Software Manual

Version 1.3.1

2006 年 8 月 11 日

株式会社アットマークテクノ
http://www.atmark-techno.com/

SUZAKU 公式サイト
http://suzaku.atmark-techno.com/

目次

1. はじめに 1
1.1. マニュアルについて 1
1.2. フォントについて 1
1.3. コマンド入力例の表記について 1
1.4. 謝辞 2
1.5. 注意事項 2
2. 作業の前に 3
2.1. 準備するもの 3
2.2. 接続方法 4
2.3. ジャンパピンの設定について 4
3. 開発環境の準備 6
3.1. クロス開発環境パッケージのインストール 6
3.1.1. SUZAKU-S 6
3.1.2. SUZAKU-V 7
3.2. uClinux-distのビルドに必要なパッケージ 8
4. 使用方法 9
4.1. 起動の前に 9
4.2. 起動モード 9
4.2.1. オートブートモード 9
4.2.2. ブートローダーモード 9
4.2.3. モトローラＳレコード形式ダウンロードモード 10
4.3. 起動 11
4.4. ディレクトリ構成 12
4.5. 終了 12
4.6. ネットワーク設定 13
4.6.1. ネットワーク設定の確認 13
4.6.2. 固定IPアドレスで使用する場合 13
4.6.3. DHCPを使用する場合 14
4.7. telnetログイン 14
4.8. ファイル転送 15
4.9. Webサーバ 15
5. Flashメモリの書き換え方法 16
5.1. リージョン指定について 16
5.2. Hermitを使ってFlashメモリを書き換える 17
5.2.1. ブートローダーモードで起動する 17
5.2.2. ダウンロード 17
5.3. netflashを使ってFlashメモリを書き換える 20
5.4. モトローラSレコード形式を使ってFlashメモリを書き換える 21
6. ブートローダー 22
6.1. ブートローダーの種類 22
6.2. ブートシーケンス 22
6.2.1. 第１ステージ（BBoot） 22
6.2.2. 第２ステージ（Hermit） 22
6.2.3. 第３ステージ（カーネル） 22
6.2.4. 第４ステージ（ユーザーランド） 23
7. uClinux-distでイメージを作成 24
7.1. ソースコードアーカイブの展開 24
7.2. 設定 24

7.3. ビルド 25
8. Flashメモリについて 27
9. Flat Binary Format 29
9.1. Flat Binary Formatの特徴 29
9.2. 実行ファイルの圧縮 29
9.2.1. コンパイル済バイナリファイルを圧縮 30
9.3. スタックサイズの指定 31
9.3.1. コンパイル済バイナリファイルスタックサイズを変更 31
9.3.2. コンパイル時にスタックサイズを指定 31
9.4. ZFLAT対応カーネルを作る 32
10. Appendix 33
10.1. シリアルコンソールソフト（minicom）のインストール 33
10.2. ダウンローダ（Hermit）のインストール 33
10.3. Windows上に開発環境を構築する方法 34
10.3.1. coLinuxのインストール 34
10.3.2. 環境構築用ファイルの準備 35
10.3.3. coLinuxの実行 35
10.3.4. ネットワークの設定 35
10.3.5. coLinuxユーザの作成 36
10.3.6. Windows-coLinux間のファイル共有 36
10.3.7. クロス開発環境の導入 37
10.3.8. 特殊な場合のWindowsネットワーク設定方法 37
10.3.9. coLinuxのネットワーク設定方法 38

表目次

表　1-1 使用しているフォント 1
表　1-2 表示プロンプトと実行環境の関係 1
表　2-1 ジャンパの設定と起動時の動作 4
表　3-1 クロス開発環境パッケージ一覧 7
表　3-2 uClinux-distのビルドに必要なパッケージ一覧 8
表　4-1 シリアル通信設定 9
表　4-2 コンソールログイン時のユーザ名とパスワード 11
表　4-3 ディレクトリ構成の一覧 12
表　4-4 telnetログイン時のユーザ名とパスワード 14
表　4-5 ftpのユーザ名とパスワード 15
表　5-1 各リージョン用のイメージファイル名 16
表　5-2 各リージョン用Flashデバイス名 20
表　6-1 各ブートローダー一覧 22
表　8-1 Flashメモリマップ（SZ130-U00, Flash:8MB） 27
表　8-2 Flashメモリマップ（4MB） 27
表　8-1 Flashメモリマップ（8MB） 28
表　10-1 ネットワーク設定 38

図目次

図 2-1 SUZAKU接続例 ……4
図 2-2 ジャンパの位置 ……5
図 3-1 開発用パッケージのインストール ……6
図 3-2 環境変数PATHの設定例 ……6
図 3-3 クロス開発用パッケージのインストール ……7
図 3-4 複数パッケージのインストール ……7
図 4-1 BBoot起動画面 ……9
図 4-2 BBootメニュー画面 ……10
図 4-3 Hermit起動画面 ……10
図 4-4 起動ログ ……11
図 4-5 ifconfig実行例 ……13
図 4-6 dhcpcd-newを外す ……13
図 4-7 ifconfigファイルの編集 ……14
図 4-8 dhcpcd-newの追加 ……14
図 4-9 Webサーバ ……15
図 5-1 hermitコマンド入力例 ……17
図 5-2 Download画面 ……18
図 5-3 Download進捗ダイアログ ……19
図 5-4 Download終了画面 ……19
図 5-5 netflash実行例 ……20
図 5-6 netflashヘルプコマンド ……20
図 5-7 モトローラSレコード形式ダウンロード開始画面 ……21
図 5-8 モトローラSレコード形式ダウンロード終了画面 ……21
図 7-1 distアーカイブの展開 ……24
図 7-2 distのコンフィギュレーション ……24
図 7-3 Vendorの選択 ……24
図 7-4 Productの選択 ……25
図 7-5 Libraryの選択 ……25
図 7-6 デフォルト設定の選択 ……25
図 7-7 CustomizeおよびUpdateの選択 ……25
図 7-8 ビルド ……25
図 9-1 通常のFlat Binay Format ……30
図 9-2 圧縮されたFlat Binary Format ……30
図 9-4 スタックサイズの変更 ……31
図 9-5 FLTFLAGSによるスタックサイズの指定 ……32
図 10-1 minicomのインストール ……33
図 10-2 Hermitのインストール ……33
図 10-3 インストール先フォルダの指定 ……34
図 10-3 ネットワークの設定コマンド ……36
図 10-4 ユーザ「somebody」を作業用ユーザとして追加する場合 ……36
図 10-5 WindowsのIPアドレス:192.168.0.100、共有フォルダ名：sharedの場合 ……36
図 10-6 ifconfigコマンドの実行 ……38
図 10-7 /etc/network/interfacesファイルの編集例 ……38
図 10-8 /etc/resolve.confファイルの編集例 ……39
図 10-9 ネットワークの再設定コマンド ……39

1.はじめに

1.1.マニュアルについて

本マニュアルは、SUZAKU を使用する上で必要な情報のうち、以下の点について記載されています。

- 基本的な使い方
- Flash メモリの書き換え方法
- カーネルとユーザーランドのビルド
- カスタマイズ
- アプリケーション開発

SUZAKU の機能を最大限に引き出すために、ご活用いただければ幸いです。

1.2.フォントについて

このマニュアルでは以下のようにフォントを使っています。

表 1-1 使用しているフォント

フォント例	説明
本文中のフォント	本文
[PC ~]$ ls	プロンプトとユーザ入力文字列

1.3.コマンド入力例の表記について

このマニュアルに記載されているコマンドの入力例は、表示されているプロンプトによって、それぞれに対応した実行環境を想定して書かれています。「/」の部分はカレントディレクトリによって異なります。各ユーザのホームディレクトリは「~」で表わします。

表 1-2 表示プロンプトと実行環境の関係

プロンプト	コマンドの実行環境
[PC /]#	作業用 PC 上の特権ユーザで実行
[PC /]$	作業用 PC 上の一般ユーザで実行
[SUZAKU /]#	SUZAKU 上の特権ユーザで実行
[SUZAKU /]$	SUZAKU 上の一般ユーザで実行

1.4. 謝辞

SUZKAU で使用しているソフトウェアは Free Software / Open Source Software で構成されています。Free Software / Open Source Software は世界中の多くの開発者の成果によって成り立っています。この場を借りて感謝の意を示します。

1.5. 注意事項

本製品に含まれるソフトウェア(付属のドキュメント等も含みます)は、現状のまま(AS IS)提供されるものであり、特定の目的に適合することや、その信頼性、正確性を保証するものではありません。また、本製品の使用による結果についてもなんら保証するものではありません。

2.作業の前に

2.1.準備するもの

SUZAKU を使用する前に、以下のものを準備してください。

- 作業用 PC
  Linux もしくは Windows が動作し、1 ポート以上のシリアルポートを持つ PC です。

- シリアルクロスケーブル
  D-Sub9 ピン（メスーメス）の「クロス接続用」ケーブルです。

- D-Sub 9 ピン-10 ピン変換ケーブル
  D-Sub9 ピンと SUZAKU のピンヘッダ(10 ピン)を接続するケーブルです。

- 付属 CD-ROM（以降、付属 CD）
  SUZAKU に関する各種マニュアルやソースコードが収納されています。

- シリアルコンソールソフト
  minicom や Tera Term などのシリアルコンソールソフトです。作業用 PC にインストールしてください。

- DC3.3V 電源
  DC3.3V 出力の電源を使用してください。

2.2. 接続方法

下の図を参照して、シリアルクロスケーブル、電源、そして LAN ケーブルを SUZAKU に接続してください。

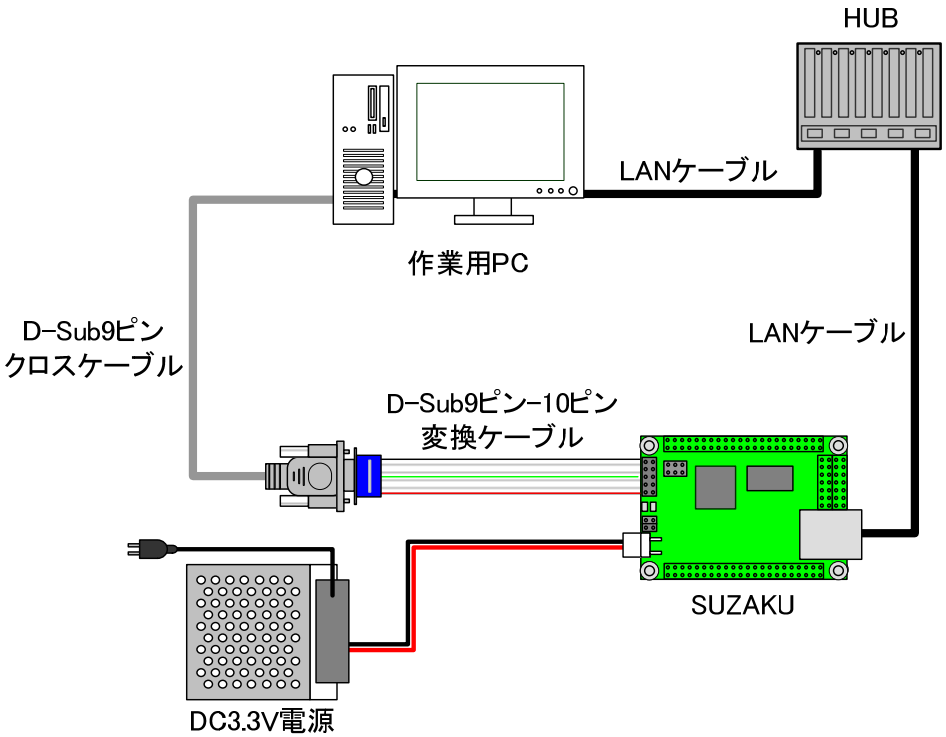

図 2-1 SUZAKU 接続例

2.3. ジャンパピンの設定について

SUZAKU ではジャンパの設定を変えることで、起動時の動作を変更することができます。以下の表に設定と動作の関連を記載します。起動モードについては、「4.2. 起動モード」を参照してください。

表 2-1 ジャンパの設定と起動時の動作

JP1	JP2	起動時の動作	起動モード
オープン	オープン	Linux カーネルを起動	オートブートモード
ショート	オープン	ファーストステージブートローダーを起動	ブートローダーモード
ー	ショート	FPGA コンフィグレーション(※1)	ー

※1 詳しくは、ハードウェアマニュアルを参照してください。

TIPS

ジャンパのオープン、ショートとは

・オープン　：　ジャンパピンにジャンパソケットを挿さない状態
・ショート　：　ジャンパピンにジャンパソケットを挿した状態

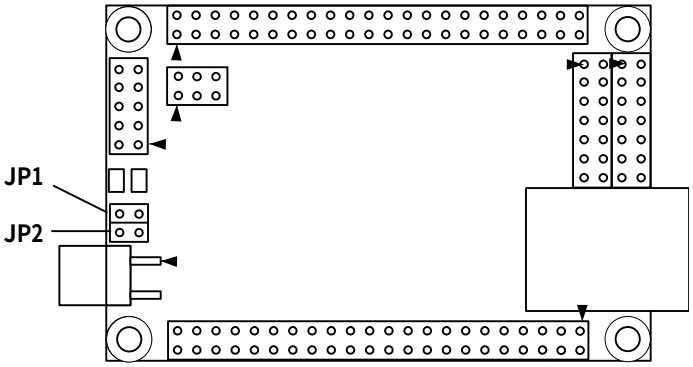

図 2-2 ジャンパの位置

3.開発環境の準備

作業用の PC 上で SUZAKU のソフトウェアをクロス開発することができます。

3.1.クロス開発環境パッケージのインストール

付属 CD の cross-dev ディレクトリにクロス開発環境パッケージが用意されているので、これらを全て作業用 PC にインストールします。インストールは必ず root 権限で行ってください。以下のパッケージが用意されています。

3.1.1. SUZAKU-S

クロス開発環境パッケージは、gzip で圧縮された tar アーカイブ形式になっています。作業用 PC の /usr/local/microblaze-elf-tools/ に展開してください。

```
[PC ~]$ su -
[PC ~]# mkdir -p /usr/local/microblaze-elf-tools/
[PC ~]# cd /usr/local/microblaze-elf-tools/
[PC microblaze-elf-tools]# tar zxvf microblaze-elf-tools-20060213.tar.gz
[PC microblaze-elf-tools]# ls
bin  include  info  lib  libexec  microblaze  share
[PC microblaze-elf-tools]# exit
[PC ~]$
```

図 3-1 開発用パッケージのインストール

次に、クロス開発環境を使いやすくするために、パッケージの実行ファイルが入っているディレクトリを環境変数 PATH に追加します。シェルによって設定方法が異なりますので、詳しくはお使いのシェルのマニュアルを参照してください。

ここでは、bash の設定例を示します。「.bashrc」ファイルに記述しておくと、次回、ログイン時にも有効になります。

```
[PC ~]$ export PATH=$PATH:/usr/local/microblaze-elf-tools/bin
[PC ~]$
```

図 3-2 環境変数 PATH の設定例

3.1.2. SUZAKU-V

クロス開発環境パッケージは付属 CD の cross_dev/powerpc ディレクトリにあります。パッケージファイルは deb(Debian 系ディストリビューション向け)、rpm(Red Hat 系ディストリビューション向け)、tgz(インストーラ非使用)が用意されています。お使いの OS にあわせて、いずれか 1 つを選択してご利用ください。

表 3-1 クロス開発環境パッケージ一覧

パッケージ名	バージョン	説明
binutils-powerpc-linux	2.15-5	The GNU Binary utilities
cpp-3.3-powerpc-linux	3.3.5-13	The GNU C preprocessor
g++-3.3-powerpc-linux	3.3.5-13	The GNU C++ compiler
gcc-3.3-powerpc-linux	3.3.5-13	The GNU C compiler
genromfs	0.5.1-3	The make equivalent for romfs filesystem
libc6-powerpc-cross	2.3.2.ds1-20	GNU C Library: Shared libraries and Timezone data
libc6-dev-powerpc-cross	2.3.2.ds1-20	GNU C Library: Development Libraries and Header Files
libc6-pic-powerpc-cross	2.3.2.ds1-20	GNU C Library: PIC archive library
libc6-prof-powerpc-cross	2.3.2.ds1-20	GNU C Library: Profiling Libraries
libdb1-compat-powerpc-cross	2.1.3-7	The Berkeley database routines
libgcc1-powerpc-cross	3.3.5-13	GCC support library
libstdc++5-powerpc-cross	3.3.5-13	The GNU Standard C++ Library v3
libstdc++5-3.3-dbg-powerpc-cross	3.3.5-13	The GNU Standard C++ Library v3 (debugging files)
libstdc++5-3.3-dev-powerpc-cross	3.3.5-13	The GNU Standard C++ Library v3 (development files)
libstdc++5-3.3-pic-powerpc-cross	3.3.5-13	The GNU Standard C++ Library v3 (shared library subset kit)
linux-kernel-headers-powerpc-cross	2.5.999-test7-bk-16	Linux Kernel Headers for development

クロス開発用パッケージのインストール例を図 3-3 に示します。

```
[PC ~]# dpkg -i binutils-powerpc-linux_2.15-5_i386.deb ←deb パッケージを
                                                        使用する場合
[PC ~]# rpm -i binutils-powerpc-linux-2.15-5_i386.rpm  ←rpm パッケージを
                                                        使用する場合
[PC ~]# tar zxf binutils-powerpc-linux-2.15.tgz -C /      ←tgz を使用する場合
```

図 3-3 クロス開発用パッケージのインストール

インストール時に依存関係でエラーになる場合は、以下のように複数のパッケージを同時に指定してください。

```
[PC ~]# dpkg -i xxx.deb yyy.deb zzz.deb
```

図 3-4 複数パッケージのインストール

3.2. uClinux-dist のビルドに必要なパッケージ

uClinux-distをビルドするためには、作業用PCに表 3-2に記されているパッケージがインストールされている必要があります。作業用PCの環境に合わせて適切にインストールしてください。

表 3-2 uClinux-dist のビルドに必要なパッケージ一覧

パッケージ名	バージョン	説明
libncurses-dev	5.4-4 以降	Developer's libraries and docs for ncurses
zlibg-dev	1.2.2-4 以降	compression library - development
genromfs	0.5.1-3 以降	This is the mkfs equivalent for romfs filesystem
sed	4.1.2-8 以降	The GNU sed stream editor
perl	5.8.4-8 以降	Larry Wall's Practical Extraction and Report Language

4.使用方法

この章では SUZAKU の基本的な使用方法について説明します。

4.1.起動の前に

SUZAKU のシリアルポート 1(CON1)と作業用 PC をシリアルケーブルで接続し、シリアルコンソールソフトを起動します。次のように通信設定を行なってください。

表 4-1 シリアル通信設定

項目	設定
転送レート	115,200bps
データ長	8bit
ストップビット	1bit
パリティ	なし
フロー制御	なし

4.2.起動モード

SUZAKU の起動モードには、オートブートモード、ブートローダーモード、モトローラＳレコード形式ダウンロードモードの３つがあります。各モードの切り換えは、「ジャンパピン (JP1)」および「BBoot メニュー画面」で行います。起動モード選択ジャンパについては、「2.3. ジャンパピンの設定について」を参照してください。

4.2.1. オートブートモード

オートブートモードでは､自動的に Linux が起動されます｡このモードに移行するには､ジャンパピン(JP1, JP2)をオープンにして、SUZAKU に電源を入れます。

4.2.2. ブートローダーモード

ブートローダーモードは、ブートローダーを起動する場合に使用します。出荷時の SUZAKU の Flash メモリには Hermit と呼ばれる高機能なブートローダー兼ダウンローダが入っています。Hermit を起動すると、ユーザからのコマンド入力を受け付けるプロンプトが表示されます。コマンドには、メモリダンプ機能や、Linux イメージを Flash メモリへ書き込む機能などが存在します。

ブートローダーモードに移行するには、ジャンパピン JP1 をショート、JP2 をオープンにして電源を入れます。シリアルに、ファーストステージブートローダーの BBoot の起動ログと BBoot メニューへの移行する指示が表示されます。

```
BBoot v2.x (microblaze)
Press ‘z’ or ‘Z’ for BBoot Menu.
```

図 4-1 BBoot 起動画面

ここで、「z」キーを押し、BBoot メニュー画面へ移動します。

注意

メッセージ（Press ‘z’ or ‘Z’ for BBoot Menu）が表示されてから「z」キーを押した場合、キーの受付に間に合わないことがあります。その際は、「z」キーを押しながら電源を入れるようにしてください。

下図のような BBoot メニュー画面がシリアルコンソールに表示されます。

```
Please choose one of the following and hit enter.
a:  activate second stage bootloader (default)
s:  download a s-record file
```

図 4-2 BBoot メニュー画面

ここで、「a」キーまたは「Enter」キーを押してください。SUZAKU のデフォルトブートローダーである Hermit が起動し、プロンプトが表示されます。

```
Hermit-At v1.1.3 (suzaku/microblaze) compiled at 15:39:32, Aug 10 2006
hermit>
```

図 4-3 Hermit 起動画面

Hermit を使って Flash メモリにデータを書き込む場合には一度 SUZAKU をこのモードで起動しておかなければなりません。Flash メモリの詳しい書き換え方法については「5.2. Hermit を使って Flash メモリを書き換える」を参照してください。

TIPS

なお、BBoot メニュー画面へ移らなかった場合は、自動的に Hermit 起動画面に移行します。

4.2.3. モトローラＳレコード形式ダウンロードモード

このモードは、Flash メモリ上のブートローダーが何らかの理由で動作しなくなった場合に、ブートローダーを再書き込みするために使用します。このモードへの移行は、BBoot メニュー画面より「s」を入力します。BBoot メニュー画面への進み方は、「4.2.2. ブートローダーモード」を参照してください。

BBoot は FPGA の Block RAM と呼ばれる領域に入っていますので、誤って Flash メモリ上の Linux や Hermit の領域を破壊してしまっても FPGA が動作している状態ならば使用できます。BBoot は SUZKAU の起動時に最初に動くプログラムのため、ファーストステージブートローダーと呼ばれます。

モトローラ S レコード形式の詳しい書き込み方法については「5.4. モトローラＳレコード形式を使って Flash メモリを書き換える」を参照してください。

4.3.起動

オートブートモードで電源を入れると、Linux が起動します。正常に起動した場合、シリアルポート 1 に次のようなログが出力されます（SUZAKU-S スターターキット，ディストリビューション: uClinux-dist-20051110-suzaku1)。

```
Linux version 2.4.32-uc0 (atmark@build) (gcc version 3.4.1 ( Xilinx EDK 8.1 Build EDK_I.17 090206 )) #1 Mon Jul 10 19:22:07
JST 2006
On node 0 totalpages: 8192
zone(0): 8192 pages.
zone(1): 0 pages.
zone(2): 0 pages.
CPU: MICROBLAZE
Kernel command line:
Console: xmbserial on UARTLite
Calibrating delay loop... 25.60 BogoMIPS
Memory: 32MB = 32MB total
Memory: 29744KB available (957K code, 1703K data, 44K init)
Dentry cache hash table entries: 4096 (order: 3, 32768 bytes)
Inode cache hash table entries: 2048 (order: 2, 16384 bytes)
Mount cache hash table entries: 512 (order: 0, 4096 bytes)
Buffer cache hash table entries: 1024 (order: 0, 4096 bytes)
Page-cache hash table entries: 8192 (order: 3, 32768 bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.4
Based upon Swansea University Computer Society NET3.039
Initializing RT netlink socket
Microblaze UARTlite serial driver version 1.00
ttyS0 at 0xffff2000 (irq = 1) is a Microblaze UARTlite
Starting kswapd
xgpio #0 at 0xFFFFA000 mapped to 0xFFFFA000
Xilinx GPIO registered
RAMDISK driver initialized: 16 RAM disks of 4096K size 1024 blocksize
eth0: LAN9115 (rev 1150001) at ffe00000 IRQ 2
uclinux[mtd]: RAM probe address=0x80125a30 size=0x174000
uclinux[mtd]: root filesystem index=0
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP: Hash tables configured (established 2048 bind 4096)
VFS: Mounted root (romfs filesystem) readonly.
Freeing init memory: 44K
Mounting proc:
Mounting var:
Populating /var:
Running local start scripts.
Setting hostname:
Setting up interface lo:
Mounting /etc/dhcpc:
Starting DHCP client:
Starting inetd:
Starting thttpd:

SUZAKU-S.STARTER-KIT login:
```

図 4-4 起動ログ

ログインユーザは root で、パスワードの初期設定は root です。

表 4-2 コンソールログイン時のユーザ名とパスワード

ユーザ名	パスワード	権限
root	root	特権ユーザ

4.4. ディレクトリ構成

ディレクトリ構成は次のようになっています。

表 4-3 ディレクトリ構成の一覧

ディレクトリ名	説　明
/bin	アプリケーション用
/dev	デバイスノード用
/etc	システム設定用
/etc/config	flatfsd 向け設定用
/lib	共有ライブラリ用
/mnt	マウントポイント用
/proc	プロセス情報用
/sbin	システム管理コマンド用
/usr	ユーザ共有情報用
/home	ユーザホームディレクトリ
/home/httpd	WEB サーバホームディレクトリ用
/tmp	テンポラリ保存用
/var	変更データ用

4.5. 終了

電源を切断し、SUZAKU を終了します。

4.6. ネットワーク設定

SUZAKU の工場出荷時には、DHCP を使用して IP アドレスを取得する設定になっています。ネットワークの設定を変更する場合は、イメージを再作成する必要があります。「7. uClinux-dist でイメージを作成」および『uClinux-dist Developers Guide』もあわせて参照してください。

4.6.1. ネットワーク設定の確認

ネットワークの設定は ifconfig コマンドで確認することができます。詳しくは、ifconfig コマンドの man を参照してください。DHCP にて IP アドレスの取得に成功した場合は、以下のような結果が表示されます。

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr XX:XX:XX:XX:XX:XX
          inet addr:192.168.1.xx  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST NOTRAILERS RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
                   Interrupt:2 Base address:0x300

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

図 4-5 ifconfig 実行例

4.6.2. 固定 IP アドレスで使用する場合

まず、dhcpcd-new (2.0/2.4)を外します。

```
[PC ~/uClinux-dist]$ make menuconfig
Main Menu
  Kernel/Library/Defaults Selection   --->
    [*] Customize Vendor/User Settings

Main Menu
  Network Applications   --->
    [ ] dhcpcd-new (2.0/2.4)                          ← チェックを外す
```

図 4-6 dhcpcd-new を外す

次に、固定 IP アドレスを設定します。設定は、vendors/AtmarkTechno/SUZAKU-S/etc/rc/ifconfig ファイルを編集します。ifconfig ファイルの上部に記述されている IP_ADDRESS に設定したい IP アドレスを明記します。

```
[PC ~/uClinux-dsit]$ vi vendors/AtmarkTechno/SUZAKU-S/etc/rc/ifconfig
#!/bin/sh

IP_ADDRESS=192.168.10.54                          ← IP アドレスを書き換え

PATH=/bin:/sbin:/usr/bin:/usr/sbin

echo "Setting up interface lo: "
ifconfig lo up 127.0.0.1

echo "Setting up interface eth0: "
ifconfig eth0 $IP_ADDRESS
```

図 4-7 ifconfig ファイルの編集

最後にイメージファイルを再作成し、Flash メモリを書き換えてください。

4.6.3. DHCP を使用する場合

固定 IP アドレスに変更したあと DHCP を使用する設定に戻したい場合には、以下のように DHCP クライアントデーモン（dhcpcd-new）を組み込みます。追加した後は、イメージファイルを再作成し、Flash メモリを書き換えてください。

```
[PC ~/uClinux-dist]$ make menuconfig
Main Menu
  Kernel/Library/Defaults Selection   --->
    [*] Customize Vendor/User Settings

Main Menu
  Network Applications   --->
    [*] dhcpcd-new (2.0/2.4)                      ← チェックを追加
```

図 4-8 dhcpcd-new の追加

4.7. telnet ログイン

次のユーザ名／パスワードで telnet ログインが可能です。

表 4-4 telnet ログイン時のユーザ名とパスワード

ユーザ名	パスワード
root	root

4.8. ファイル転送

ftp によるファイル転送が可能です。次のユーザ／パスワードでログインしてください。ホームディレクトリは「/」です。「/var/tmp」ディレクトリに移動することでデータの書き込みが可能になります。

表 4-5 ftp のユーザ名とパスワード

ユーザ名	パスワード
root	root

4.9. Web サーバ

thttpdという小さなHTTPサーバが起動しており、WebブラウザからSUZAKUをブラウズすることが出来ます。データディレクトリは「/home/httpd」です。URLは「http://(SUZAKUのIPアドレス)/」になります(例 http://192.168.1.10/)。

図 4-9 Web サーバ

5.Flash メモリの書き換え方法

Flash メモリの内容を書き換えることで、SUZAKU の機能を変更することができます。この章では Flash メモリの書き換え方法を説明します。

注意

何らかの原因により「書き換えイメージの転送」に失敗した場合、SUZAKU が正常に起動しなくなる場合があります。書き換えの最中には次の点に注意してください。

- SUZAKU の電源を切らない。
- SUZAKU と開発用 PC を接続しているシリアルケーブルを外さない。

5.1. リージョン指定について

Flash メモリの書き込み先アドレスをリージョン名で指定することができます。リージョン名は３種類あります。それぞれに書き込むイメージとあわせて以下で説明します。

- fpga
  FPGA のコンフィギュレーションデータを格納する領域です。

- bootloader
  ブートローダーと呼ばれる、電源投入後、最初に実行されるソフトウェアのイメージを格納する領域です。ブートローダーは、シリアル経由で Flash メモリを書き換える機能や、OS を起動する機能などを持ちます。

- image
  Linux のカーネルおよびユーザーランドのイメージを格納する領域です。この領域に格納されたカーネルはブートローダーによって起動されます。各アプリケーションを含むシステムイメージを格納する領域です。

付属 CD の image ディレクトリには、各リージョン向けのイメージファイルが格納されています。

表 5-1 各リージョン用のイメージファイル名

リージョン	ファイル名	備考
fpga	fpga-sz###.bin	###には型番が入ります
bootloader	loader-suzaku-microblaze-v#.#.#.bin loader-suzaku-powerpc-v#.#.#.bin	使用する CPU によって、ファイルが異なります #.#.#にはバージョン番号は入ります
image	image-sz###.bin	###には型番が入ります

以下の説明では、ファイル名を簡略化するために、それぞれのリージョン用のファイルを fpga.bin、loader-suzaku.bin、image.bin とします。適時読み替えてください。

Flashメモリのメモリマップは「8.Flashメモリについて」を参照してください。

5.2. Hermit を使って Flash メモリを書き換える

以下の手順で Flash メモリの書き換えを行ないます。作業用 PC から SUZAKU にデータを転送することから、Flash メモリを書き換えることを「ダウンロード」とも言います。

Hermit がインストールされていない場合は、「10.2. ダウンローダ（Hermit）のインストール」を参照してください。

5.2.1. ブートローダーモードで起動する

作業用 PC と SUZAKU のシリアルポート１（CON1）をシリアルケーブルで接続し、SUZAKU のジャンパピンを次のように設定します。

- JP1：ショート
- JP2：オープン

電源を投入後、BBoot メニュー画面からブートローダーモードに移動します。詳しくは、「4.2.2. ブートローダーモード」を参照してください。

5.2.2. ダウンロード

以降の手順は、作業用 PC の OS によって異なります。

注意

Minicom などのシリアル通信ソフトがシリアルポートを使用している状態では、Hermit がシリアルポートを使用できないためダウンロードに失敗します。必ずシリアル通信ソフトを終了（シリアルポートを使用できる状態）してから Hermit を起動してください。

注意

ブートローダー領域に誤ったイメージを書き込んでしまった場合、オンボード Flash メモリからの起動ができなくなります。この場合は「5.4. モトローラ S レコード形式を使って Flash メモリを書き換える」を参照してブートローダーを復旧してください。

1) Linux の場合

Linux が動作する作業用 PC でターミナルを起動し、イメージファイルとリージョンを指定して hermit コマンドを入力します。

下の図では uClinux のイメージ（image.bin）と Hermit のイメージ（loader-suzaku.bin）をダウンロードする例を示しています。-i オプションでファイル名を、-r オプションでリージョン名を指定します。

```
[PC ~]$ hermit download -i fpga.bin -r fpga --force-locked
[PC ~]$ hermit download -i loader-suzaku.bin -r bootloader --force-locked
[PC ~]$ hermit download -i image.bin -r image
```

図 5-1 hermit コマンド入力例

作業用 PC で使用するシリアルポートが「ttyS0」以外の場合、オプション「--port “ポート名”」を追加してください。

TIPS

FPGA およびブートローダー領域を書き換える際は、「--force-locked」を追加する必要があります。これを指定しない場合、警告が表示されブートローダー領域への書き込みは実行されません。

書き換え終了後、JP1、JP2 をオープンに設定して SUZAKU を再起動すると、新たに書き込んだイメージで起動されます。

2) Windows の場合

「10.2.ダウンローダ（Hermit）のインストール」にてファイルを展開したフォルダにある、「Hermit-At WIN32（hermit.exe)」を起動します。

「Download」ボタンをクリックすると Download 画面が表示されます。

"Serial Port" には、SUZAKU と接続しているシリアルポートを設定してください。
"Image" には、書き込むイメージファイルを指定します。ファイルダイアログによる指定も可能です。
"Region" には、書き込むリージョンまたは、アドレスを指定します。

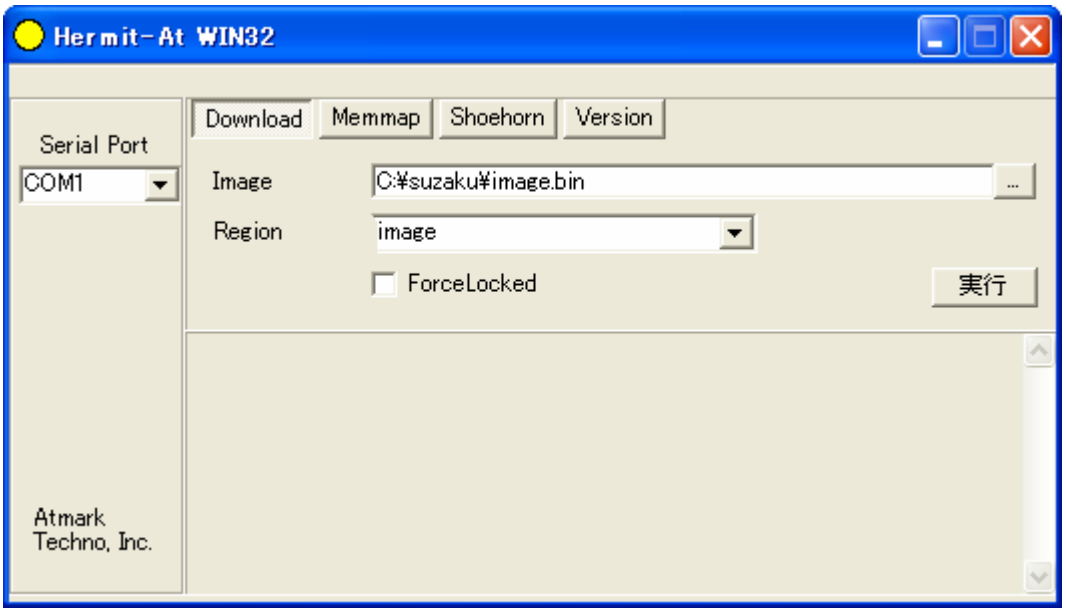

図 5-2 Download 画面

「実行」ボタンをクリックすると、Flash メモリの Download が開始されます。Download 中は、進捗状況が下図のように表示されます。ダイアログは、Download が終了すると自動的にクローズし、図 5-4 のような Download 終了画面が表示されます。

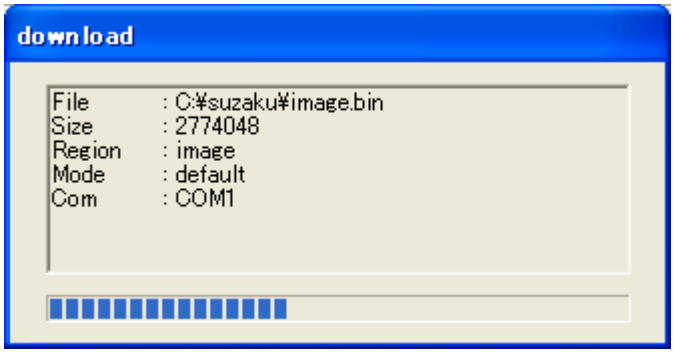

図 5-3 Download 進捗ダイアログ

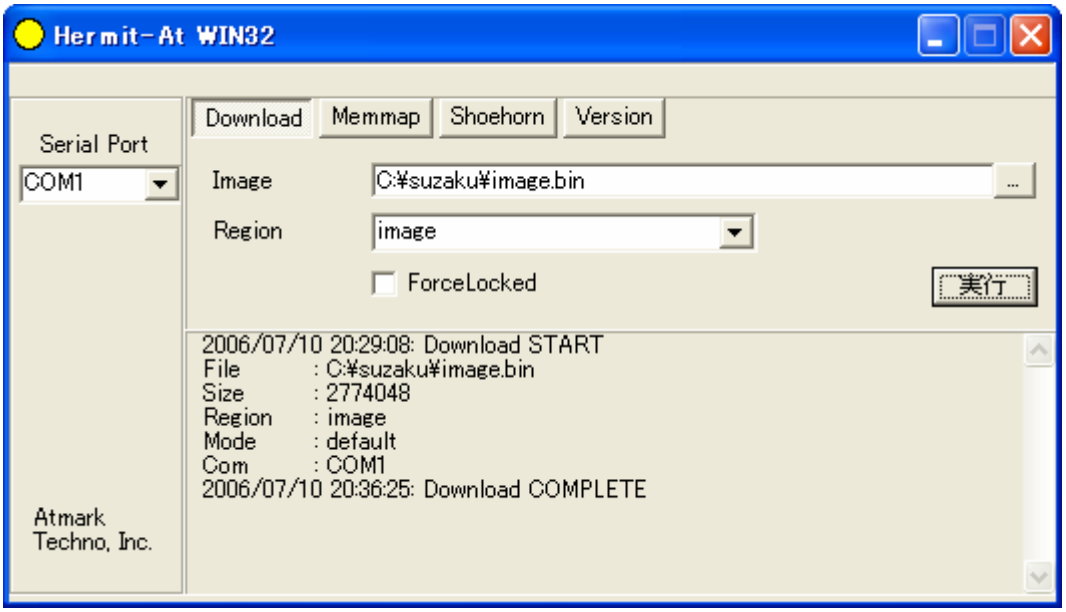

図 5-4 Download 終了画面

TIPS

FPGA およびブートローダー領域を書き換える際は、「ForceLocked」をチェックする必要があります。これを選択しない場合、警告が表示されブートローダー領域への書き込みは実行されません。

書き換え終了後、JP1、JP2 をオープンに設定して SUZAKU を再起動すると、新たに書き込んだイメージで起動します。

5.3. netflash を使って Flash メモリを書き換える

Flash メモリの内容を書き換える方法として、ユーザアプリケーションの netflash を使用することも可能です。netflash を使用して Flash メモリを書き換える方法を説明します。

netflash は、HTTP や FTP サーバからファイルを取得し、Flash メモリへ書き込みます。はじめに、HTTP や FTP サーバにイメージファイルを置いておく必要があります。

SUZAKU 上でのイメージ（image.bin）を変更するコマンド例です。

```
[SUZAKU ]# netflash -r image http://local.server.name/suzaku/image.bin
```

図 5-5 netflash 実行例

オプションの “-r /dev/flash/image” でリージョンを指定します。リージョンの指定は下記表を参照してください。

表 5-2 各リージョン用 Flash デバイス名

リージョン	Flash デバイス名
fpga	/dev/flash/fpga
image	/dev/flash/image
config	/dev/flash/config

netflash のヘルプは以下のコマンドで参照することができます。

```
[SUZAKU ~]# netflash -h
```

図 5-6 netflash ヘルプコマンド

注意

SZ130-U00 では、現在のディストリビューション（uClinux-dist-20051110-suzaku2）において、netflash には対応しておりません。ディストリビューションの更新をお待ちください。

5.4. モトローラ S レコード形式を使って Flash メモリを書き換える

モトローラ S レコード形式でのダウンロードは、ソフトウェアで SUZAKU の Flash メモリを更新する最終手段です。この方法は Hermit や Linux が起動しなくても使えます。ここでは、minicom を使ってモトローラ S レコード形式の Hermit を SUZAKU にダウンロードする方法を説明します。

モトローラ S レコード形式の Hermit は付属 CD の bootloader/s-record ディレクトリに loader-suzaku-microblaze-v1.x.x-4M.srec （SZ010-U00）、loader-suzaku-microblaze-v1.x.x-8M.srec（SZ030-U00, SZ130-U00）、loader-suzaku-powerpc-v1.x.x-8M.srec（SZ310-U00）という名前で入っています。このファイルがホームディレクトリにコピーされていると仮定します。以降、4MB の Flash メモリを例に話を進めます。他の場合は適時読み替えてください。

注意

Flash メモリのサイズによって、Hermit が書き込まれるアドレスが異なります。モトローラSレコード形式ではファイル内に書き込みアドレスが書かれているために、同じ SUZAKU-S でも Flash メモリのサイズによってファイルを分けてあります。

まず、シリアルクロスケーブルで SUZAKU と作業用 PC を接続し、作業用 PC 上で minicom を起動します。次に SUZAKU の起動モードジャンパ(JP1)をショートして SUZAKU の電源を入れ、BBoot メニュー画面に移行します。これで minicom の画面に起動モード選択画面が表示されます。

BBoot メニュー画面で「s」を入力すると以下のようなメッセージが表示されます。

```
Start sending S-Record!
```

図 5-7 モトローラ S レコード形式ダウンロード開始画面

このメッセージの後、SUZAKU はモトローラ S レコード形式のファイルを待ちます。

minicom のメニュー画面から「Send files」を選択します。アップロードサブメニューの中から ASCII を選択し、ファイル選択画面に移動します。ファイル選択画面で bootloader-suzaku.srec を選択して転送を開始します。転送中は minicom の中に小さな画面が表示されます。転送が終了すると

```
Erasing SPI ...
Programming SPI ...
done.
Reboot.
```

図 5-8 モトローラ S レコード形式ダウンロード終了画面

と表示され、SUZAKU が再起動されます。再起動した SUZAKU でブートローダーモードを選択すると Hermit が起動されます。

6. ブートローダー

この章では、SUZAKU のブートローダーとブートシーケンスについて説明します。

6.1. ブートローダーの種類

SUZAKU で用意されているブートローダーは、以下のとおりです。

表 6-1 各ブートローダー一覧

ブートローダー名	説明
BBoot	ファーストステージ・ブートローダー
Hermit	セカンドステージ・ブートローダー

6.2. ブートシーケンス

6.2.1. 第１ステージ（BBoot）

リセット時にプログラム・カウンタが 0 に初期化されます。SUZAKU では 0 番地に FPGA の内部メモリがマップされており、BBoot という第１ステージ・ブートローダーが格納されています。

BBoot は第２ステージ･ブートローダーを起動する役目と､モトローラ S レコード形式のファイルを Flash メモリにダウンロードする機能を持っています。FPGA の内部メモリは非常に高価なリソースなことから、できるだけ小さなプログラムにし、汎用的な高機能ブートローダーを Flash メモリに置いています。

BBoot は SUZAKU の起動モード選択ジャンパを調べ、BBoot メニュー画面を表示し、モトローラ S レコード形式のダウンロードを行うのか、または第２ステージ・ブートローダーを起動するのかを、ユーザから受け付け実行します。

6.2.2. 第２ステージ（Hermit）

第２ステージ・ブートローダーには Hermit を使用しています。Hermit は ARM CPU ボード Armadillo にも採用している高機能ブートローダー兼ダウンローダーです。

Hermit の一番の役割はカーネルをブートすることです。Hermit が起動モード選択ジャンパを調べ、カーネルをブートするのかブートローダーモードで起動するのかを判断します。オートブートモードの場合は、カーネルイメージを Flash メモリから SDRAM にコピーしてカーネルに制御を渡します。

Hermit をブートローダーモードで起動した場合（BBoot メニュー画面から選択可能)、Hermit は独自のバイナリ転送方式で uClinux のイメージを Flash メモリに書き込みます。

6.2.3. 第３ステージ（カーネル）

カーネルはブートローダーから制御を渡された後、システムの初期化を行います。システムの初期化の多くは一般的な Linux と同じです。スケジューリングに必要なタイマの初期化や割り込みベクタの初期化、メモリ・マネージメント・サブシステムによる有効な RAM 領域の初期化などが行われます。そして、カーネルは最後にルート・ファイル・システムをマウントして、ユーザ・ランドに制御を渡します。

uClinux 特有の機能として、カーネル・イメージの最後尾にルート・ファイル・システムを結合することができます。SUZAKU では、ROM の使用量を抑えるために、ルート・ファイル・システムはカーネルの

BSS セクションを上書きするように結合しています。カーネルは初期化時にルート・ファイル・システムのイメージを発見すると、BSS セクションを 0 で初期化します。

6.2.4. 第４ステージ（ユーザーランド）

カーネルはデフォルトで最初に/sbin/init を実行します。/sbin/init は/etc/inittab に従ってシリアル・コンソールからのログイン用に getty を起動したり、システムが機能するための設定やデーモンの起動を行います。使用している/sbin/init の実装はデスクトップやサーバ用のディストリビューションとは異なりますが、シェル・スクリプトを順次実行するという基本的な動作は同じです。

7.uClinux-dist でイメージを作成

この章では、uClinux-dist を使用して、カーネル／ユーザーランドのイメージを作成する方法を説明します。uClinux-dist に関する詳しい使用方法は、『uClinux-dist Developers Guide』を参照してください。

注意

uClinux-dist を使用した開発作業では、基本ライブラリ・アプリケーションやシステム設定ファイルの作成・配置を行ないます。各ファイルは uClinux-dist ディレクトリ配下で作成・配置作業を行ないますが、作業ミスにより誤って作業用 PC 自体の OS を破壊しないために、すべての作業は root ユーザではなく一般ユーザで行なってください。

7.1. ソースコードアーカイブの展開

付属 CD の dist ディレクトリに uClinux-dist-YYYYMMDD-suzakuX.tar.gz というファイル名のソースコードアーカイブがあります。このファイルを任意のディレクトリに展開します。ここでは、ユーザのホームディレクトリ(~/)に展開することとします。

```
[PC ~]$ tar zxvf uClinux-dist-YYYYMMDD-suzakuX.tar.gz
```

図 7-1 dist アーカイブの展開

7.2. 設定

ターゲットボード用の dist をコンフィギュレーションします。以下の例のようにコマンドを入力し、コンフィギュレーションを開始します。

```
[PC ~/uClinux-dist-YYYYMMDD-suzakuX]$ make config
```

図 7-2 dist のコンフィギュレーション

続いて、使用するボードのベンダー名を聞かれます。「AtmarkTechno」と入力してください。

```
*
* Vendor/Product Selection
*
*
* Select the Vendor you wish to target
*
Vendor (3com, ADI, Akizuki, Apple, Arcturus, Arnewsh, AtmarkTechno, Atmel, Avnet,
Cirrus, Cogent, Conexant, Cwlinux, CyberGuard, Cytek, EMAC, EPSD, Exys, Feith, Future,
GDB, Hitachi, Imt, Insight, Intel, KendinMicrel, LEOX, Mecel, Midas, Motorola, NEC,
NetSilicon, Netburner, Nintendo, OPENcores, OpenGear, Philips, Promise, SNEHA, SSV,
SWARM, Samsung, SecureEdge, Signal, SnapGear, Soekris, Sony, StrawberryLinux, TI,
TeleIP, Triscend, Triscend, Via, Weiss, Xilinx, senTec) [SnapGear] (NEW) AtmarkTechno
```

図 7-3 Vendor の選択

次にボード名を聞かれます。SUZAKU-Sシリーズの場合は「**SUZAKU-S**」と、なかでもSUZAKU-Sスターターキットをお使いの方は「**SUZAKU-S.STARTER-KIT**」、SUZAKU-Vシリーズの場合は「**SUZAKU-V**」と入力してください。

```
*
* Select the Product you wish to target
*
AtmarkTechno Products (SUZAKU-S, SUZAKU-S.STARTER-KIT, SUZAKU-UQ-XIP, SUZAKU-V)
[SUZAKU-S] (NEW) SUZAKU-S.STARTER-KIT
```

図 7-4 Product の選択

使用する C ライブラリを指定します。使用するボードによってサポートされているライブラリは異なります。SUZAKU では、「**uClibc**」を選択します。

```
*
* Kernel/Library/Defaults Selection
*
*
* Kernel is linux-2.4.x
*
Libc Version (None, glibc, uC-libc, uClibc) [uClibc] (NEW) uClibc
```

図 7-5 Library の選択

デフォルトの設定にするかどうか質問されます。「**y**」(Yes)を選択してください。

```
Default all settings (lose changes) (CONFIG_DEFAULTS_OVERRIDE) [N/y/?] (NEW) y
```

図 7-6 デフォルト設定の選択

最後の３つの質問は全て「**n**」(No)と答えてください。

```
Customize Kernel Settings (CONFIG_DEFAULTS_KERNEL) [N/y/?] n
Customize Vendor/User Settings (CONFIG_DEFAULTS_VENDOR) [N/y/?] n
Update Default Vendor Settings (CONFIG_DEFAULTS_VENDOR_UPDATE) [N/y/?] n
```

図 7-7 Customize および Update の選択

質問事項が終わるとビルドシステムの設定を行ないます。すべての設定が終わるとプロンプトに戻ります。

7.3. ビルド

実際にビルドするには以下のコマンドを入力してください。

```
[PC ~/uClinux-dist-YYYYMMDD-suzakuX]$ make dep all
```

図 7-8 ビルド

dist のバージョンによっては、make の途中で一時停止し、未設定項目の問合せが表示される場合があります。通常はデフォルト設定のままで構いませんので、そのような場合はそのままリターンキーを入力して進めてください。

ビルドが終了すると、uClinux-dist/images ディレクトリに、カーネルとユーザランドを合わせたイメージファイル（image.bin）が作成されます。作成したイメージファイルをSUZAKUに書き込む方法は「5.Flash メモリの書き換え方法」を参照してください。

TIPS

dep ターゲットは依存関係の解決を行います。2.4 系までの Linux カーネルのビルドシステムでは、make の前に依存関係の解決を行わなければなりません。2.6 系では必要ありません。

8.Flash メモリについて

SUZAKU の Flash メモリは、リージョンと呼ばれる複数の領域に分割されています。Flash メモリのリージョンの区分は、製品毎に異なります。

SUZAKU-S スターターキットで使われている SZ130-U00 の Flash メモリマップは、表 8-1 になります。

表 8-1 Flash メモリマップ (SZ130-U00, Flash:8MB)

アドレス	リージョン		サイズ	説明
0x00000000 0x000FFFFF	fpga		1MB	
0x00100000 0x0011FFFF	bootloader		128KB	Hermit ブートローダー
0x00120000 0x0041FFFF	image	kernel	3MB	Linux カーネル
0x00420000 0x007EFFFF		user	約 3.81MB	ユーザーランド
0x007F0000 0x007FFFFF	config		64KB	コンフィグ領域

SUZAKU-S シリーズの SZ010-U00 の Flash メモリマップは、表 8-2 です。

表 8-2 Flash メモリマップ (4MB)

アドレス	リージョン		サイズ	説明
0x00000000 0x0007FFFF	fpga		512KB	
0x00080000 0x0009FFFF	bootloader		128KB	Hermit ブートローダー
0x000A0000 0x0020FFFF	image	kernel	約 1.44MB	Linux カーネル
0x00210000 0x003EFFFF		user	1.875MB	ユーザーランド
0x003F0000 0x003FFFFF	config		64KB	コンフィグ領域

SUZAKU-S シリーズの SZ030-U00 や SUZAKU-V シリーズの SZ310-U00 の Flash メモリマップは、表 8-2 のとおりです。

表 8-3 Flash メモリマップ (8MB)

アドレス	リージョン		サイズ	説明
0x00000000 0x0000FFFF	free1		64KB	
0x00010000 0x0007FFFF	free2		448KB	
0x00080000 0x000FFFFF	fpga		512KB	
0x00100000 0x0011FFFF	bootloader		128KB	Hermit ブートローダー
0x00120000 0x0041FFFF	image	kernel	3MB	Linux カーネル
0x00420000 0x007EFFFF		user	約 3.81MB	ユーザーランド
0x007F0000 0x007FFFFF	config		64KB	コンフィグ領域

注意

fpga のリージョンは、FPGA のコンフィグレーションデータを格納しています。この領域に不適切なデータを書き込んだ場合、SUZAKU の異常動作により SUZAKU 及び周辺機器が発熱、劣化、破損する可能性があります。正常に動作させるためには、FPGA コンフィギュレーションデータの再プログラミングが必要になります。詳しくは Hardware Manual を参照してください。

9. Flat Binary Format

SUZAKU に MMU を持たないソフトコアプロセッサ Microblaze を搭載したときには、MMU が必要な一般的な Linux は動作しません。このため出荷状態の SUZAKU-S では uClinux を利用しています。

uClinux が対象としている組み込み機器では実行ファイルのサイズは大きな問題です。一般の Linux が採用している ELF は柔軟性に富んだフォーマットですが、サイズが大きくなってしまいます。そこで uClinux では昔ながらの a.out フォーマットに似た新たなバイナリーフォーマットを採用しています。この章では、その一つである Flat Binary Format について説明します。

はじめに Flat Binary Format の特徴を説明した後、実行ファイルの圧縮方法とスタックサイズの変更方法を説明します。

9.1. Flat Binary Format の特徴

Flat Binary Format には次のような特徴があります。

- シンプル
  シンプルな設計はバイナリファイルの実行速度と大きさに貢献しています。もちろん ELF に比べ柔軟性は劣りますが、組み込み機器にとっては必要なトレードオフと言えます。

- 圧縮可能
  Flat Binary Format は圧縮可能なフォーマットです。圧縮には 2 種類あり、ファイル全体の圧縮とデータ領域のみの圧縮が可能です。実行ファイルはロードされる時に伸長されるため、起動速度は非圧縮の実行ファイルに比べ低速です。起動後は非圧縮の実行ファイルとの差はなく、常駐プロセスのように起動停止を繰り返さないプログラムには適しています。

- スタックサイズフィールド
  再コンパイルせずに変更可能なスタックサイズのフィールドを持っています。MMU を持たない CPU では動的にスタック領域を拡張することが難しいため、固定サイズのスタック領域を持ちます。このフィールドは flthdr と呼ばれるツールで変更することが可能です。また、コンパイル時にサイズを指定することも可能です。

- XIP 対応
  Flat Binary Format は XIP にも対応しています。XIP とは eXecute In Place（その場実行）の略で、一般的には RAM に実行バイナリをコピーすることなく、保存されている ROM 上で実行する機能のことです。

9.2. 実行ファイルの圧縮

例として『uCliux-dist Developers Guide』で作成した Hello World プログラムを圧縮します。後半ではコンパイル時に圧縮を指定する方法を説明します。

注意

デフォルトのカーネルでは、圧縮された Flat Binary Format (ZFLAT) のファイルを実行できません。カーネルを ZFLAT 対応にするには、「9.4. ZFLAT 対応カーネルを作る」を参照してください。

9.2.1. コンパイル済バイナリファイルを圧縮

flthdr を使ってコンパイル済のバイナリファイルを圧縮します。flthdr は Flat Binary Format のファイルを編集・表示するプログラムです。Microblaze ツールチェーンには mb-flthdr という名前で存在していますので、以降 mb-flthdr として説明します。

通常のコンパイルで生成された実行ファイルでは以下のようになります。

```
[PC ~/hello]$ make
[PC ~/hello]$ mb-flthdr hello                                  --- ②
hello
    Magic:        bFLT
    Rev:          4
    Build Date:   Fri Jun  30 18:33:00 2006
    Entry:        0x50
    Data Start:   0x4f40
    Data End:     0x5cf0
    BSS End:      0x7d10
    Stack Size:   0x1000
    Reloc Start:  0x5cf0
    Reloc Count:  0x51
    Flags:        0x1 ( Load-to-Ram )
[PC ~/hello]$
```

図 9-1 通常の Flat Binay Format

次に mb-flthdr で圧縮します。

```
[PC ~/hello]$ mb-flthdr -z hello                               --- ①
zflat hello --> hello
[PC ~/hello]$ mb-flthdr hello                                  --- ②
    Magic:        bFLT
    Rev:          4
    Build Date:   Fri Jun  30 18:35:00 2006
    Entry:        0x50
    Data Start:   0x4f40
    Data End:     0x5cf0
    BSS End:      0x7d10
    Stack Size:   0x1000
    Reloc Start:  0x5cf0
    Reloc Count:  0x51
    Flags:        0x5 ( Load-to-Ram Gzip-Compressed )          --- ③
[PC ~/hello]$
```

① mb-flthdr に圧縮オプション'-z'を渡す
② mb-flthdr コマンドで実行ファイルのヘッダ部を表示させる
③ Gzip-Compressed フラグが確認できる

図 9-2 圧縮された Flat Binary Format

9.3. スタックサイズの指定

スタックサイズを指定する方法を２種類紹介します。

9.3.1. コンパイル済バイナリファイルスタックサイズを変更

mb-flthdr の -s でスタックサイズを指定します。アーキテクチャによりますが、スタックサイズのデフォルト値は 4096（0x1000）が多いです。

```
[PC ~/hello]$ mb-flthdr -s 8192 hello                    --- ①
[PC ~/hello]$ mb-flthdr hello                            --- ②
hello
    Magic:        bFLT
    Rev:          4
    Build Date:   Fri Jun  30 18:34:00 2006
    Entry:        0x50
    Data Start:   0x4f40
    Data End:     0x5cf0
    BSS End:      0x7d10
    Stack Size:   0x2000                                 --- ③
    Reloc Start:  0x5cf0
    Reloc Count:  0x51
    Flags:        0x1 ( Load-to-Ram )
[PC ~/hello]$
```

① mb-flthdr にスタックサイズ変更オプション'-s'とスタックサイズ（10 進数)を渡す
② mb-flthdr コマンドで生成された実行ファイルのヘッダ部を表示させる
③ 8192byte に変更されているのが確認できる

図 9-3 スタックサイズの変更

9.3.2. コンパイル時にスタックサイズを指定

コンパイル時にスタックサイズを指定するには、FLTFLAGS 環境変数を使います。以下に Hello World での例を示します。

```
[PC ~/hello]$ make FLTFLAGS=' -s 8192'                   --- ①
[PC ~/hello]$ mb-flthdr hello                            --- ②
hello
    Magic:        bFLT
    Rev:          4
    Build Date:   Fri Jun  30 18:34:00 2006
    Entry:        0x50
    Data Start:   0x4f40
    Data End:     0x5cf0
    BSS End:      0x7d10
    Stack Size:   0x2000                                 --- ③
    Reloc Start:  0x5cf0
    Reloc Count:  0x51
    Flags:        0x1 ( Load-to-Ram )
 [PC ~/hello]$
```

① FLTFLAGS 環境変数に‘-s 8192’をセットして make を実行する
② mb-flthdr コマンドで生成された実行ファイルのヘッダ部を表示させる
③ 8192byte に変更されたスタックが確認できる

図 9-4 FLTFLAGS によるスタックサイズの指定

9.4. ZFLAT 対応カーネルを作る

ZFLAT とは圧縮された Flat Binary Format の実行ファイルです。実際に ZFLAT に対応したカーネルを作ってみます。

uClinux-dist のディレクトリで、make menuconfig を実行し、「Kernel/Library/Defaults Selection」メニューから「Customize Kernel Settings」を選択します。メインメニューから Exit を選択し、設定を保存し終了します。

「Linux Kernel Configuration」のメインメニューの中から、「General setup」に移動します。「Kernel support for flat binaries」の下にある「Enable ZFLAT support」を選択します。メインメニューから Exit を選択し、設定を保存し終了します。

make clean && make dep && make を実行し、image.bin を生成します。生成されたイメージのカーネルでは ZFLAT ファイルを実行することができます。

10. Appendix

10.1. シリアルコンソールソフト（minicom）のインストール

作業用 PC に「シリアルコンソールソフト（minicom）」をインストールします。

付属 CD よりパッケージファイルを用意し、インストールします。必ず root 権限で行ってください。
パッケージファイルは deb（Debian 系ディストリビューション向け）、rpm（Red Hat 系ディストリビューション向け）が用意されています。お使いの OS にあわせて、いずれか１つを選択してご利用ください。

```
[PC ~]# dpkg -i minicom_2.1-4.woody.1_i386.deb        ←deb パッケージを使用する場合

[PC ~]# prm -i minicom_2.1-1-rh7.3.i386.prm           ←rpm パッケージを使用する場合
```

図 10-1 minicom のインストール

minicom に-s オプションを付けて起動します。-s を指定することで、設定画面に移行します。シリアルポートの通信設定を行ってください。

TIPS

minicom の初期設定では、起動時にモデムの初期化を行うようになっていることが多いようです。設定で初期化用の AT コマンドを外すか、minicom に-o オプションを付けて起動することでモデム初期化コマンドを省略することができます。
また、使用するシリアルポートの読み込みと書き込み権限が無い場合、minicom の起動に失敗します。使用するシリアルポートの権限を確認してください。詳しくは minicom のマニュアルまたはご使用の OS のマニュアルをご覧ください。

10.2. ダウンローダ（Hermit）のインストール

作業用 PC に「ダウンローダ(hermit)」をインストールします。ダウンローダは SUZAKU の Flash メモリの書き換えに使用します。

1) Linux の場合
付属 CD よりパッケージファイルを用意し、インストールします。必ず root 権限で行ってください。
パッケージファイルは deb(Debian 系ディストリビューション向け)、rpm(Red Hat 系ディストリビューション向け)が用意されています。お使いの OS にあわせて、いずれかを選択してください。また、tar.gz（インストーラ非使用、要コンパイル）が用意されています。

```
[PC ~]# dpkg -i hermit-at_x.x.x_i386.deb              ←deb パッケージを使用する場合

[PC ~]# rpm -i hermit-at-x.x.x-x.i386.rpm             ←rpm パッケージを使用する場合

[PC ~]$ tar zxf hermit-at-x.x.x-source.tar.gz -C /    ←tar.gz を使用する場合
```

図 10-2 Hermit のインストール

2) Windows の場合
付属 CD より「Hermit-At WIN32 (hermit-at-win-vx.x.xx.zip)」を任意のフォルダに展開します。

10.3. Windows 上に開発環境を構築する方法

Linux環境を実現するcoLinux(http://www.colinux.org/)を利用することで、Windows上にクロス開発環境を構築することができます。対応しているOSはWindowsXP、Windows2000 です。

10.3.1. coLinux のインストール

1) 付属 CD の colinux ディレクトリにある coLinux-0.6.4.exe を実行する
2) インストール先フォルダには c:¥coLinux を指定し、それ以外はデフォルトの設定のままでインストール作業を行なう

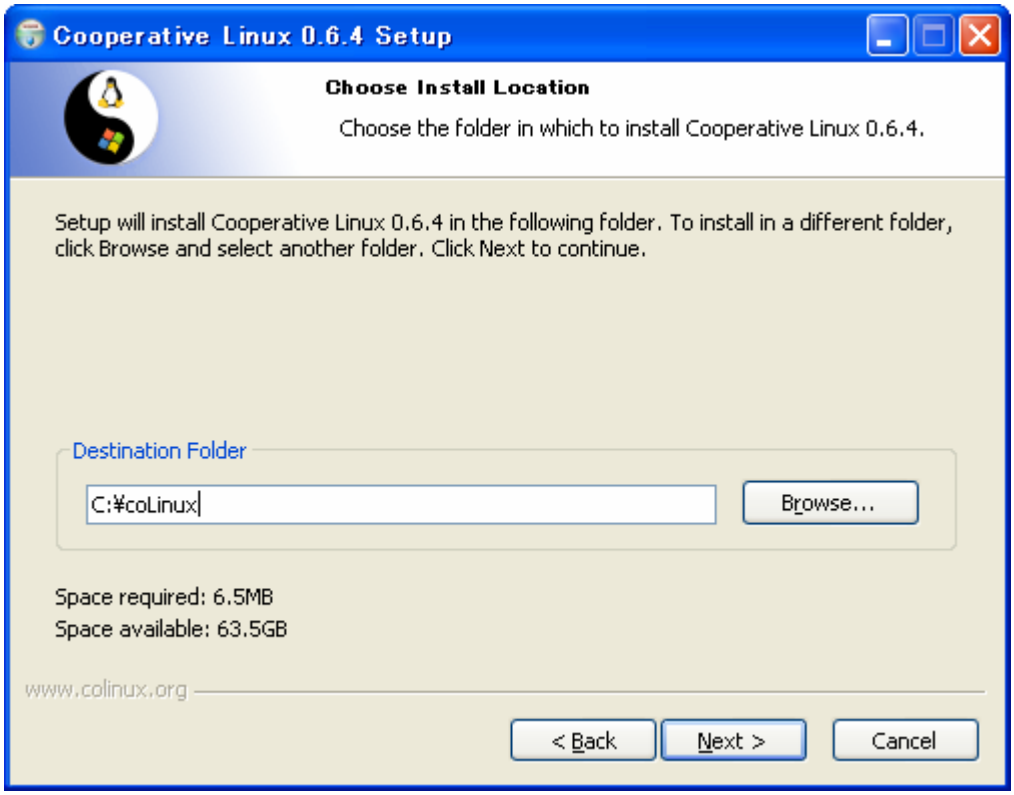

図 10-3 インストール先フォルダの指定

TIPS
インストール先に他のディレクトリを指定する場合は、次の手順で用意する default.colinux.xml を編集し、ディレクトリ名を適切に変更する必要があります。

10.3.2. 環境構築用ファイルの準備

付属 CD の colinux ディレクトリから以下のファイルを用意し、coLinux のインストールフォルダ(c:¥coLinux)に展開します

- root_fs.zip (ルートファイルシステム)
- swap_device_256M.zip (swap ファイルシステム)
- home_fs_2G.zip (/home にマウントされるファイルシステム)
- default.colinux.xml.zip (デバイス情報の設定ファイル)

TIPS

swap_device_..., home_fs_... のファイル名の数値は展開後のファイルサイズです。他のサイズのファイルも用意していますので、必要と思われるサイズのファイルを展開してください。

展開ソフトによっては展開に失敗する場合があります。WindowsXP の標準機能で正常に展開できることを確認してあります。

10.3.3. coLinux の実行

1) DOS プロンプトを起動し、インストールフォルダ(c:¥coLinux)に移動する
2) 「colinux-daemon.exe -c default.colinux.xml」とコマンド入力する
3) 起動ログの表示後「colinux login:」と表示されたら「root」でログインする

TIPS

colinux の実行時に、青画面(DRIVER_IRQL_NOT_LESS_OR_EQUAL)になった時は、c:¥boot.ini の”/noexecute=option”を”/noexecute=AlwaysOff”と書き換えてみてください。

10.3.4. ネットワークの設定

coLinux では Windows とは別の IP アドレスを持ち、Windows を介してネットワークにアクセスするため、Windows のネットワーク設定の変更が必要となります。

設定方法には「ルーター接続」「ブリッジ接続」などがありますが、ここでは「ルーター接続」の方法を説明します。

(WindowsXP の場合)

1) コントロールパネルから「ネットワーク接続」を開く
2) 外部に接続しているネットワークを右クリックして「プロパティ」を開く
3) 「詳細設定」タブを開き、インターネット接続の共有を有効にする

(Windows2000 の場合)

1) コントロールパネルから「ネットワークとダイアルアップ接続」を開く
2) 外部に接続しているネットワークを右クリックして「プロパティ」を開く
3) 「共有」タブを開き、インターネット接続の共有を有効にする

次に、ネットワークの設定を有効にするためのコマンドを coLinux 上で実行します。

```
colinux:~# /etc/init.d/networking restart
Reconfiguring network interfaces: done.
colinux:~#
```

図 10-4 ネットワークの設定コマンド

「ルーター接続」では 192.168.0.0/24 のネットワークアドレスが自動的に使用されるため、外部接続用のネットワークアドレスが同じ 192.168.0.0/24 の場合、設定に失敗します。この場合は外部接続用のネットワークアドレスを変更してください。
外部接続用のネットワークアドレスを変更できない場合は「10.3.8.特殊な場合の Windowsネットワーク設定方法」を参照してください。

10.3.5. coLinux ユーザの作成

coLinux の画面で以下のようにコマンドを入力し作業用ユーザを作成します。適宜パスワードなどを設定してください。

```
colinux:~# adduser somebody
Adding user somebody...
Adding new group somebody (1000).
Adding new user somebody (1000) with group somebody.
Creating home directory /home/somebody.
Copying files from /etc/skel
Enter new UNIX password:
```

図 10-5 ユーザ「somebody」を作業用ユーザとして追加する場合

10.3.6. Windows-coLinux 間のファイル共有

Windows の共有フォルダを利用して、coLinux と Windows 間でファイルを交換する方法です。coLinux の画面で以下のように smbmount コマンドを実行して、共有フォルダのパスワードを入力してください。

```
colinux:~# mkdir /mnt/smb
colinux:~# smbmount //192.168.0.100/shared /mnt/smb
212: session request to 192.168.0.100 failed (Called name not present)
212: session request to 192 failed (Called name not present)
Password:
```

図 10-6 Windows の IP アドレス:192.168.0.100、共有フォルダ名：shared の場合

ユーザ名が Windows 側と異なる場合は、ユーザ名をコマンドのオプションで指定します。詳しくは「man smbmount」を実行してヘルプを参照してください。
以後、windows の共有フォルダ”shared”と coLinux のディレクトリ”/mnt/smb” のデータは同じものになります。

10.3.7. クロス開発環境の導入

「3.開発環境の準備」を参照して、クロス開発環境をcoLinux上に導入してください。
環境の構築に必要なファイルは､前項で使用した共有フォルダを通じて coLinux からアクセスできます。

これで Windows 上で開発を行なうことができます。以降の説明は特殊なケースです。

10.3.8. 特殊な場合の Windows ネットワーク設定方法

外部接続用のネットワークアドレスが 192.168.0.0/24 の場合のネットワーク設定方法です。

(WindowsXP の場合)

「ブリッジ接続」を利用する方法です。

1) コントロールパネルから「ネットワーク接続」を開く
2) 外部に接続しているネットワークと「TAP-Win32 adapter」というデバイス名のネットワークの二つを選択状態にする
3) メニューの「詳細設定」から「ブリッジ接続」を選択する

(Windows2000 の場合)

Windows2000 では 192.168.0.0/24 以外のネットワークアドレスをプライベートネットワークで使用する方法です。ここでは 192.168.1.0/24 を使用します。

1) コントロールパネルから「ネットワークとダイアルアップ接続」を開く
2) 外部に接続しているネットワークを右クリックして「無効」にする
3) 外部に接続しているネットワークを右クリックして「プロパティ」を開く
4) 「全般」タブの「インターネットプロトコル(TCP/IP)」を選択し「プロパティ」ボタンを押す
5) 「次の IP アドレスを使う」を選択して 192.168.100.100 を設定する
6) 「共有」タブを開き、インターネット接続の共有を有効にする
7) 「TAP-Win32 adapter」というデバイス名のネットワークを右クリックして「プロパティ」を開く
8) 「全般」タブの「インターネットプロトコル(TCP/IP)」を選択し「プロパティ」ボタンを押す
9) 「次の IP アドレスを使う」を選択して 192.168.1.1 を設定する
10) 外部に接続しているネットワークを右クリックして「プロパティ」を開く
11) 「全般」タブの「インターネットプロトコル(TCP/IP)」を選択し「プロパティ」ボタンを押す
12) IP アドレスの設定を元の設定に戻す
13) 外部に接続しているネットワークを右クリックして「有効」にする

10.3.9. coLinux のネットワーク設定方法

インストール状態では DHCP が使用されますが、DHCP サーバが動作していない環境等では固定で IP アドレスを設定する必要があります。
ネットワーク設定は ifconfig コマンドで表示することができます。

```
colinux:~# ifconfig
eth0      Link encap:Ethernet  HWaddr XX:XX:XX:XX:XX:XX
          inet addr:192.168.0.151  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:189 errors:0 dropped:0 overruns:0 frame:0
          TX packets:115 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:24472 (23.8 KiB)  TX bytes:9776 (9.5 KiB)
          Interrupt:2

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

colinux:~#
```

図 10-7 ifconfig コマンドの実行

eth0 デバイスの IP アドレスが表示されない場合は、固定で IP アドレスを設定する必要があります。設定すべき IP アドレスですが、「ルーター接続」の場合は「TAP-Win32 adapter」のネットワークに合わせ、「ブリッジ接続」の場合は外部ネットワークに合わせます。
ここでは、以下の表の内容に設定を変更する方法を説明します。

表 10-1 ネットワーク設定

項目	設定
IP アドレス	192.168.1.100
ネットマスク	255.255.255.0
ゲートウェイ	192.168.1.1
DNS サーバ	192.168.1.1

1) coLinux 上で/etc/network/interfaces を以下のように編集する

```
auto lo eth0
iface lo inet loopback
iface eth0 inet static
  address 192.168.1.100
  gateway 192.168.1.1
  netmask 255.255.255.0
```

図 10-8 /etc/network/interfaces ファイルの編集例

2) coLinux 上で/etc/resolv.conf を以下のように編集する

```
nameserver 192.168.1.1
```

図 10-9 /etc/resolve.conf ファイルの編集例

3) 以下のコマンドを実行し、編集した内容でネットワーク設定を更新する

```
colinux:~# /etc/init.d/networking restart
Reconfiguring network interfaces: done.
colinux:~#
```

図 10-10 ネットワークの再設定コマンド

改訂履歴

Ver	年月日	改訂内容
1.0.0	2004/04/29	・初版発行
1.0.1	2004/06/04	・Minicom のインストールを apt から dpkg に変更 ・ツールチェーン名の誤字を修正 ・OTC の Makefile を汎用化
1.0.2	2004/12/15	・会社住所変更
1.1.0	2005/01/31	・8MB Flash の記述を追加
1.2.0	2005/03/01	・SUZAKU-V 用の記述を追加
1.3.0	2006/07/14	・全体構成の見直し ・ネットワーク関連情報（設定、telnet、ftp）の追加 ・ダウンローダ（Hermit）の Windows 版の記述を追加 ・SZ130-U00 Flash メモリマップの記述を追加 ・Windows 上に開発環境を構築する方法（coLinux）を追加
1.3.1	2006/08/11	・Flash メモリの書き換え方法に fpga リージョンの記述を追加

SUZAKU Software Manual　　2006 年 8 月 11 日　version 1.3.1

株式会社アットマークテクノ
060-0035 札幌市中央区北 5 条東 2 丁目 AFT ビル 6F　TEL:011-207-6550　FAX:011-207-6570