大統一 Debian 勉強会

U-Boot についてあれこれ

野島 貴英
nozzy@debian.org

2012 年 6 月 13 日

自己紹介

- Linux は長年触ってます（でも詳しくない...orz）
- Linux との付き合いは組み込み用途の安価な OS として MSDOS 以外で OS 探してたのが動機といえば動機。
- IT の波に飲まれて、今は IT 系のサーバー屋。でも Cloud 時代もきちゃって仕事なくなりそうなので、次は何しようかなぁーと模索中。

ところでみなさん組み込みシステムって知ってますかー？

組み込みシステムとは:
「特定の機能を実現するためのコンピュータシステムの事」
だそうで...

- 特定の機能を実現できれば良いので、部品削ってケチりまくる事多数。
- 一方で、極端にタフな動作環境、動作が求められます
- 特定の機能を実現の為、妙な入出力センサ沢山搭載してます

組み込みシステムのプロの方いますー？

- プロの方が過半数越えた場合：プランA（BOF、アドリブ多め）
- プロの方が過半数未満の場合：プランB（資料を元に...）

どんなCPU使ってます？

差し支え無い範囲で教えていただけますと嬉しいな...

- SH シリーズ？
- ARM?
- PIC 系？
- Intel x86 系?
- Z80?
- Motrora 68000 系？
- PowerPC?
- その他

今回の発表はARM機材で

巷でARM流行ってるよね?

- 昨今のスマホ事情、Android端末などの関係で、いろいろ評価ボードの入手が便利。
- 電子書籍端末もARMベースだったり（今回のターゲットボード）
- Rasberry Piも（入荷状況は混雑しまくりの模様ですが...）

というわけで、今回の発表は妙にARM CPUに偏ります。
他のCPUの話を聞きたい方はすみませんが、情報交換をさせていただければ嬉しいな。

※他の評価機材を自分は持っていない事は秘密です。

今回SoCはTexas Instrumentsで

自分の趣味ですが、「ありがとう！TI子供の時からファンだったよ！」
(74シリーズとか。当時7セグドライバから始まり... 大人になってはDSP...)
良いところ

- サポートとコミュニティが諸々ある。
- いろいろオープンにしてくれてる（エコシステムとの事）
- 流行りのチップいろいろのっけてる (VRシリーズのOpenGL ESアクセラレーションつきGraphicsチップとか)

なので、実機のSoCはOMAP36シリーズにて。

自分の家でも組み込みやってます？

差し支え無い範囲で教えていただけますと嬉しいな...

- ICE 持ってます?
- ロジアナある？
- オシロある？

ところでブートローダ

外部記憶装置に搭載したOSを主記憶へ展開して、実行を引き継ぐ為のシンプルなソフトウェア。
でも昨今は、

- スクリプト言語は積むわ、
- 様々な外部記憶装置にアクセスするために、数々のチップセットの制御コードを搭載するわ、
- 数々のフォーマット/ファイルシステムに対応するようになるわ、
- インタラクティブにユーザとやりとりする為の簡易デバッガ（リモートデバッグの為のドライバも含む）、モニタなど積むわ、

など、シンプルだった「はず」のソフトウェアは機能強化の道をひた走る...

主なブートローダ一覧

Comparison of boot loaders
http://en.wikipedia.org/wiki/Comparison_of_boot_loaders
そりゃもういろいろある模様。

U-Boot

DENX Software Engineering 社が主にメンテしている GPL なブートローダの実装。

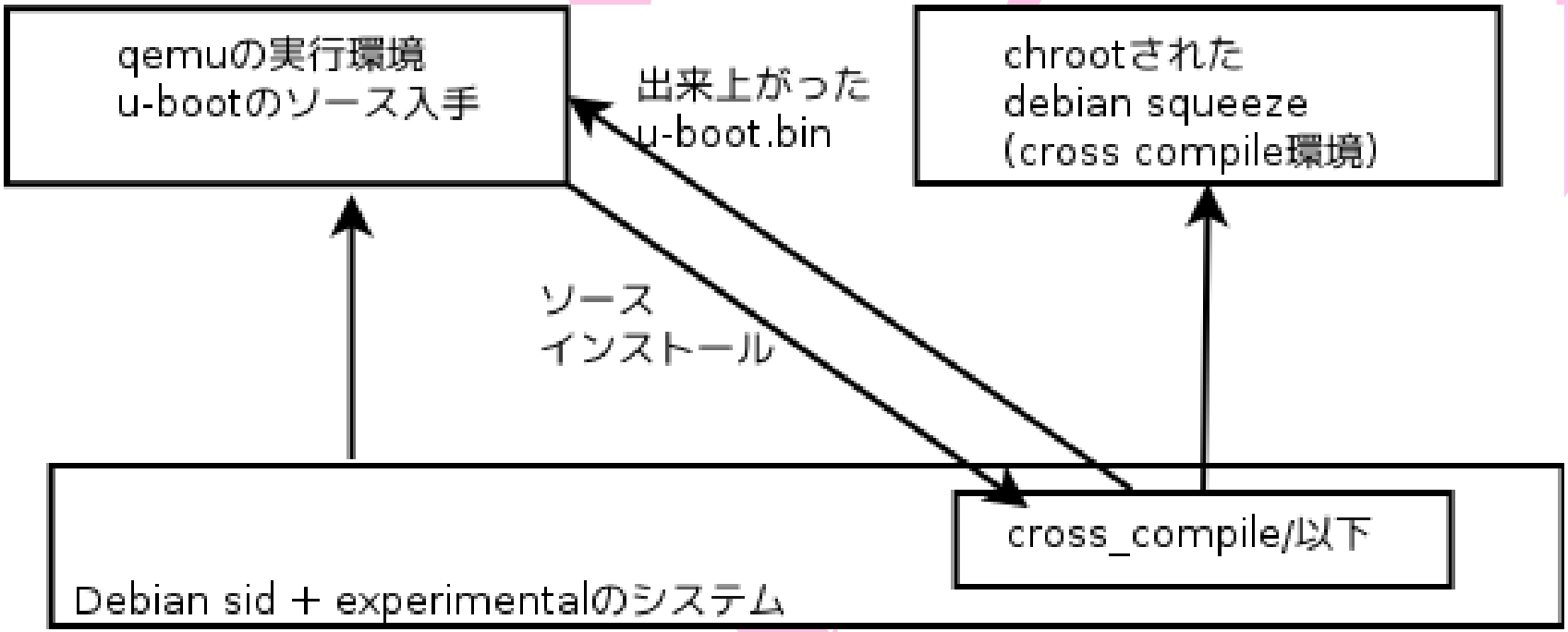
- 今では Das U-Boot といったりもします。
- 先の”Comparison of boot loaders” にあるとおり、圧倒的な量のサポート CPU 数を誇る。
- モニタ、デバッガスタブ、独自のスクリプト言語を搭載
- ブート対象の OS は Linux に偏ってるかな...

百聞は一見にしかず

百聞は一見にしかず。まず versatilepb エミュレータで動作させて試してみます。(... やり方は書くと長いので、大統一 Debian 勉強会資料を参照...)
QEMU のおかげで、ブートからシミュレーションも一発さ！

え？やり方分かりにくいって?

大統一 Debian 勉強会資勉強会資料のやり方を図示するとわかりやすいです。※全部で 4 ページの制約がありまして割愛しちゃいました... ごめん



補足：ところでversatilepbって何

ARM 社の評価ボード VERSATILE シリーズの 1 つ。http://www.arm.com/ja/products/tools/development-boards/versatile/platform-baseboards.php
CPU は ARM926EJ-S。
感想：本物は触ってないので実際はどうかは未評価。QEMU では HDD デバイス、PCI デバイス、ネットまでつながる素敵な奴。debian の ARM 版試したりするのに最高（東京エリア Debian 勉強会資料「Deb 専」2012/04 月参照）メモリを 512MBytes 以上に QEMU で設定するとカーネルがなぜか死ぬけど...(なぜだ？)

補足その2：なんで experimental の qemu なのか？

なんでわざわざ experimental 版 qemu なの？
理由：
`http://www.elinux.org/Virtual_Development_Board` のパッチが当たっているバージョンの QEMU が必要な為です。実際デバッガ回しながら QEMU 追いかけるとわかるのですが、versatilepb 版の U-Boot が pflash の初期化をしようとすると、QEMU 側の versatilepb エミュレーションにそれを受け入れるコードが無い為、exception を発生してしまい停止します。

U-Boot 初めて触る人には簡単なコマンド

- help
  help のみで一覧。あとは help コマンド名で help が読める。
- bdinfo
  ボードの情報を表示。
- printenv
  環境変数（いじるといろいろ U-Boot の動作を変えられる)
- setenv
  環境変数いじってみる。

ところで U-Boot 便利

自分から見た良いところ

- LCD/Flash/USB とかの出力デバイスなど共通/便利/必要そうなものは共通化されている。
- 数々のファイルシステムに対応 (cramfs/ext2/fat/jffs2/reiserfs... 等)
- 設定ファイル（というか設定バイナリ）である程度作を変更可能
- LCD についてはフォントすら内臓してるのでラスタ形式のビットマップしか LCD に描画できなくても余裕。
- 基本設計は変に複雑でないところ（抽象化やりすぎとかがないので...。）また、沢山のわかりやすいマクロとスイッチ。

U-Boot を俺ボードで動かす時は (初心者向け 1/2)

- Step1.
  u-boot-src/README を読む
- Step2.
  CPU/SoC が u-boot-src/arch 以下にすでにあれば再利用を検討
- Step3.
  ボードのチップセット/回路が近いものがあれば、u-boot-src/board/以下をパクる
- Step4.
  u-boot-src/include/configs 以下に対応したファイルを作成し、U-Boot の数々のマクロ定数を定義する。

U-Boot を俺ボードで動かす時は (初心者向け 2/2)

- Step5. u-boot-src/boards.cfg に俺ボードに対応した定義を入れてやり、Step1/Step2/Step3 を融合する。

で、あとは make ARCH=XXX CROSS_COMPILE=gcc のクロスコンパイル用コマンドの prefix 俺ボード名_config して make。(「俺ボード名」の部分は u-boot-src/boards.cfg に記載した名前)
ただし、最近は fdt(Flatten Device Tree) とか便利なものもあるらしい。

実機にBN Nook Colorを使ってみる

これは電子書籍端末。
スペック

- SoCにはOMAP36シリーズを利用
- LCD/miniSDカード/USB搭載

実はOMAP用ブート形式のフォーマットでminiSD用意して、差し込んで電源ONすると思わずそちらからブートしちゃうという特典機能あり。

BN の公開している U-Boot ソースを改造

ただ動かすだけではまったくつまらんので、改造をしてみる。
とりあえず、LCD に

- カラーパターンだしてみる。
- U-Boot メッセージ出してみる

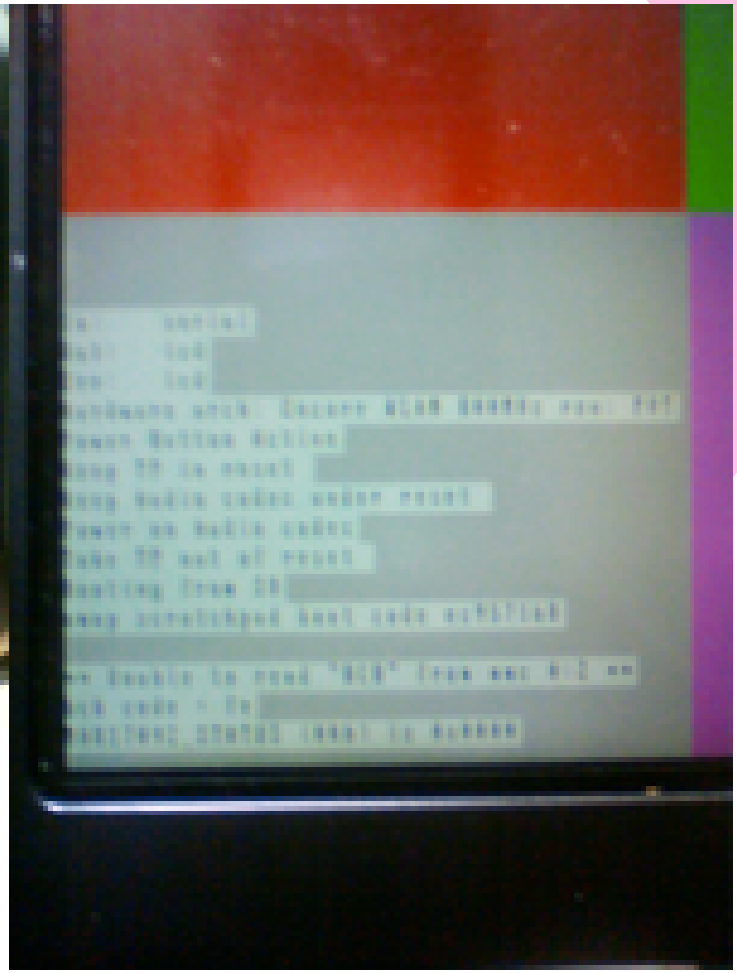
改造解説

とりあえず勉強会資料のパッチを見てもらえばわかりやすいかと。

- lcd.c いじっているのは、Nook Color の場合、起動画像（ビットマップ）を出して細かい事をかくしているのを解除する為。パッチの変更箇所下にある lcddev.putc/lcddev.puts の変数は C 分かる人にはお馴染みの関数 (putc/puts) に見えますでしょうか？
- omap3621_evt1a.h は、stdout/stderr を全部 LCD へ振り向ける環境変数を新たに定義し、さらに入出力デバイスは環境変数の中身を優先して使ってくれというマクロを ON にする。
- ついでに「カラーパターンだしてね」という内容と、「初期化時も LCD 出しっぱなしにしてね」というマクロを ON にする。
- itemize

ほら改造も簡単。

動作しました！



最後に：Flatten Device Tree について

以下が参考になるかと

- FDTWiki
  `http://devicetree.org/Main_Page`
- ePAPR
  `http://www.power.org/resources/downloads/Power_ePAPR_APPROVED_v1.0.pdf`