or success with 2.0 and 2.4 kernel
mbuff: registered as MISC device minor 254
http://www.tux.org/lkml/#export-tainted for information about tainted modules
Module mbuff loaded, with warnings
Warning: loading /usr/rtlinux-3.2-pre1/modules/rtl.o will taint the kernel: no l
```

root でログインします。

“rtlinux start”で、RTLinux 本体のモジュールを組み込みます。

```
Module rtl_posixio loaded, with warnings
Warning: loading /usr/rtlinux-3.2-pre1/modules/rtl_fifo.o will taint the kernel:
no license
Module rtl_fifo loaded, with warnings
Warning: loading /usr/rtlinux-3.2-pre1/modules/rtl_sched.o will taint the kernel
: no license
Module rtl_sched loaded, with warnings

Scheme: (-) not loaded, (+) loaded
(+ ) mbuff
(+ ) rtl
(+ ) rtl_fifo
(+ ) rtl_posixio
(+ ) rtl_sched
(+ ) rtl_time

greenart:~# █
```

(RTLinux 本体のモジュール組み込み完了)

サンプルプログラム(frnk)は、frnk_module.o モジュールを組み込んだ後、frnk_app を実行する事で動作します。

```
GNOME 端末
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(T) ヘルプ(H)
greenart:~# ls
frnk_app frnk_module.o
greenart:~# insmod frnk_module.o
Warning: loading frnk_module.o will taint
See http://www.tux.org/lkml/#export-tainted for information about tainted modules
Module frnk_module loaded, with warnings
greenart:~# ./frnk_app
FIFO handler: sending the "1" command to task 0; period 500000
Task 0: executing the "1" command to task 0; period 500000
FIFO handler: sending the "1" command to task 0; period 500000
Task 1: executing the "1" command to task 0; period 500000
FIFO 1: Frnk
FIFO 2: Zappa
FIFO 2: Zappa
FIFO 2: Zappa
FIFO 1: Frnk
FIFO 2: Zappa
FIFO 2: Zappa
FIFO 2: Zappa
FIFO 1: Frnk
FIFO 2: Zappa
FIFO 2: Zappa
FIFO 2: Zappa
FIFO 1: Frnk
```

このような表示になれば、RTLinux クロス開発環境上でのコンパイルは成功です。

4. 参考情報

Vine v3.2 あるいは Red Hat Linux v9 で、USB 接続 CF リーダ/ライターで接続した CF を、umount した時、“デバイス使用中”で umount 出来ない事がありました。

この場合、“fuser -vm CF のマウントポイント”(例えば、fuser -vm /mnt/MCR-CF2-sda1) と実行し、誰が使用しているか調べます。

```
[root@localhost root]# fuser -vm /mnt/MCR-CF2-sda1
                USER      PID  ACCESS COMMAND
/mnt/MCR-CF2-sda1  root    1209  f...    fam
[root@localhost root]#
```

fam が使用していた場合、“fuser -km CF のマウントポイント”(例えば、fuser -km /mnt/MCR-CF2-sda1)と実行すると、CF の umount ができます。