５ＭＳ実験ＯＳ　第６回　マイコン(sh2)ボードのデバッグとサンプルプログラムの実行

１．GDB（デバッガ）の使用法

本 gdb はターゲットとして T-Monitor の機能を使用していますので、gdb に制御が渡されている間、ターゲットは T-Monitor に制御が移行しています。

（１）ホスト側（パソコン）の用意

gdb 用の設定ファイルを用意します。gdb は起動時のカレントディレクトリにある「.gdbinit」ファイルを設定ファイルとして読み込みます。そのファイルの内容は以下のようにしてください。

```
set remotebaud 38400
target tmon /dev/ttyS0
directory .:../src
```

各設定の意味は

・set remotebaud 38400

ターゲットとの通信スピードを指定します。ターゲット側の設定と合わせてください。

・target tmon /dev/ttyS0

ターゲットモニタを指定します。/dev/ttyS0 により通信ポートを変更することができます。linux の場合、/dev/ttyS0 が com1 を意味します。

・directory .:../src

ソースファイルを検索するディレクトリを指定します。

レポート 1-1：/home/te/kappl/sample/src/.gdbinit に記述されている内容を書け。

（２）T-Kernel ベースのソフトウエアの準備

- ターゲットプログラムはデバッグ用にコンパイルします。標準的な開発環境では tbat91.debug ディレクトリにおいて make するとデバッグ用として CFLAGS に -g が追加され、デバッグ情報付きとしてコンパイルします。
- T-Kernel ベースのソフトウエアは、T-Monitor の機能によりメモリ上にロードし、IMS の lodspg コマンドにより実行します。デバッグ中と完成後のプログラムはその格納場所の違いから、アドレスを変更する必要があります。
- 完成したプログラムは ROM 上に格納することを主に想定しています。しかも、実行中に内容が変化しない text コードは ROM 上のものをそのまま使用し、data,bss, stack など、書き変える必要のある領域は RAM 上に配置します。
- デバッグ中はブレークポイントの設定をするため text も含め、すべてのプログラム領域を RAM 上に配置する必要があります。これらの配置はリンクスクリプトとリンク時の引数によって決まります。
- $BD/lib/tbat91/static-rom.lnk は完成時のリンクスクリプトの一例です。リンク後のアドレスは、リンクスクリプト内の指定よりもリンク時の引数で指定した値が優先されますので、Makefile 内でアドレスを指定することにより、所望のアドレスに配置するのがよいでしょう。

レポート 1-2：T-Kernel ベースのプログラムのある src/Makefile の内容より(1)feq ($(MACHINE), tbat91)から，それに対応する endif までを書け。(2)これより，デバッグ中の text はメモリの何番地に置かれることになるか。

（3）/home/te/kappl/sample/でのデバッグ作業

＊　作業用ディレクトリ/home/te/kappl/sample/tbat91.debug に移動する。

レポート 1-3：このディレクトリにある Makefile の記述内容を書け。

＊　meke する。

レポート 1-4：make により表示されるメッセージの最初から-g までを書け。

レポート 1-5：make の結果作成されたファイルを書け。

（3）T-Kernel ベースのプログラムのデバッグ

sample.mot を T-Monitor の機能を使いメモリー上にロードした後、プログラム先頭にブレークポイントを設定します。その後、IMS の「lodspg@アドレス」コマンドにより実行します。実行直後、ブレークポイントによりターゲットマシンが T-Monitor に移行した後、ホスト側の gdb を起動します。

＊　ターゲットマシンの起動

ディップスイッチ（DSW1）は 4 つとも OFF になっていることを確認し，RS232C ケーブル，電源をつなぎ，リセットボタンを押す。

＊　ターゲットプログラムの実行

- ターゲットファイル sample.mot のあるディレクトリで gterm の起動
- T-Monitor に入り，以下のコマンドを実行していく

TM>.load sample.mot ※S-format 形式のデバッグ対象プログラムをメモリ上にロード

レポート 1-6：ロードされたメモリのアドレス領域を書け。

TM>b 20100000　※T-Monitor の Break コマンドでプログラム先頭にブレークポイントを設定します。

TM>g　※T-Monitor の Go コマンドで IMS に戻ります。

[IMS]% lodspg @20100000　※デバッグ対象プログラムを実行します。

レポート 1-7：表示されるメッセージを書け。

TM>.q ※gterm 終了

- デバッガ gdb を起動しデバッグする

/home/ubuntu/te/tool/Linux-i686/bin/arm-unknown-tmonitor-gdb sample

(gdb)l　※l(list)コマンドで現在位置前後を表示します。

(gdb)l　※l(list)コマンドで現在位置前後を表示します。

(gdb)b 19　※19 行目にブレークポイントを設定します。

(gdb)c　※c(continue)で実行継続します。

レポート 1-8：表示されるメッセージを書け。

(gdb)p av[i]　※p(print) により引数 av[i] の内容を表示します。

レポート 1-9：表示されるメッセージを書け。

(gdb)p i

レポート 1-10：表示されるメッセージを書け。

(gdb)n　※n(next)により 1 行実行します。

(gdb)n　※n(next)により 1 行実行します。

レポート 1-11：表示されるメッセージを書け。

(gdb)p i

レポート 1-12：表示されるメッセージを書け。

```
(gdb)c
Continuing.
from target> SYSPRG @20100000 [0]
```

※デバッグ対象のプログラムは終了してしまいましたので gdb 側に戻ってくることはありません。このような場合は Ctrl-C を 2 回入力して，Give up に y と答える。gdb のプロンプトが表示されます。

```
from target> [IMS]% ^C#[IMS]% ^CI  nterrupted while waiting for the program.
Give up (and stop debugging it)? (y or n) y
(gdb)q
```

※ q(quit) により gdb を終了します。

このようにブレークポイントをかけたまま、Ctrl-C で gdb を終了した場合、T-Monitor によって設定したブレークポイントが残っています。手動で T-Monitor に移行してブレークポイントを削除してください。

- ターゲットファイル sample.mot のあるディレクトリで gterm の起動
- T-Monitor に入り，以下のコマンドを実行していく

```
TM>b
```

※残っているブレークポイントの確認

```
TM>bc
```

※ブレークポイントの削除

```
TM>g
```

※実行継続し IMS に戻る

```
[IMS]% unlspg @20100000
```

※ T-Kernel ベースプログラムの常駐を解除するには IMS コマンド「unlspg @アドレス」を使用します

```
hello (ac=-1)
```

※プログラム的には第一引数が-1 として起動されたのと同じです。

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

※gterm を終了させる。

2．入門向けチュートリアルのＬＥＤ制御サンプル

LED の点滅パターンを制御する T-Kernel ベースのプログラムを作成します。

- ファイルの解凍、コピー、コンパイル
- 演習プログラムを置く場所として/home/ubuntu/te/ex ディレクトリを作る。
- led 用のディレクトリとしてを作る/home/ubuntu/te/ex/led ディレクトリを作る。
- CD-ROM の入門チュートリアルの実習用サンプルプログラム led.zip(jp/tutorial 内)を,led ディレクトリにコピーし、解凍（右クリック、ここに展開する）する。ls コマンドで src ディレクトリができたことを確認する。
- src ディレクトリ内の Makefile を修正する。(今後サンプルプログラムを make するときは，この修正を行う必要がある。)

```
# ロードアドレス
$(TARGET): START_ADR = -Wl,-Ttext,0x20100000,-Tdata,0x20110000
```

の Ttext,0x20100000 を Ttext,0x10100000 に修正する

- 実行ファイルを作成するためのディレクトリ/home/ubuntu/te/ex/led/tbat91 を作成し、移動する。
- tbat91 ディレククトリに、以下の 1 行を記入した Makefile を作る。

```
include ../src/Makefile
```

- make する。

レポート 2-1：make 後，tbat91 ディレククトリに出来たファイル名を書け。

(2)実行ファイルの転送と実行

- gterm を起動

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

- T-Monitor への移行する。
- 実行ファイル led.mot をボードに転送する。(第 5 回８．参照.fload)
- ボードをリセットする。すると IMS へ移行する。

レポート 2-2：プログラムを実行し(第 5 回参照 lodspg)，LED の点滅の様子を書け。（短時間で終わるので注視する。実行は何度でもできる。）

- プログラム終了(第 5 回８．参照 unlspg)，モニタと gterm の終了処理(第 5 回８．参照）をする。
- ボードをリセットする。

レポート 2-3：src ディレクトリにある main.c を，LED の点滅間隔を 0.5 秒，点滅順を１，３，２，４に変更し，make，転送，実行を行え。プログラムの変更部分を書け。

３．入門向けチュートリアルのネットワーク送受信サンプル

ネットワーク送受信サンプル net.zip(jp/tutorial 内)に対して，前章のＬＥＤ制御サンプルと同様のことを行う。異なるのは main.c の自 IP アドレスの値を変更する（変更する値は教室中央よりの使用していないパソコンで windows を起動し，windows システムツールのコマンドプロプトで ipconfig コマンドを実行し IP アドレスを調べそのアドレスとする。起動したパソコンの LAN ケーブルを抜く）のと，実行する前に LAN ケーブル（机の下の DELL またはＨＰ）のパソコンに繋がっているもをを使用）をボードに繋ぐ点である。

実行後，Web ブラウザ（Linux の firefox)で http://設定した IP アドレス にアクセスする。

レポート 3-1：ブラウザからアクセスしたときに，端末(ブラウザではない）に表示されるメッセージを上から 5 行書け。

４．調査

レポート 4-1：ITRON Project について A4，１ページ程度で説明せよ。