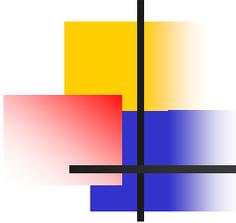


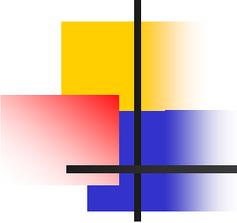
カーネルソースの歩き方 RaspberryPi演習

TOPPERS/SSPカーネルソース勉強会
アライブビジョンソフトウェア株式会社 高橋和浩



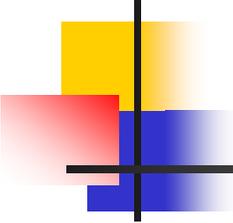
概要

TOPEPRS/SSP編 TOPPERS/SSP(smallest set profile)は、ITRONの最小セットプロファイルに対応した「待ち」状態のないカーネルの公開ソースコードです。昨年から名古屋市工業研究所の主催で継続中のカーネルソース勉強会の成果として、RaspberryPiへの移植版を実際に演習します。



目標 目的

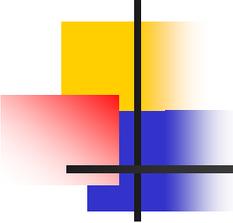
- Linuxが動くものをベアメタルベースのTOPPERSで動作させる目的は？
- 例えば....
- RaspberryPi買ったがあまり使っていない
- マイコンボードと共用したい。何も無い状態でLEDチカしたい
- ARMの学習、ARMアセンブラの学習



動かすまでの流れその1

1. 必要機材

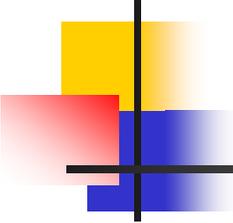
- 1) RaspberryPi B /B+ /2のいずれか
- 2) TTLシリアルケーブル (PL2303HX)
- 3) SDカード(B+ 2の場合はmicroSD)
100MBもあれば十分



動かすまでの流れその2

2. 必要なソフトのダウンロード

- 1) RaspberryPI用SHRINK版SSPフル
- 2) RaspberryPI用ブートローダーファイル
- 3) SDFormatter
- 4) teraterm
- 5) PL2303HXドライバ
- 6) ARM用クロスコンパイラ
- 7) eclipse Pleiades All in One 日本語ディストリビューション



動かすまでの流れその3

3.ビルド環境の構築

Windows環境での方法を紹介します

4.SDカードの準備

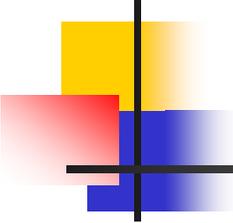
5.シリアルドライバとターミナルソフトの設定

6.ビルドと実行

機材 (TTLシリアルケーブルなど)

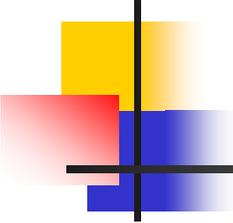
- SWEST会場では、ケーブルをいくつか貸し出しています。5V電源をとる場合は接続にご注意ください。





ビルド環境構築

- 1.Eclipse Windows環境
オールインワンで、MingWも同時セットアップされます。IDE環境とCUIといずれも使用できます。
- 2.VMware上のubuntu 32bit環境
- ARMクロスコンパイラもセットアップ済
- 上記いずれかですが、前者Eclipse環境をベースに説明をします。以降**CUIでの説明**になります。またVmwareは付録ページ参照ください。



Pleiades All in One 日本語ディストリビューションのダウンロードと設定その1

以下のURLの

- C/C++
- 64bit
- Full Edition

をダウンロードします。

バージョンは Eclipse 4.3.2 Kepler ケプラー
SR2 for Windows ベース です

<http://mergedoc.osdn.jp/#pleiades.html>

Pleiades All in One 日本語ディストリビューションのダウンロードと設定その2

MergeDoc Project

Pleiades プラグイン日本語化
IStyle 改行タブ表示プラグイン

フォーラム
チケット
プロジェクト Wiki
Qiita
ブログ

Pleiades All in One 日本語ディストリビューション (zip) ダウンロード

Pleiades All in One 4.3.2.v20140321
Eclipse 4.3.2 Kepler ケブラー SR2 for Windows ベース

- 開発対象となる言語に合わせてパッケージをダウンロードしてください。
- Java パッケージ Full Edition は Vector の自己解凍書庫 (.exe) にリダイレクトされます。
- Full Edition には Eclipse 実行用の JRE や各言語の処理系が含まれています。
よく分からない場合は Full Edition を選んでください。

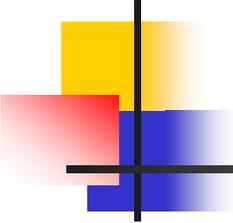
- plugins、features ディレクトリーに格納されたプラグイン
- dropins ディレクトリーに格納されたプラグイン
- Eclipse 実行用の JRE や各言語のコンパイラー、ランタイムなどの処理系

| | | PlatformUltimate | Java | C/C++ | PHP | Python |
|-------|------------------|------------------|----------|----------|----------|----------|
| 32bit | Full Edition | Download | Download | Download | Download | Download |
| | Standard Edition | Download | Download | Download | Download | Download |
| 64bit | Full Edition | Download | Download | Download | Download | Download |
| | Standard Edition | Download | Download | Download | Download | Download |

Eclipse 実行用 JRE 7 s s s s s s

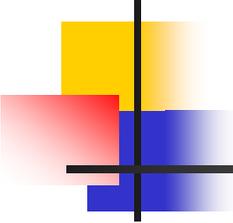
開発対象用 JDK [6u45](#)、[7u51](#) s s

SWEST17 カーネルソースの歩き方
RaspberryPi演習



Pleiades All in One 日本語ディストリビューションのダウンロードと設定その3

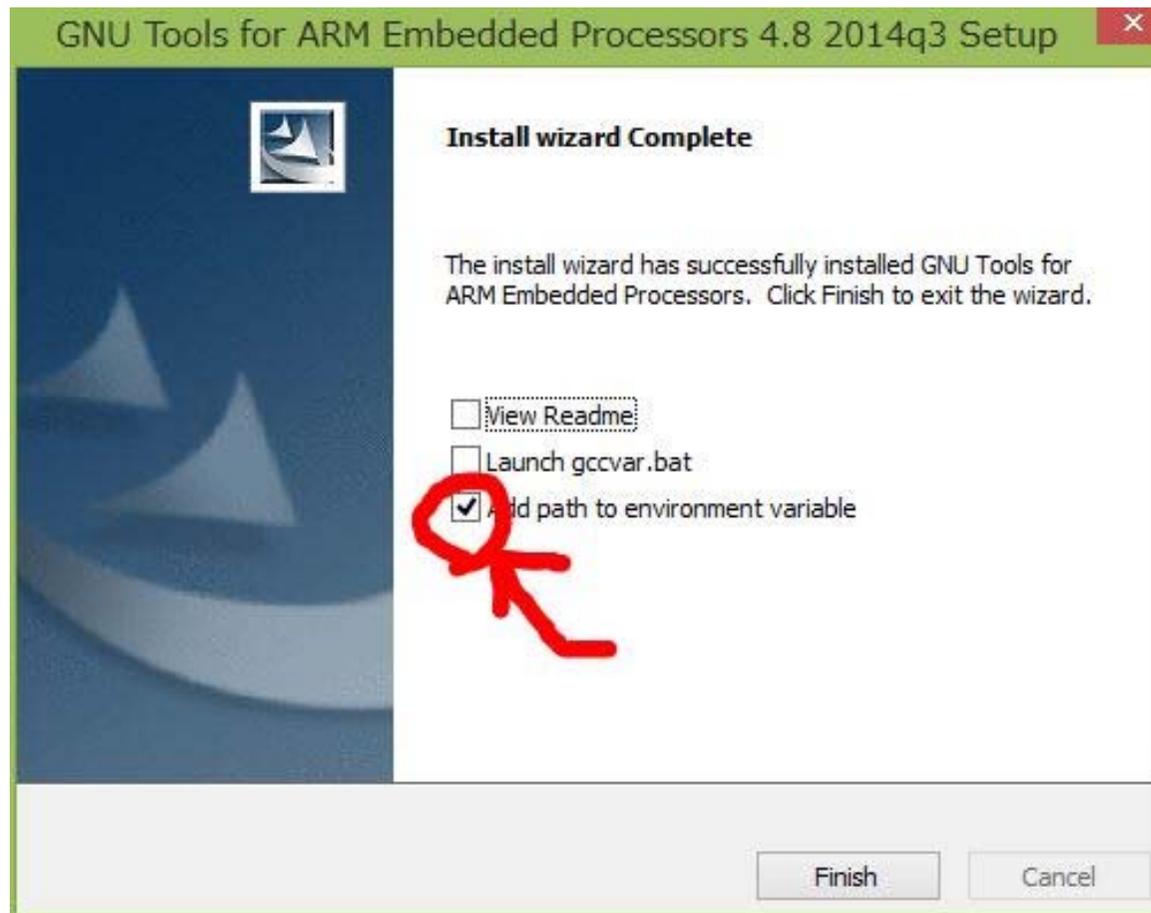
- Zipファイルを解凍して、そのまま使います。
- CUIを使う場合は、
- 解凍フォルダ¥eclipse¥mingw¥msys. bat
- 参考までにIDEを使う場合は、
- 解凍フォルダ¥eclipse¥eclipse. exe



コンパイラのダウンロードと設定その1

- GCC ARM Embedded 以下のバージョンをダウンロード
<https://launchpad.net/gcc-arm-embedded/4.8/4.8-2014-q3-update>
- 以下のファイルをクリックしてインストール
[gcc-arm-none-eabi-4_8-2014q3-20140805-win32.exe](#)
- インストール時にPATHの設定のチェックを入れます

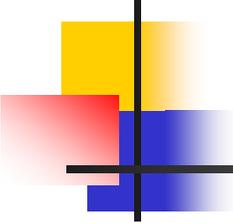
コンパイラのダウンロードと設定その2



SWEST17 カーネルソースの歩き方
RaspberryPi演習

MINGW32:~

```
takahashi@Acer-i5 ~  
$ arm-none-eabi-gcc -v  
Using built-in specs.  
COLLECT_GCC=d:\usr\ARM-GCC48\bin\arm-none-eabi-gcc.exe  
COLLECT_LTO_WRAPPER=d:/usr/arm-gcc48/bin/./lib/gcc/arm-none-eabi/4.8.4/lto-wrap  
per.exe  
Target: arm-none-eabi  
Configured with: /home/build/work/GCC-4-8-build/src/gcc/configure --build=i686-l  
inux-gnu --host=i686-w64-mingw32 --target=arm-none-eabi --prefix=/home/build/wor  
k/GCC-4-8-build/install-mingw --libexecdir=/home/build/work/GCC-4-8-build/instal  
l-mingw/lib --infodir=/home/build/work/GCC-4-8-build/install-mingw/share/doc/gcc  
-arm-none-eabi/info --mandir=/home/build/work/GCC-4-8-build/install-mingw/share/  
doc/gcc-arm-none-eabi/man --htmldir=/home/build/work/GCC-4-8-build/install-mingw  
/share/doc/gcc-arm-none-eabi/html --pdfdir=/home/build/work/GCC-4-8-build/instal  
l-mingw/share/doc/gcc-arm-none-eabi/pdf --enable-languages=c,c++ --disable-decim  
al-float --disable-libffi --disable-libgomp --disable-libmudflap --disable-libqu  
admath --disable-libssp --disable-libstdcxx-pch --disable-nls --disable-shared -  
-disable-threads --disable-tls --with-gnu-as --with-gnu-ld --with-headers=yes --  
with-newlib --with-python-dir=share/gcc-arm-none-eabi --with-sysroot=/home/build  
/work/GCC-4-8-build/install-mingw/arm-none-eabi --with-libiconv-prefix=/home/bui  
ld/work/GCC-4-8-build/build-mingw/host-libs/usr --with-gmp=/home/build/work/GCC-  
4-8-build/build-mingw/host-libs/usr --with-mpfr=/home/build/work/GCC-4-8-build/b  
uild-mingw/host-libs/usr --with-mpc=/home/build/work/GCC-4-8-build/build-mingw/h  
ost-libs/usr --with-isl=/home/build/work/GCC-4-8-build/build-mingw/host-libs/usr  
 --with-cloog=/home/build/work/GCC-4-8-build/build-mingw/host-libs/usr --with-li  
belf=/home/build/work/GCC-4-8-build/build-mingw/host-libs/usr --with-host-libstd  
cxx='-static-libgcc -Wl,-Bstatic,-lstdc++,-Bdynamic -lm' --with-pkgversion='GNU  
Tools for ARM Embedded Processors' --with-multilib-list=armv6-m,armv7-m,armv7e-m  
,armv7-r  
Thread model: single  
gcc version 4.8.4 20140725 (release) [ARM/embedded-4_8-branch revision 213147] (  
GNU Tools for ARM Embedded Processors)
```



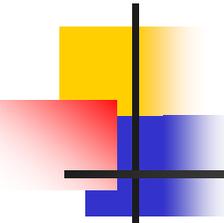
シリアルドライバとターミナルソフトの設定その1

- PL2303HXは最新版はWindows8 8.1は対応していませんが、旧バージョンが利用できます。
- 注意:設定変更により、Windowsに支障がある場合もありますのでご注意ください。
- ドライバ以下のURLの v1.5.0を利用ください
- <http://akizukidenshi.com/catalog/faq/goodsfaq.aspx?goods=M-00720>
- 以下のファイルを解凍、
- PL2303_Prolific_DriverInstaller_v1.5.0.zip
- 以下のファイル(インストーラ)を実行
- PL2303_Prolific_DriverInstaller_v1.5.0
- デバイスマネージャーで、動作していない場合は、ドライバファイルを指定しなおしてください
- 結果的に次のバージョンが確認できればOKです

シリアルドライバとターミナルソフトの設定その2

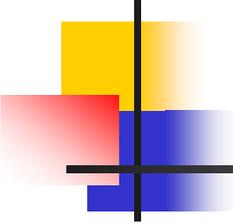


SWEST17 カーネルソースの歩き方
RaspberryPi演習



シリアルドライバとターミナルソフトの設定その3

- Teratermをインストールします。



SDカードの準備その1

- SDカードは、FAT32にてフォーマット
SDcardformatter 4.0でFAT32でフォーマット
https://www.sdcard.org/jp/downloads/formatter_4/
- 以下のURLからブートローダーをダウンロード
<https://github.com/raspberrypi/firmware/tree/master/boot>

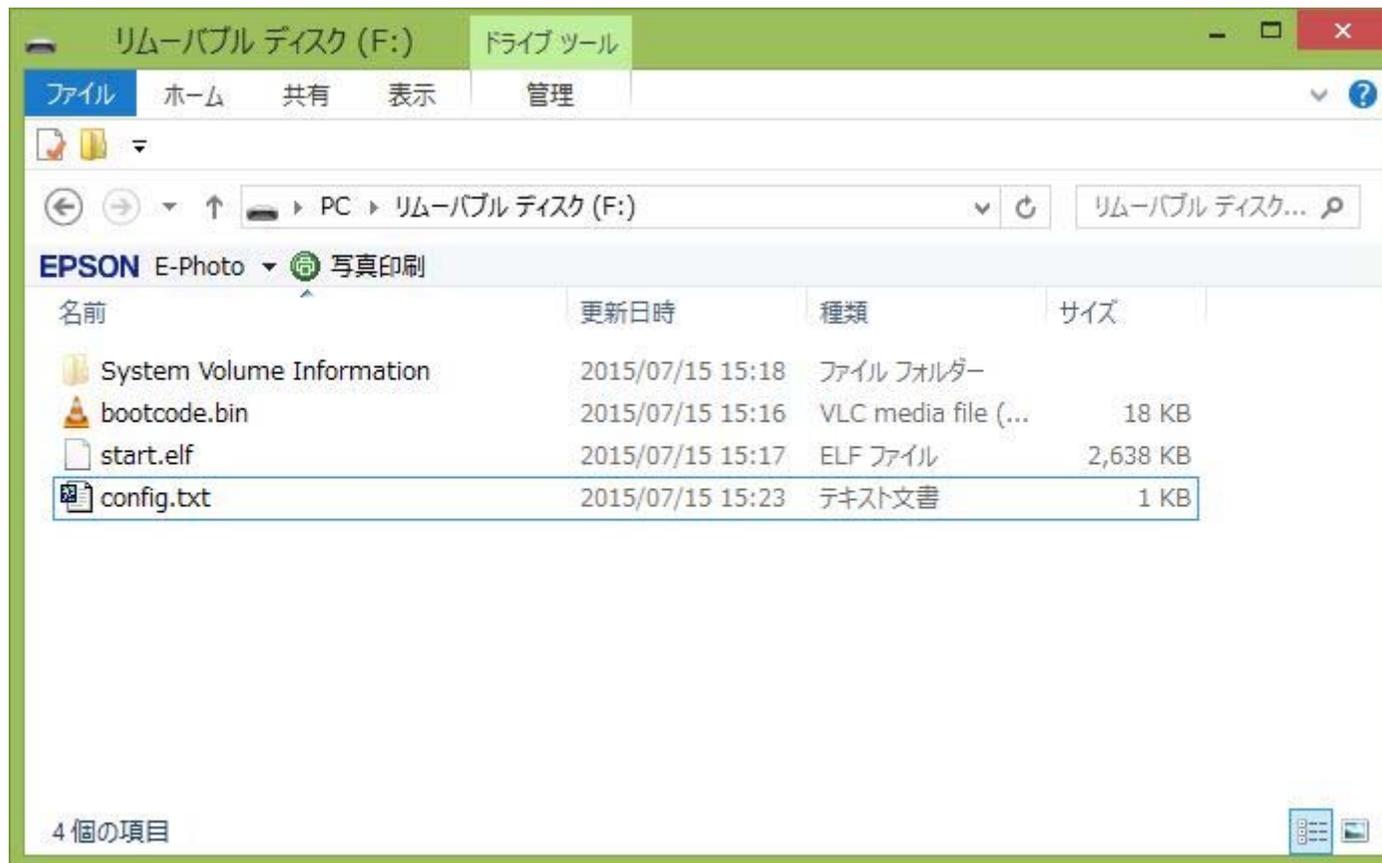
Bootcode.bin と start.elf の2つ

この2つのファイルをSDカードにコピー

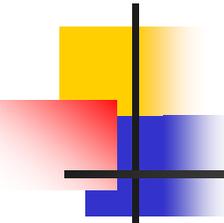
- さらに config.txt というファイル名に中身を以下の内容のものをSDカードに置く

```
kernel=rpi-micon.img
```

SDカードの準備その2



SWEST17 カーネルソースの歩き方
RaspberryPi演習



ソースのダウンロードとビルドその1

RaspberryPI用SHRINK版SSPフルをダウンロードして解凍します。

<https://github.com/alvstakahashi/RPI-SHRINK-SSP-FULL>

「Download ZIP」ボタンでダウンロードします。

ソースのダウンロードとビルドその2

alvstakahashi / RPI-SHRINK-SSP-FULL

7 commits 1 branch 0 releases 1 contributor

branch: master RPI-SHRINK-SSP-FULL / +

| File Name | Commit Hash | Time |
|------------------------------------|------------------|---------------|
| Update README.md | ee83cc9306 | latest commit |
| SSP_TimerIRQ | 20150618 | 7 hours ago |
| LICENSE | 20150618 | 7 hours ago |
| README.md | Update README.md | 6 hours ago |
| raspberry pi用 sspカーネルシュリンク版の紹介.ppt | 20150618 | 6 hours ago |

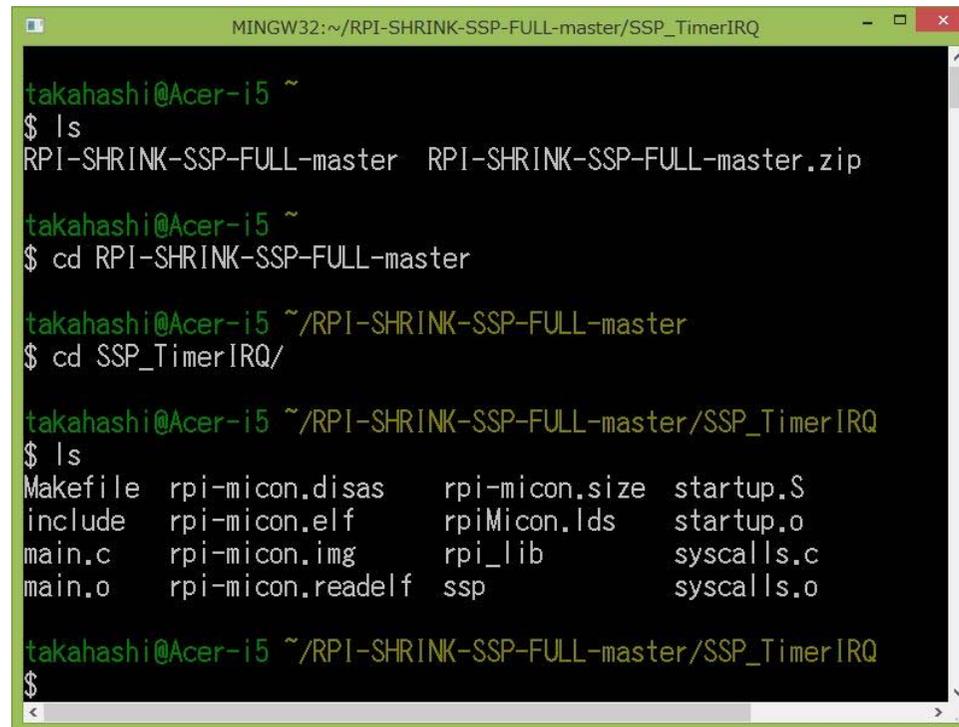
RPI-SHRINK-SSP-FULL

BareMetalで遊ぶRaspberryPi[達人出版]の環境にて、SSPのシュリンク版を移植したものです。

Clone in Desktop
Download ZIP

ソースのダウンロードとビルドその3

- Mingwのhomeに解凍した場合



```
MINGW32:~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
takahashi@Acer-i5 ~
$ ls
RPI-SHRINK-SSP-FULL-master  RPI-SHRINK-SSP-FULL-master.zip

takahashi@Acer-i5 ~
$ cd RPI-SHRINK-SSP-FULL-master

takahashi@Acer-i5 ~/RPI-SHRINK-SSP-FULL-master
$ cd SSP_TimerIRQ/

takahashi@Acer-i5 ~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
$ ls
Makefile  rpi-micon.disas  rpi-micon.size  startup.S
include   rpi-micon.elf   rpiMicon.lds   startup.o
main.c    rpi-micon.img   rpi_lib        syscalls.c
main.o    rpi-micon.readelf  ssp            syscalls.o

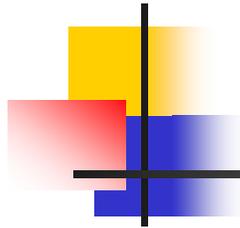
takahashi@Acer-i5 ~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
$
```

ソースのダウンロードとビルドその4

- \$make clean 後 make します。
- \$make RPITYPE=RP12

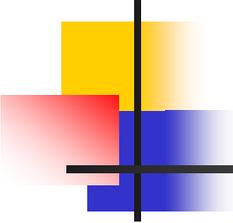
```
MINGW32:~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
takahashi@Acer-i5 ~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
$ make clean
rm -f -f *.o *.img *.elf **/*.o **/*.o
rm -f -f tags *~

takahashi@Acer-i5 ~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
$ make RPITYPE=RP12
arm-none-eabi-gcc -mcpu=arm1176jzf-s -mtune=arm1176jzf-s -mfloat-abi=softfp -O0
-ggdb -g -mlittle-endian -I./ssp -I./include -nostartfiles -ffreestanding -O0 -
g -gdwarf-2 -DRP12 -c startup.S -o startup.o
arm-none-eabi-gcc -mcpu=arm1176jzf-s -mtune=arm1176jzf-s -mfloat-abi=softfp -O0
-ggdb -g -mlittle-endian -I./ssp -I./include -nostartfiles -ffreestanding -O0 -
g -gdwarf-2 -DRP12 -c main.c -o main.o
main.c:9:12: warning: extra tokens at end of #ifdef directive [enabled by default]
#ifdef RPIB-PLUS
^
arm-none-eabi-gcc -mcpu=arm1176jzf-s -mtune=arm1176jzf-s -mfloat-abi=softfp -O0
-ggdb -g -mlittle-endian -I./ssp -I./include -nostartfiles -ffreestanding -O0 -
g -gdwarf-2 -DRP12 -c syscalls.c -o syscalls.o
syscalls.c: In function '_sbrk':
syscalls.c:141:23: warning: comparison between pointer and integer [enabled by default]
if (heap_end + incr > stack_ptr)
```



```
MINGW32:~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
arm-none-eabi-gcc -mcpu=arm1176jzf-s -mtune=arm1176jzf-s -mfloat-abi=softfp -O0
-ggdb -g -mlittle-endian -I./ssp -I./include -nostartfiles -ffreestanding -O0 -
g -gdwarf-2 -DRPI2 -c ssp/startup.c -o ssp/startup.o
arm-none-eabi-gcc -mcpu=arm1176jzf-s -mtune=arm1176jzf-s -mfloat-abi=softfp -O0
-ggdb -g -mlittle-endian -I./ssp -I./include -nostartfiles -ffreestanding -O0 -
g -gdwarf-2 -DRPI2 -c ssp/task.c -o ssp/task.o
arm-none-eabi-gcc -mcpu=arm1176jzf-s -mtune=arm1176jzf-s -mfloat-abi=softfp -O0
-ggdb -g -mlittle-endian -I./ssp -I./include -nostartfiles -ffreestanding -O0 -
g -gdwarf-2 -DRPI2 -c ssp/task_manage.c -o ssp/task_manage.o
arm-none-eabi-gcc -static -nostartfiles -T rpiMicon.lds startup.o main.o syscall
s.o rpi_lib/rpi_init.o rpi_lib/gpio/gpio.o rpi_lib/timer/timer.o rpi_lib/delay/d
elay.o rpi_lib/serial/serial.o rpi_lib/bss/clearbss.o ssp/rpi_prc_support.o ssp/
kernel_cfg.o ssp/startup.o ssp/task.o ssp/task_manage.o -o rpi-micon.elf -lc
-lm -lgcc
arm-none-eabi-objdump -D rpi-micon.elf > rpi-micon.disas
arm-none-eabi-size rpi-micon.elf > rpi-micon.size
arm-none-eabi-readelf -a rpi-micon.elf > rpi-micon.readelf
arm-none-eabi-objcopy -O binary rpi-micon.elf rpi-micon.img

takahashi@Acer-i5 ~/RPI-SHRINK-SSP-FULL-master/SSP_TimerIRQ
$
```

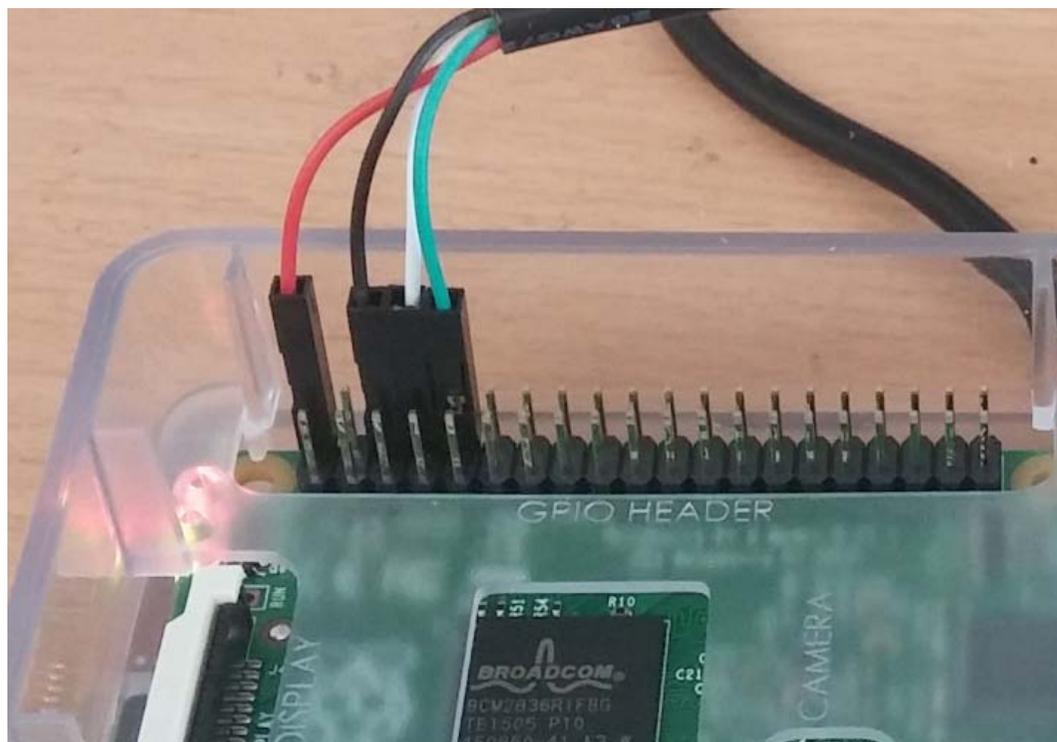


プログラムの実行その1

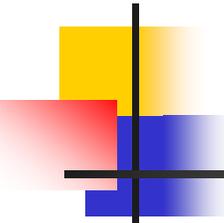
- SDカードに rpi-micon.img をコピー
- TTLシリアルケーブル接続
- PIN 2(5V) -- 接続しません
- PIN 6(GND) -- 黒 GND
- PIN 8(TX) -- 白 (RX)
- PIN 10(RX) -- 緑 (TX)

プログラムの実行その2

- 電源が入るとプログラムは動作します。



SWEST17 カーネルソースの歩き方
RaspberryPi演習



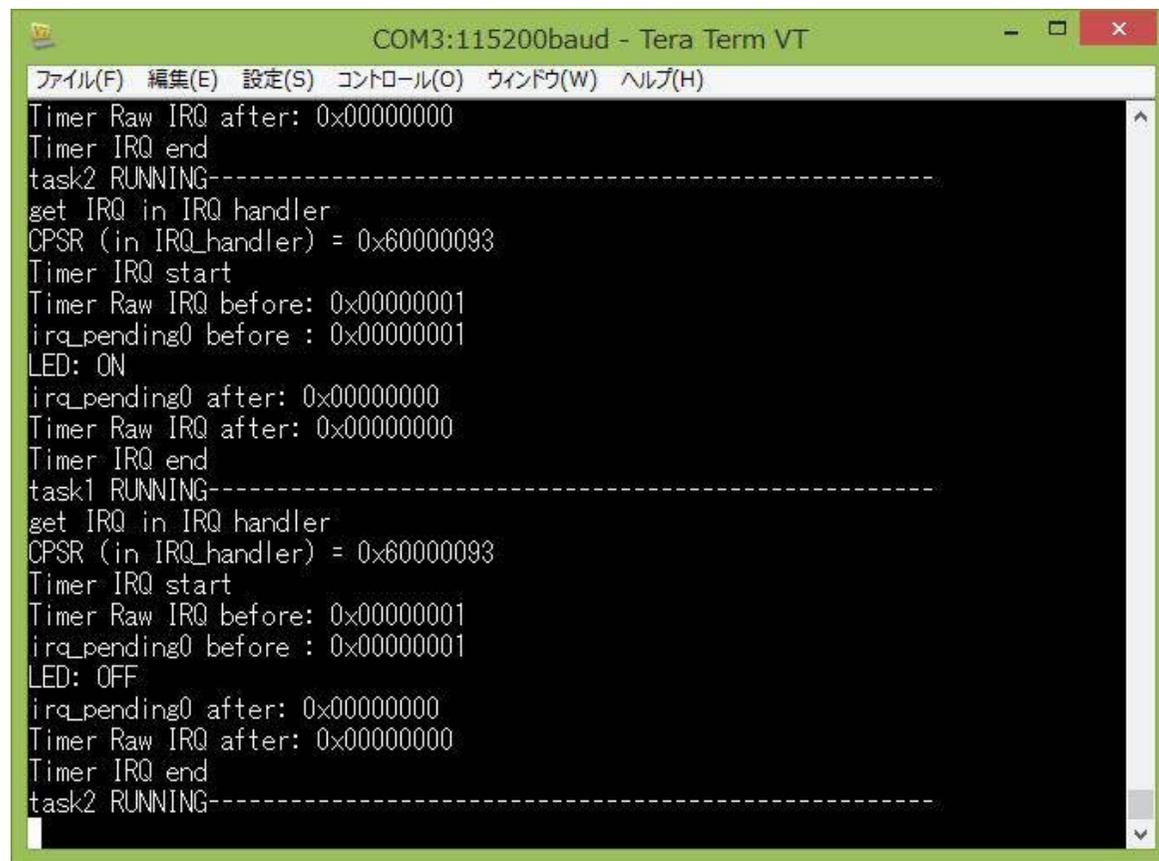
プログラムの実行その3

1)ターミナルの準備

- USBシリアルポート番号を確認して、そのポートでターミナルソフトを起動します。
- 設定
- 115200BPS 8none1bit フロー制御なし
- 端末はLFコード

プログラムの実行その4

お疲れ様でした。



```
COM3:115200baud - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
Timer Raw IRQ after: 0x00000000
Timer IRQ end
task2 RUNNING-----
get IRQ in IRQ handler
CPSR (in IRQ_handler) = 0x60000093
Timer IRQ start
Timer Raw IRQ before: 0x00000001
irq_pending0 before : 0x00000001
LED: ON
irq_pending0 after: 0x00000000
Timer Raw IRQ after: 0x00000000
Timer IRQ end
task1 RUNNING-----
get IRQ in IRQ handler
CPSR (in IRQ_handler) = 0x60000093
Timer IRQ start
Timer Raw IRQ before: 0x00000001
irq_pending0 before : 0x00000001
LED: OFF
irq_pending0 after: 0x00000000
Timer Raw IRQ after: 0x00000000
Timer IRQ end
task2 RUNNING-----
```

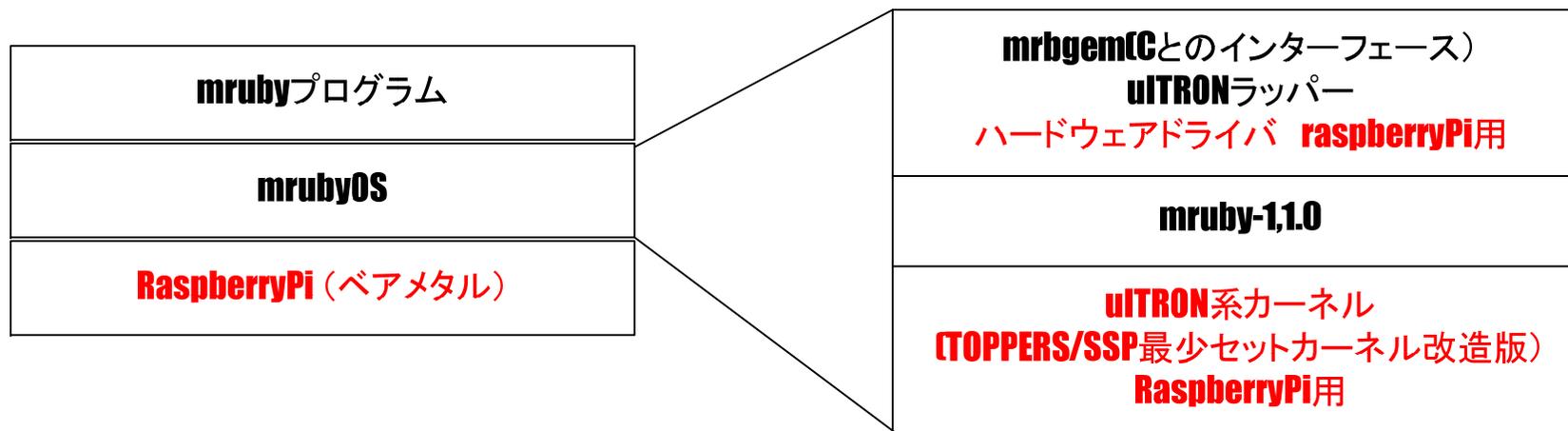
SWESTIT / カーネルソースの歩き方

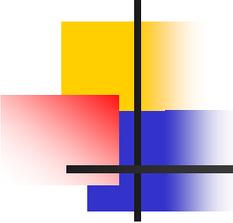
RaspberryPi演習

今後の展開(やりたいこと)

■ OS無環境でのmrubyのプラットフォーム

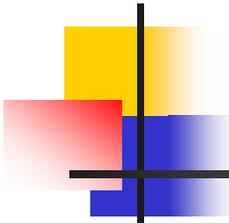
ソフトウェア構成 **raspberryPi**の場合





ありがとうございました。

- [https://github.com/alvstakahashi\(Other Service\)](https://github.com/alvstakahashi(Other Service))
- 弊社ウェブサイト
- <http://homepage3.nifty.com/ALVS/>
- Facebook
- <https://www.facebook.com/firstandlastandallways>



付録

VMwareのubuntu 32bitを使う場合その1

- Pleiades All in One 日本語ディストリビューションを使わない場合は、こちらの方法でビルド環境を利用します。
- クロスコンパイラの設定済みですので、ファイルが大きいですが、設定は基本的にありません。
- 必要なソフト
- VMwarePlayer 6.x (非商用無償版)
- VMwarePlayerのインストールは割愛します。

付録

VMwareのubuntu 32bitを使う場合その2

- 以下から、VMwareイメージをダウンロードします。
- http://dev.toppers.jp/trac_user/contrib/browser/CROS-TOOL-VMWARE/Ubuntu1404%20x86%20for%20ARM%20CROSS-4.8.7z
- VMwarePlayerから、仮想マシンを開いてください。
Ubuntu1404 x86 for ARM CROSS.vmx
- ユーザー toppers パスワード toppers
- USB2.0のUSBメモリアダプタなら、仮想マシンから直接SDカードにアクセス可能です。
- 起動後再度オプションの共有フォルダを設定するとWindowsとファイル交換できます。
- PL2303HXのドライバもインストールされているのでgtktermのインストールすればターミナルも利用できます。