OPEN LANDISK PLATFORM Linux Ver. 2.0 README

2.0 版

（株）アイ・オー・データ機器

改版履歴

版数	年月日	内容	担当
初版	2007.4.20	新規	伊達
1.1	2007.9.21		伊達
2.0	2008.4.17	GXR を統合	伊達

目次

1 概要

このディストリビューションは、(株)アイ･オー･データ機器が開発・販売を行っている、NAS 製品を通常の Linux システムとして利用するためのものです。このドキュメントでは、そのための手順を解説します。

※ Linux は、Linus Torvalds 氏の米国およびその他の国における、登録商標または商標です。

2 適用

下記の各製品に対応しています。

- HDL-GT シリーズ
- HDL-GTR シリーズ
- HDL-GXR シリーズ

以下の各製品については、現時点で対応しておりませんが、ご要望によって対応可能です。

- HDL-GX シリーズ
- HDL4-G シリーズ

以下の各製品については、対応予定はありません。

- HDL-U シリーズ
- HDL-G シリーズ
- HDL-F シリーズ
- HDL-GS シリーズ

各製品のハードウエア仕様につきましては、それぞれの製品のマニュアルを参照してください。

3 著作権の帰属

このディストリビューションを構成するソフトウェアの著作権およびすべての知的財産権は、各ソフトウエアの原権利者に帰属します。

4 保証の免責

このディストリビューションに含まれる各ソフトウェアは、製品の瑕疵の不存在、市場性、利用者における利用可能性、特定目的への適合性、その他一切の事項に関する保証なしに、現状のままの状態で利用者に提供されます。元配布者ならびに各ソフトウエアの開発者は、各ソフトウェアに関して、明示的または黙示的を問わず、第三者の権利を侵害しないことや、ソフトウェアの性能および品質上の問題に起因して発生する問題、商品性の保証、特定の目的に対する適合性に関する黙示の保証を含む一切の保証責任を負いません。ソフトウェアの導入および運用等は、利用者ご自身の責任および費用負担をもって処理されるものとします。

5 責任の制限

このディストリビューションの元配布者およびソフトウェアの原権利者、その他プログラムの改変者および再頒布者は、ソフトウェアの誤作動、データの消失、他のプログラムと一緒に動作しないといった不具合など、プログラムの使用に関連して生じた直接損害、間接損害、特別損害、派生的損害、懲罰的損害その他一切の損害について、損害の可能性について予測可能であったか知っていたかを問わず、賠償責任を負いません。また、第三者からお客様に対してなされた損害賠償請求にもとづく損害についても一切責任を負いません。

6 メーカー保証の消失

このディストリビューションに含まれるソフトウエアを対応製品にインストールして使用することは、対応製品の品質保証条件を逸脱する改造にあたります。インストールを行った製品についてはメーカーの保証対象外です。修理、サポートは製品の保証期間内であっても受けることができなくなりますのでご注意ください。

このディストリビューションを組み込んだ対応製品を再販する場合は、製品のエンドユーザー保証は、販売者が個別に提供してください。

別途、(株)アイ･オー･データ機器との間で、保証・保守契約などを結んでいる場合は、契約内容に従った保証が受けられます。

7 システム構成時の注意点【重要】

7.1 動作温度の監視【注意！】

対象ハードウエアの設計には安全上の配慮がおこなわれていますが、ソフトウエアの動作環境によっては、機器の動作温度上限を越える可能性があります。そのような場合、機器や記憶媒体を破損するばかりでなく、発煙や発火の危険があります。

OLP を組み込んだハードウエアを製品として出荷する際には、温度試験や負荷試験などを十分に行ってください。

7.1.1 ハードディスクの温度監視

S.M.A.R.T.を使って情報を取得し、動作条件範囲内であるか確認してください。異常がある場合や、動作条件範囲外である場合は、すみやかにシステムを停止してください。

7.1.2 本体基板での温度監視

「デバイスのステイタスと制御」の項をご確認ください。動作条件範囲外である場合は、すみやかにシステムを停止してください。

7.1.3 ファンの稼動監視

「デバイスのステイタスと制御」の項をご確認ください。動作条件範囲外である場合は、すみやかにシステムを停止してください。

HDL-GT/GTR では、電源ファンの監視はできません。動作条件に合致した環境で使用し、定期的な点検をおこなってください。

7.2 設置条件【注意！】

各機器のハードウエア仕様書をご確認ください。

8 CPU アーキテクチャ

HDL-Gx シリーズは、CPU コアに ARM アーキテクチャを採用した Marvell 社の NAS 用 SoC である「88F5182」を使用しています。

kernel のチューニングやドライバのデバッグなどを行うためには、CPU のデータシートが必要です。この SoC のデータシートを入手するには、個別に Marvell 社との機密保持契約を締結する必要があります。データシートをご希望の方は、Marvell 社の日本窓口にお問い合わせください。

http://www.marvell.com/sales/index.jsp?location=japan

9 ディストリビューション構成

このディストリビューションは、もととなる環境に Debian GNU/Linux Ver.4.0(Etch)を採用しています。そのため、Debian のほとんどのバイナリパッケージとパッケージ管理システムをそのまま利用することができます。

Debian は Linux の代表的なディストリビューションの１つであり、ARM を含む多くの CPU アーキテクチャに標準で対応しています。

Debian の標準環境とは主として以下の点が異なります。

- kernel コードは、SoC 対応の独自コードが含まれます。
  そのため、Debian の標準環境を利用していません。
- kernel コンパイル時の設定も異なります。
  そのため、udev や initramfs, SE Linux などは使用していません。
- エンディアンはリトルエンディアンです。
- 環境は OABI です。ARM EABI は使用していません。
- ハードウエア固有の機能のためのドライバやユーティリティを含んでいます。

この文書では、Linux の操作方法や、Debian 操作方法やパッケージ管理についての情報には触れません。詳細につきましては、Debian の公式ドキュメント、市販の書籍や Web サイトなどの情報を参照してください。

10 ソフトウエアのライセンス

このディストリビューションを構成するソフトウエアは、オープンソースライセンスが適用されています。各コードのライセンスについては、配布物に含まれるソースコード内の文書を参照してください。

このディストリビューションを構成するソフトウエアのソースコードは、配布物に含まれています。多くのオープンソースライセンスに含まれるライセンス上の義務は、配布時点で完遂されていますので、配布元にソースコードを再請求しないでください。

また、このディストリビューションをそのまま再配布する場合は、ドキュメント、バイナリコード、ソースコードを一体のものとして再配布してください。

このディストリビューションをカスタマイズしたものを再配布する場合は、各コードのライセンス条件に従ってください。独自のディストリビューションとして再配布する場合や、ハードウエアに組み込んで再配布する場合がこれに該当します。

このディストリビューションには、GNU General Public License Version 2 に基づいて許諾されるフリーソフトウェア・モジュールを含んでいます。利用者は、Free Software Foundation が定めた GNU General Public License Version 2 または当該ライセンス条件の改訂版が定める条件に従って、ライセンスの該当するソフトウエアを再頒布または変更することができます。これらのソフトウエアの頒布にあたっては、弊社および開発者は、いかなる保証も行ないません。

GNU General Public License Version 2 の全文は、ディストリビューションインストール後の "/usr/share/common-licenses/"ディレクトリに格納されていますので、利用を開始する前に必ずお読みください。

11 添付ソフトウエアの構成

インストール用のファイルは２つのアーカイブにまとめられています。

・OLP02_boot_etch[日付][マイナーリビジョン記号].tgz
起動に必要な加工を行った kernel イメージと initrd が含まれます。(以下 boot.tgz と表記)

・OLP02_base_etch[日付][マイナーリビジョン記号].tgz
rootfs 環境が含まれます。(以下 base.tgz と表記)

このドキュメント執筆時点のファイル名、サイズ、タイムスタンプ、md5sum は、以下の通りです。

OLP02_base_etch20071203a.tgz 427681217 2007-12-03 19:47 3fe6f396b49c1860276ee2bd8ed50122
OLP02_boot_etch20071203a.tgz 3916922 2007-12-03 19:48 ccc60f745d07b979f54c985284719375

最新版については、個別にお問い合わせください。

12 インストール方法

12.1 利用可能なドライブについて

SATA2 仕様のハードディスクを使用してください。
起動には容量は無関係ですが、動作確認は 120GB-1TB のドライブで行っています。

12.2 インストール環境

Debian GNU/Linux 4.1 環境がインストールされ、SATA ドライブが接続可能な PC をご用意ください。
boot.tgz, base.tgz ファイルはあらかじめ作業環境にコピーしておいてください。

12.3 インストールドライブのパーティション構成

fdisk などのコマンドで、以下のようなプライマリ・パーティションを構成してください。

番号	ID	サイズ
1	83	任意 (通常、128MB 程度を確保します)
2	82	任意 (通常、1GB 程度を確保します。fsck 時などで必要になります)
3	83	任意 (通常、最低 2GB 程度必要です。)

例：

```
# fdisk -l /dev/sdb

        Disk /dev/sdb: 250.0 GB, 250059350016 bytes
        255 heads, 63 sectors/track, 30401 cylinders
        Units = cylinders of 16065 * 512 = 8225280 bytes

     Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1          17      136521   83  Linux
/dev/sdb2              18         142     1004062+  82  Linux swap / Solaris
/dev/sdb3             143        1388    10008495   83  Linux
/dev/sdb4            1389       30401   233046922+  83  Linux
```

12.4 ファイルシステムのフォーマット

ID 82 で確保した領域を swap に、それ以外を ext3 ファイルシステムでフォーマットします。

例：

```
# mkswap /dev/sdb2
# mkfs.ext3 /dev/sdb1
# mkfs.ext3 /dev/sdb3
```

12.5 アーカイブの展開

第一パーティションをマウントし、boot.tgz を展開します。
インストール対象に対応した uImage と initrd のコピーを行います。

GT/GTR の場合：
"uImage.OLP02GT"を"uImage"にファイル名を変更します。

GXR の場合：
"uImage.OLP02GXR"を"uImage.gxr"にファイル名を変更します。
"initrd"を"initrd.gxr"にファイル名を変更します。

第三パーティションをマウントし、base.tgz を展開します。

例：

```
# mount /dev/sdb1 /mnt/
# cd /mnt
# cp [アーカイブ格納場所]/boot.tgz /mnt/
# tar zxvfp boot.tgz
# rm boot.tgz
# mv uImage.OLP02GXR uImage.gxr
# mv initrd initrd.gxr
# cd /
# umount /mnt/

# mount /dev/sdb3 /mnt/
# cd /mnt
# cp [アーカイブ格納場所]/base.tgz /mnt/
# tar zxvfp base.tgz
# rm base.tgz
# cd /
# umount /mnt/
```

12.6 インストールの終了

インストール用の環境を終了し、インストール済みのドライブを取り外します。
取り外したドライブを対象ハードウエアに取り付けます。
RHD **カートリッジならびに、**HDL-GXR **の分解方法については、ここでは扱いません。個別にお問い合わせください。**

13 起動確認とシステム終了

13.1 メインスイッチでの起動

システムの起動は、FlashROM に内蔵されたブートローダーが行います。HDL-Gx シリーズでは、ブートローダーとして u-boot を採用しています。

本体正面の POWER スイッチを入れると、u-boot が起動されます。

u-boot は、接続されたドライブの有無を調べ、最初に見つかったドライブの第一パーティションを読み出し、格納された kernel イメージと initrd をメモリにロードし、kernel を起動します。

起動プロセスには、数分必要です。

13.2 メインスイッチでの終了

POWER スイッチを 1 秒以上押し続けると、安全にシステムを終了し、自動的に電源を切ります。

対象ハードウエアの POWER スイッチは、ソフトウエアで制御されています。システムに含まれるソフトウエアが正常に動作していない場合、電源を切ることはできません。

AC コンセントなどを抜いて電源を切ると、ファイルやファイルシステムを壊し、正常に起動できなくなる恐れがあります。

14 ネットワーク設定

インストールされたシステムでは、DHCP サーバーより IP アドレスやネットワーク環境情報を取得します。

本体に設定された MAC アドレスは、背面のシールに記載されています。DHCP サーバーにて取得アドレスを確認するか、MAC アドレスに対して IP アドレスを割り振ってください。

15 リモート接続と初期ユーザー設定

インストールされたシステムでは、ssh サーバーが動作しています。ssh クライアントを使って、取得された IP アドレスに接続してください。

初期環境で設定されているユーザーとパスワードは以下になります。

ユーザー	パスワード
admin	olpadmin
root	olpadmin

リモート接続では、root ユーザーでログインすることはできません。admin ユーザーでログインの上、su コマンドなどで root ユーザー権限を利用してください。

16 シリアルコンソールの接続

起動確認など、コンソールを使った開発が必要な場合は、基板から直接シリアルコンソールを接続することができます。
製品の分解方法ならびにシリアルコンソールの接続方法については、ここでは扱いません。個別にお問い合わせください。

17 デバイスのステイタスと制御

各装置には、ボタン、LED、ブザーなどがあり、ソフトウエアから制御することができます。制御方法は装置によって異なります。

17.1 HDL-GT/GTR のハードウエア制御

HDL-GT/GTR でのボタン、LED、ブザーなどの制御は、本体に内蔵されたマイコンによって行われています。

内臓マイコンでは、以下の項目の制御を行っています。

デバイス	説明
LED	装置の LED 制御 (NIC の Link/Act LED を除く)
	ステイタス LED, HDD1～4 の LED の点灯、点滅、消灯、色変更
HDD	HDD1～4 の電源 ON,OFF 制御 (USB/eSATA 外付けドライブを除く)
ボタン	装置のボタン入力検出
ブザー	圧電ブザーを鳴動
装置温度検出	温度センサからの入力をもとにした FAN 制御を行う
FAN	FAN 回転数を取得し、FAN 故障を通知
	温度センサからの入力をもとに動的に FAN 回転数を変動
HDD の Act 反映	Act 信号を監視し LED 状態に反映

ATX 電源制御	メイン電源の ON,OFF を制御
リセット信号送出	CPU、周辺コントローラに対してリセット信号を送出
RTC アラーム検出	RTC からのアラーム信号を検出し反映

このマイコンと通信するための daemon として、R8contd がインストールされています。

他のプロセスから、これらのハードウエアの状態を取得したり変更したりする場合は、R8contd との通信を FIFO 経由で行います。FIFO は、ライト側が/dev/r8write,リード側は/dev/r8read として用意されています。プログラムから、FIFO にコマンドを送ったり、読み出したりすることで、マイコンを制御することでがきます。

17.1.1 ボタンの入力監視

ボタンの入力監視は、R8contd によって行われます。R8contd は/usr/local/sbin に置かれています。

R8contd はシステム起動時に init.d スクリプト経由で動作させておきます。OLP 環境では、他のハードウエアと共存するために、/etc/init.d/hwchk.sh を経由して起動されています。

R8contd はボタン制御を行っている R8 マイコンからの信号を受信し、内容に応じた引数を指定して/usr/local/bin/R8event を起動します。

R8event に渡される引数と内容は以下の通りです。

P	POWER ボタン長押しの検知：shutodown 相当の機能割り当て
C	STATUS ボタン長押しの検知
R	背面リセットボタン長押しの検知
w	スロット１のドライブが取り外し
W	スロット１のドライブが挿入
x	スロット２のドライブが取り外し
X	スロット２のドライブが挿入
y	スロット３のドライブが取り外し
Y	スロット３のドライブが挿入
z	スロット４のドライブが取り外し
Z	スロット４のドライブが挿入
F	FAN 停止
T	装置温度異常：ボード上のセンサが 50 度以上を検知

R8event は、シェルスクリプトですので、適宜動作を変更して使用してください。初期状態では、POWER ボタンによるシステム停止のみが定義されています。

17.1.2 STATUS LED の制御

ステイタス LED **の制御は以下のコマンドで行います。**

状態取得 :sts

例：# echo ":sts" > /dev/r8write

返り値 0：**正常**(sts on)
1：**処理中**(sts blink)
2：**エラー**(err)
3：**装置リセット直後**

例：# cat /dev/r8read
;0[¥n]

状態変更 :sts [on/blink/err]
on： **正常**(sts on)
blink： **処理中**(sts blink)
err： **エラー**(err)

17.1.3 HDD スイッチ/電源と LED の制御

HDD **スイッチと** LED、HDD **の電源の制御は以下のコマンドで行います。**

状態取得 :hdd

返り値	0：**起動時スイッチオン**	LED **青点灯**	**正常運用　起動時デフォルト**
	1：-	LED **赤点灯**	
	2：-	LED **赤点滅**	
	3：**起動後スイッチオン**	LED **青点滅**	HDD **電源** ON※1
			ホットプラグ挿入時自動遷移
	4：-	LED **青点滅**	
	5：**未装着状態**	LED **消灯**	HDD **電源** OFF※2
			HDD **未挿入時デフォルト**

hdd0～3 **までを連続して出力**

例：；5000[¥n]

※1 5 からの遷移時のみ電源 ON を行います。

※2 処理の前にソフト的に HDD 切り離しをおこなってください。

状態変更 :hdd [0-3] [0-5]

※HDD スイッチが OFF で状態が５の場合は、操作を行っても反映されず無視されます。

17.1.4 ボタンの状態一覧取得

ボタンの状態を R8contd を経由せずに直接確認できます。

状態取得 :btn

返り値 c,C：COPY
r,R：RESET
p,P：POWER
w,W：HDD0
x,X：HDD1
y,Y：HDD2
z,Z：HDD3
全ボタンの状態を連続して出力
押されてる：大文字発行
離されてる：小文字発行

例：；cRpWXYZ[¥n]

17.1.5 ブザーの制御

現在のアラーム信号の状態を返します。

状態取得 :alarm

返り値 a：アラーム OFF
A：アラーム ON

例：；A[¥n]

状態変更 :mml [演奏データ]

1 行分の演奏が終わると；0[¥n]を返す。

演奏キューが一杯の時、ボタンもしくはコマンドで演奏中止した時に；1[¥n] を返す。
引数なしで、演奏を中止する。

例：:mml cdefgab

演奏データ	音符	A-G	A〜G がそれぞれハ長調のラシドレミファソに対応
	音階	-+	MML の後に+, -(+が♯に, -が♭)をつけ、音階を半音ずつ変更
	音長	.48	無指定時は音長, "."で 1.5 倍の音長, 4 で四分音符, 8 で八分音符
			デフォルトの音調は L で設定する。。初期状態は L4。
	休符	r	
	テンポ	T[30, 40, 50, 60, 75, 100, 120, 150, 200, 300]	
			数値は 1 分間に演奏される 4 分音符の数に相当。初期値は 120。
	オクターブ	O[3-7<>]	
			オクターブを設定。初期値は O4。
			< >でオクターブを相対的に+1,-1 する
	ゲートタイム	Q[1-8]	
			1 音中で実際に発音している時間を音長の n/8 時間単位で設定する
	音量	V[1-15]	ボリュームの設定を行う。
	繰り返し	|;n ・・・・・ ;|	
			|;n 〜 ;| で囲まれた部分を n 回演奏。n の設定範囲は 0〜255。
			0 を指定すると無限回演奏。また、n を省略時は 0 と解釈。
			ネスト記述は未対応。繰り返し出来る範囲は 1 行分。
	文字数		受け付ける 1 行は最大 31 文字まで。
	キューの長さ		演奏キューは 2 行分。
			演奏が 3 行以上になる場合は、2 行入力した後、
			マイコンからの ack(；0[¥n])を待って次の行を入力してください。

例：

```
:mml t200o4l4q7
:mml <g.e8g.e8l8gegb<d2
:mml l4c.>a8<c.>a8
:mml l8<c>af+ad2
```

MML コマンドの仕様については、下記 URL を参考にしました。
http://www2s.biglobe.ne.jp/~yyagi/zmc2man/zmsman/basicmml.html

17.1.6 ファンの状態取得

ファンの回転数を取得することができます。

状態取得　　:fan

返り値　　回転数は 16bit の unsigned データで返します。

例：；0a1d[¥n] (2589rpm)

ファンの速度制御は基板センサの温度検知結果によって自動的に行われます。

40 度　　37 度
高速 → 中速 → 低速
← ←
45 度　　42 度

17.1.7 基板センサの温度取得

基板上にある温度センサの温度を取得します。

状態取得　　:temp

返り値　　2 バイトで返します。
装置温度は 8bit の signed データで返します。(マイナスも含むため)単位は℃です。
計算方法 : 1/(LN(1/(256/ADV-1))/3380+1/298.15)

例：；2e[¥n] (0x2e=46℃)

17.2 HDL-GXR のハードウエア制御

本体の LED, ブザー, ボタンの制御は、kernel 組み込みのドライバで行います。

17.2.1 LED ドライバ (leddrv)

LED の状態取得や動作変更は、/dev/leddrv を読み出したり、値を書き込むことで行います。

状態取得　　/dev/leddrv を読み出し

例：# cat /dev/leddrv

状態変更　　[LED 名] [on/off/blink]

LED 名 power_LED
status_LED
error_LED
usb1_LED
usb2_LED
esata_LED

例：# echo "status_LED on" > /dev/leddrv

17.2.2 ブザードライバ (buzdrv)

ブザーの状態取得や動作変更は、/dev/buzdrv を読み出したり、値を書き込むことで行います。

状態取得 /dev/buzdrv を読み出し

例：# cat /dev/buzdrv

状態変更 [count [回数]] pattern [パターン文字列]

繰り返し count に続く数値の回数だけ繰り返します
回数指定が 0 では 100 万回になります

パターン文字列 単音 . (ピリオド)
長音 - (ハイフン)
無音 _ (アンダースコア)

例：# echo "pattern .__.__." > /dev/buzdrv
echo "count 5 pattern .__.__." > /dev/buzdrv

17.2.3 ボタンドライバ (btndrv) FAN の GPIO 初期化含む

ボタンならびに外付けドライブの状態取得や動作変更は、/dev/btndrv を読み出したり、値を書き込むことで行います。

状態取得 /dev/btndrv を読み出し

現在の状態が一覧で表示されます。
ボタンは、ON/OFF 状態が表示されます。
LED の状態が enable/disable で表示されます。

検出状況は一度 read すると再度状態の読み込みを行います。
ボタンの ON は、約 1 秒押し続けた場合に検出されます。

例：# cat /dev/btndrv

```
power_SW        off     enable
copy_SW  off    enable
reset_SW off    enable
esata_LED       disable
usb2_LED        disable
usb1_LED        disable
```

状態変更　　ボタン： [ボタン名/LED 名] [on/off]

ボタン名/LED 名

power_SW	電源ボタン
copy_SW	コピーボタン
reset_SW	背面リセットボタン
esata_LED	外付け HDD LED
usb1_LED	外付け USB HDD LED
usb2_LED	外付け USB HDD LED

例：# echo "copy_SW off" > /dev/btndrv
echo "usb1_LED on" > /dev/btndrv

外付けハードディスクの LED は後述するボタンからの入力時以外は自動で制御されません。ソフトウエアでデバイスの確認とマウント・アンマウントができた場合に、LED を明示的に変更してください。

17.2.4 ボタンが押された時の自動制御

各ボタンが押されると、以下の動作が行われます。

- ブザー音停止
- error_LED の消灯
- status_LED の点灯
- 増設ドライブの接続確認と、接続時点灯、取り外し時消灯

18 付録：u-boot で起動可能な kernel と initrd の作成

独自にコンパイルした kernel イメージや、独自に用意した initrd イメージを使用する場合、そのままでは起動することができません。

u-boot で起動可能な各イメージを作成するには、u-boot に含まれる mkimage ユーティリティが必要です。mkimage バイナリは、インストール環境に含まれています。

mkimage を使って、作成した各イメージを u-boot で起動可能にします。

18.1 kernel の加工

新規に作成した kernel イメージに対して、起動可能にする処理を mkimage コマンドで行っておきます。

例：

```
# mkimage -A arm -O linux -T kernel -C none -a 0x00008000 -e 0x00008000 ¥
-n "OPEN LANDISK PLATFORM Linux 0.1" -d vmlinuz-2.6.12.6-arm1 uImage
Image Name:   OPEN LANDISK PLATFORM Linux 0.1
Created:      Wed Apr 18 16:08:17 2007
Image Type:   ARM Linux Kernel Image (uncompressed)
Data Size:    1775612 Bytes = 1734.00 kB = 1.69 MB
Load Address: 0x00008000
Entry Point:  0x00008000
```

18.2 initrd の加工

initrd_uboot として作成したファイルシステムイメージをもとに、アーカイブして u-boot 用の起動イメージを作成する例をあげます。

例：

```
# gzip -9 initrd_uboot

# ls -al initrd_uboot.gz
-rwxrwxrwx  1 root ftp 2855022 Apr 10 18:05 initrd_uboot.gz

# mkimage -A arm -T ramdisk -C none -d initrd_uboot.gz initrd_new
Image Name:
```

```
Created:         Wed Apr 11 17:28:23 2007
Image Type:    ARM Linux RAMDisk Image (uncompressed)
Data Size:      2855022 Bytes = 2788.11 kB = 2.72 MB
Load Address: 0x00000000
Entry Point:   0x00000000

# ls -al initrd_new
-rw-r--r--   1 root ftp 2855086 Apr 11 17:28 initrd_new

# file initrd_new
initrd_new: PPCBoot image
```

既存の u-boot **用** initrd **イメージを展開するには、先頭の** 64 **バイトを取り除いてやることにより、** ext2 **ファイルシステムイメージの** gzip **アーカイブに戻すことができます。**

添付の initrd **イメージを展開するには、以下のようにします。**

```
# cp /boot/initrd /home/

# cd /home/

# dd if=initrd bs=64 skip=1 of=initrd_uboot.gz
44552+1 records in
44552+1 records out
2851351 bytes transferred in 2.618202 seconds (1089049 bytes/sec)

# ls -al initrd_uboot.gz
-rw-r--r--   1 root ftp 2851351 Apr 10 18:05 initrd_uboot.gz

# file initrd_uboot.gz
initrd_uboot.gz: gzip compressed data, was "initrd", from Unix, max compression
```

gzip **アーカイブを展開して、** ext2 **ファイルシステムイメージをループバックマウントすると、内容が確認できます。**

```
# gzip -d initrd_uboot.gz

# ls -al initrd_uboot
-rw-r--r--   1 root ftp 16777216 Apr 10 18:05 initrd_uboot
```

```
# file initrd_uboot
initrd_uboot: Linux rev 1.0 ext2 filesystem data

# mount -o loop ./initrd_uboot /mnt
# cd /mnt
# ls
bin  dev  etc  home  lib  linuxrc  lost+found  mnt  proc  sbin  sys  tmp  usr  var
```

すでに存在する ext2 **ファイルシステムイメージをループバックマウントして加工すると、アーカイブした時にサイズが十分に小さくならないことがあります。**

そのような場合は、内容が空のファイルシステムイメージを作成して、そこにアーカイブしておいた initd **のファイル群をコピーするといいでしょう。**

```
# dd if=/dev/zero of=./initrd.null bs=2M count=1
# mkfs.ext2 -N 4096 -F -L OLPLinux0.1 -m 0 initrd.null
# tune2fs -c 0 -l initrd.null
# mount -o loop initrd.null /mnt/
# tar zxvfp initrd.tgz -C /mnt/
# umount /mnt/
# mv initrd.null initrd.OLP01a
```