５ＭＳ実験ＯＳ　第５回　マイコン(sh2)ボードの開発環境

1. 概要

本製品では、以下の Linux ホスト用の GNU 開発環境を提供しています。

gcc version 3.2.2

binutils version 2.13

gdb version 5.3

以下の GNU 開発環境パッケージの圧縮ファイルは、CD-ROM(Rel1.03) 内の /jp/soft , /common/soft ディレクトリにあります。

Linux 用プラットホーム共通部分 te.Linux-i686.common.14.tar.gz

Linux 用 ARM 対応部分 te.Linux-i686.arm.08.tar.gz

T-kernel リソース部分 te.resource.tbat91.02.tar.gz

※ T-kernel リソース部分には、T-kernel、T-kernel Extension 用のヘッダファイル、ライブラリ、Makefile、デバイスドライバ/サンプル/ユーティリティソースが含まれています。

以降コマンド入力の[E]の記述は省略する。

2. 開発環境のインストール

(1) 開発環境をインストールするディレクトリを作成します。

ホームディレクトリ(/home/ubuntu)の下に te ディレクトリを作成し、そこ下に開発環境を整備する。

```
mkdir /home/ubuntu/te
```

2. ディレクトリ /home/ubuntu/te で、GNU 開発環境用のパッケージをすべて展開します。

＊以下の 3 個の圧縮ファイルを/home/ubuntu/te にコピーする。(cp コマンドでなく，GUI ツールのファイルマネージャを利用してもよい)

```
te.resource.tbat91.02.tar.gz
te.Linux-i686.common.14.tar.gz
te.Linux-i686.arm.08.tar.gz
```

＊/home/ubuntu/te で tar コマンドで展開する。

展開後 ls コマンドでファイルが展開されていることを確認する。

ファイルを展開する。

```
tar zxvf te.resource.tbat91.02.tar.gz
tar zxvf te.Linux-i686.common.14.tar.gz
tar zxvf te.Linux-i686.arm.08.tar.gz
```

レポート 2-1 : /home/ubuntu/te/tool/Linux-i686/etc ディレクトリにあるファイルを書け。

3. 開発環境を利用するためのツールの設定

＊　/usr/local/bin/perl で perl を起動できるようにする。

/usr/local/bin に ファイル perl があるか確認する。

なければ/usr/local/bin から perl に対してシンボリックリンクを以下のように作成する。

```
cd /usr/local/bin/
sudo ln -s /usr/bin/perl
```

レポート 3-1 : ls -l の応答を書け。

cd　でホームディレクトリ (/home/ubuntu)に戻る。

＊ gdb や gterm などで使用する通信ポートを利用できるようにする。
com ポート 1 を使用する場合、デバイスファイル/dev/ttyS0 をユーザ ubuntu が読み書き可能にする。(この処理は，パソコンを再起動する度に行う必要がある。)

```
sudo chmod 666 /dev/ttyS0
```

4．シェルの環境整備

以下の環境変数を設定する。

BD

開発環境のベースディレクトリ開発環境をインストールしたディレクトリを設定します。例えば、開発環境を/home/ubutu/te にインストールしたのであれば、 BD には /home/ubutu/te を設定します。

GNUs

GNU 関連ツール GNU make があるディレクトリを指定します。 GNU make が /usr/bin にあれば、(/usr/bin ではなくて) /usr を設定します。

GNU_BD

クロス開発用の GNU 関連ツールのベースディレクトリ開発環境に含まれる GNU 関連ツールのディレクトリを設定します。通常は$BD/tool/Linux-i686 を指定してください。

GNUarm

arm 用の GNU 関連ツールのディレクトリ開発環境に含まれる arm 用 GNU 関連ツールのディレクトリを設定します。通常は$GNU_BD/arm-unknown-tkernel を指定してください。

GCC_EXEC_PREFIX

gcc 関連ディレクトリ gcc で使用する各種プログラムのあるディレクトリを設定します。
通常は$GNU_BD/lib/gcc-lib/ を設定してください。

以上の設定を.bashrc に記述しシェル(端末)を起動すると自動的に設定されるようにする。.bashrc ファイルの最後（fi の次の行）に以下を追加して保存する。

```
export BD=/home/ubuntu/te
export GNUs=/usr
export GNU_BD=$BD/tool/Linux-i686
export GNUarm=$GNU_BD/arm-unknown-tkernel
export GCC_EXEC_PREFIX=$GNU_BD/lib/gcc-lib/
```

端末を終了して、もう一度起動するとこれらの環境変数が設定される。

レポート 4-1：これらの環境変数が正しく設定されていることを確認する方法を書け。

これらの環境変数は make 時に $BD/etc/makerules 中で使われます。環境変数を正しく設定しているのに make がうまくいかない場合や、環境変数の意味するところを正確に知りたい場合は、直接$BD/etc/makerules を参照してください。

5．ボードの初期化

ボードの構成図は，pdf ファイル「第 5 回テキスト_ボード構成図」に示す。

（1）ボードの接続

DIP-SW の設定はかならず電源オフの状態（電源コードを抜く）で行ってください。
ボードのディップスイッチ(DSW1 1~4)が OFF であることを確認する。
RS-232C ケーブルをボードのシリアルコネクタとパソコンの RS-232C 端子に接続する。

ボードに電源ケーブルを接続する。

（2）パソコン側での操作

CD-ROM(rel 1.03)の jp/soft/romimage.mot を/home/ubuntu/te にコピーする。

端末で/home/ubuntu/te ディレクトリにて

```
/home/ubuntu/te/tool/Linux-i686/etc/gterm -l /dev/ttyS0 -38400
```

＊ターミナルソフト gterm を起動

以下のメッセージが出る

<<Gterm ver 2.50:130118>>

（3）初期化用ファイルを転送

ボードの SW1 を押しながら SW5 を押した後，両方はなす。

（ダメな場合は，DW-1 を ON にして SW5 を押す。）

[IMS]%と表示されれば #[E]

（[IMS]%でなく TM>と表示されていればそのままでＯＫ）

TM>と表示されれば `.flload romimage.mot[E]`

ファイルが転送され

[IMS]%と表示される。

（出ないときは SW5 を押す）

終了作業

```
exit[E]
```

<<SYSTEM SHUTDOWN>>と表示

Ctrl-c Ctrl-c .q で gterm を終了

（DW-1 を ON にしている場合は DW-1 を OFF にする。）

（4）ボードのリセットボタン（SW5)を押すと，初期化される。

μTeaboard の LED5 (緑) および LED1 (赤) が点灯します。

以降[IMS]%，TM>へのコマンド入力の[E]の記述は省略する。

6．モニターの利用

（1）ターゲット側と開発環境側の起動と接続

＊　ターゲット側（μTeaboard）の起動

ボードの初期化の（1）と同様

＊　開発環境側 gterm の起動

```
/home/ubuntu/te/tool/Linux-i686/etc/gterm -l /dev/ttyS0 -38400
```

＊gterm 起動

<< Gterm ver 2.50：130118 >>

＊　IMS (Initial Monitor System)に入る。

Enter キーを押す

[IMS]% と表示されれば OK

＊　注意

gterm を 2 個以上重ねて起動しないでください。

IMS のプロンプトが出ていない場合（T-Monitor のプロンプト「TM>」が出ているなど）は、μTeaboard のリセットスイッチ（ボード中央付近の SW5）を押してターゲット側（ボード）を再起動してください。

（2）コンソールの利用

＊　IMS (Initial Monitor System)

T-Kernel ベースのモニタです。プロンプトは標準では「[IMS]%」です。詳しくは『μTeaboard/AR7-AT91 取扱説明書』の「5 章 IMS」をご覧下さい（CD-ROM(Rel1.03)　トップの index.html ファイルをダブルクリックすると見ることができる。）

＊　T-Monitor

最も基本的なモニタです。ハードウェアレベルの操作などに威力を発揮します。プロンプトは「TM>」です。T-Monitor のコマンド一覧は（ターゲット側ソフトウェアの説明書と仕様書）『TMonitor 仕様書』の「3.3. コマンド一覧」をご覧下さい。さらにμTeaboard で追加された機能もあります。追加されたコマンドは（ターゲット側ソフトウェアの説明書と仕様書）『実装仕様書』の「2. T-Monitor 実装仕様」をご覧下さい。

＊　以下に IMS と T-Monitor の簡単な実習例を示します。

(1) IMS のプロンプトは以下である。

```
[IMS]%
```

(2) IMS コマンド一覧表示

```
[IMS]%?
```

(3) IMS コマンドの説明表示

```
[IMS]%? ref_tsk
```

(4) タスク一覧表示

```
[IMS]%ref_tsk
```

レポート 6-1: ref_tsk で何個のタスクが表示されたか。

レポート 6-2:ref_tsk で表示される内容「TID PRI:BPR SLT WUP SUS STS(*:NODISWAI) ST+UT(x10) RID EXINF/NAME」の取扱説明書に記載されている意味を写せ。

(5) T-Monitor への移行

```
[IMS]%#
TM>
```

(6) T-Monitor コマンド一覧表示

```
TM>?
```

(7) T-Monitor コマンド説明表示

```
TM>? ow
```

レポート 6-3：応答を書け。

(8) I/O 書き込み

T-Monitor の “ow” コマンドを使って消灯用ポーPIO_SODR(0xffff0030 番地) または点灯用ポートPIO_CODR(0xffff0034 番地) に 32 ビットの値を書き込んで、LED1~ LED4 (赤) を制御します。

LED1～4 は 1 から順に PIO (0bit~31bit)の 20,21,23,24bit に接続されている。

```
TM> ow ffff0030,00100000   * LED1 を消灯
//00100000 は 32 ビット(0000,0000,0001,0000,0000,0000,0000,0000)の 16 進数表現
TM> ow ffff0034,00100000   * LED1 を点灯
```

レポート 6-4：LED3 を点灯するコマンドを書け。

(9)終了処理

```
TM>g   *IMS へ戻る
```

```
[IMS]%exit
Ctrl-c Ctrl-c .q
```

7． T-Kernel ベースのプログラム

T-Kernel の機能を直接使うプログラムです。ハードウェア制御や割込みなどを扱うことができます。デバイスドライバなどを作成することができます。メモリ保護は効きません。メモリ空間はシステム全体で一つの空間を共有し、その中に複数の T-Kernel ベースのプログラムが配置される形です。

レポート 7-1：「μTeaboard/ARM7-AT91 取扱説明書」の「3. ソフトウェア開発方法」の章より T-Kernel ベースのソフトウェアのメモリ空間の説明よりユーザ領域のアドレス（フラッシュメモリと RAM）を調べよ。

T-Kernel ベースのプログラムは lodspg コマンドで起動を行います。この時に main() 関数が呼ばれ、第一引数 ac は渡される引数の個数を示します。通常この中でタスクやハンドラの生成などの初期化処理を行います。また unlspg コマンドでプログラムを終了することができます。この時にも main() 関数が呼ばれ、第一引数 ac はマイナスの値です。通常この中で、最初に生成したタスクやハンドラの削除などの終了処理を行います。T-Kernel ベースのプログラムから使える API については、の CDRel 1.03 内の「ターゲット側ソフトウェアの説明書と仕様書」以下のドキュメントをご参照ください。

・『T-Kernel 仕様書』

タスクやセマフォ、割込みハンドラなどのリアルタイム OS としての基本機能（T-Kernel/OS）と、デバイスドライバ管理などの機能（T-Kernel/SM）などの仕様書です。

・『ライブラリ説明書』

C 言語標準ライブラリをはじめとするライブラリの説明書です。

・『デバイスドライバ説明書』

デバイスドライバを作成したりデバイスドライバを呼び出す場合に必要となる説明書です。

・『TCP/IP マネージャ説明書』

ネットワーク（TCP/IP）のプログラミングを行う際に必要となる説明書です。

レポート 7-2：「ターゲット側ソフトウエアの説明と仕様書」－＞『T-Kernel 仕様書』の「第 1 章 T-Kernel の概要」の図 1 では，システムを何層に分けているか。また T-Kernel は下から何番目の層か。

レポート 7-3：『T-Kernel 仕様書』の目次の T-Kernel/OS の機能と T-Kernel/SM の機能より，それぞれのおもな機能はどのようなものか。

8．コンパイルと実行方法

コンパイル/リンクは、共通の makerules (etc/makerules) を利用して行います。make を行うディレクトリは、かならず tbat91 または tbat91.* の名称としてください。ディレクトリの名称に含まれる文字列により設定が変わります。tbat91 の場合は ARM7-AT91 用のリリースモードのバイナリが作成されます。tbat91.debug 等とすると ARM7-AT91 用のデバッグモードのバイナリが作成されます。gdb を使ってデバッグをする場合は tbat91.debug ディレクトリ中でコンパイルをしてください。以降の説明では、開発中の様子を示します。開発作業が終了しましたら、make install と実行することにより、各環境のインストールディレクトリ$BD/bin/tbat91 または$BD/driver/bin/tbat91 にインストールされます。

（1）/home/ubuntu/te/kappl/sample ディレクトリにあるサンプルプログラムをコンパイル，実行する。

```
cd /home/ubuntu/te/kappl/sample/tbat91
```

レポート 8-1：/home/ubuntu/te/kappl/sample/tbat91 ディレクトリにあるファイルは何か。そのファイルの内容を書け。

レポート 8-2：/home/ubuntu/te/kappl/sample/src ディレクトリにはどのようなファイルがあるか。

tbat91 デレクトリで make する。

```
make
```

レポート 8-3：make 後　/home/ubuntu/te/kappl/sample/tbat91 にはどのようなファイルがあるか。

- ボードをリセット後，tbat91 ディレクトリで gterm を起動して，TM モニタに入る。
- 実行ファイル sample.mot をボードに転送する。.mot が実行ファイルを意味する。

```
TM> .fload sample.mot ※RAM 上に Flash ROM のイメージを一時的に作成します。
```

- ボードをリセットする(今後 fload 使用後は必ずリセット)。すると IMS へ移行する。
- IMS コマンド「lodspg @アドレス」により T-Kernel ベースプログラムを実行する。

```
[IMS]%lodspg @10100000
```

レポート 8-4：端末への出力を書け。

- lodspg で起動したプログラムを終了する

```
[IMS]%unlspg @10100000
```

レポート 8-5：端末への出力を書け。

- モニタ，gterm の終了処理

```
[IMS]%exit
Ctrl-c Ctrl-c .q
```

ボードをリセットする。

レポート 8-6：main.c に記述されている 2 つの printf 文を写せ。

レポート 8-7：lodspg と unlspg では、それぞれ、main 関数の引数 ac に、どのような値が渡されたか。

レポート 8-8：「ターゲット側ソフトウェアの説明書と仕様書」の「実装仕様書」のメモリマップの記載より 10100000 は，何に割り当てられているアドレスか。

9．調査

レポート 9-1：リアルタイムＯＳについて A4，1 ページ程度で説明せよ。