ラズベリーパイはいかが？

—ARM への招待—

Why don't you try the Raspberry Pi？

—Invitation to the ARM World－

ネットワーク情報学部　石原秀男
School of Network and Information　Hideo ISHIHARA

Keywords: ARM, mbed, Raspberry Pi, Arduino Due

Abstract

During last two decades, the ARM is the most popular brand in the fields of embedded systems. The ARM is a RISC CPU architecture designed by ARM Holdings. Because of its lower power consumption, it is widely used in various devices such as smartphones, digital televisions, and mobile computers. This paper surveys three different ARM based micro controller boards, the mbed, the Raspberry Pi and the Arduino Due.

１．　はじめに

いきなりの質問だが，「最も売れている CPU は？」と聞かれたら何と答えるだろうか。ブランドとしてなら、Intel で正解だ。しかし、アーキテクチャ（＝設計）となると話は違う。x86 や x64 では不正解だ。

Rank	Company	Sales（$M）
1	Intel	49,697
2	Samsung	33,483
3	TSMC	14,600
4	TI	12,900
5	Toshiba	12,745
6	Renesas	10,653
7	Qualcomm	9,910
8	ST	9,631
9	Hynix	9,403
10	Micron	8,571

Table.1　2011 Semiconductor Sales Leaders

Semiconductor Sales Leaders

Table.1 は，文献[1]から作成した半導体業界の売り上げ上位 10 社だが，見てのとおり 1 位は Intel である。確かに CPU のトップブランドは Intel だ。

ところで，表からそのIntelと，メモリ専業のHynix，Micronを除いた残りの 7 社には共通点がある。程度の差こそあるもののARMアーキテクチャのCPUに関わっている[1]ことだ。しかも各社の業績は，ARMとの関係の深さを映す鏡のようでもある。10 社の中で最も成長率が高く，対前年比＋38%もの売り上げを達成しているQualcommはARMベースのスマートフォン向けチップセットを開発している。一方，経営難に直面しているRenesasがARMに参入したのは，2012 年になってからのことだ。

ARMは英ARM社が開発している 32bit[2]のRISCアーキテクチャである。ARM社は，基本設計だけを行いそのライセンスを販売するというプロセッサIP[3]に特化した企業であり，自身は製造を行っていない。2011 年度の売上も 7 億 8500 万ドルに過ぎず，一般的な知名度はIntel

[1] TSMC は製造専門のファウンドリだが，Qualcomm や NVIDIA 等の製造を担当していることが知られている。
[2] もちろん最近では 64bit 版もある。
[3] Intellectual Property，知的財産。

ほど高くない。しかし低消費電力であることや，ライセンスを受ければ自由に改変して自社製SoC[4]に搭載できることなどから，PC以外の用途では広く使われている。たとえば，Qualcomm のSnapdragon[5]，iPodやiPadのA5，A6，NVIDIAがタブレット用に開発したTegra，そしてNINTENDO DSやPlay Station VITAなどはすべてARMアーキテクチャである。もちろんARMは，自動車，電気機器，産業用機械などの多くにも搭載されている。その結果，2011 年のARM CPUの販売台数は 79 億にも達した。同時期のIntel製CPUが 3 億 3000 万であることと比べれば，ARMのすごさがわかるだろう。スマートフォン用プロセッサの 95%[2]はARM系だが，2013 年には，PCの市場規模を携帯電話が超える[3]とされる。今やPCの時代＝Intelの時代が終わりを迎え，スマホの時代＝ARMの時代が到来しようとしているのである。

Intel の CPU は高性能の代償として消費電力が大きく，マイコン的な用途には向いていない。しかし ARM ならばメイン CPU とマイコンが同じアーキテクチャということが現実になる。そうなれば負荷の重い処理をこなしながら I/O を行うことも，マイコンで可能になるだろう。その意味で ARM は，コンピュータ処理に新たな世界を切り開く存在でもある。近年，マイコン開発用に複数の ARM ベースのボードが入手できるようになってきた。その中には，もはやマイコンボードではなく，PC そのものという製品もある。本稿では，それらのいくつか紹介しよう。

２． 32bit マイコンボード mbed

2.1. mbedシリーズ

mbedは，蘭Philips社の半導体部門から独立したNXPセミコンダクタ社が製造するマイコンボードである。複数のバージョンが存在するが，数多く流通しているのは Cortex-M0 コアのNXP LPC11U24 と，Cortex-M3 コアのNXP LPC1768 である。ARMではMシリーズはローエンドの組み込み用だが，M0 とM3 は基本設計が異なっており，処理能力もM0 の 0.9DMIPS[6]/MHzから，M3 では 1.25DMIPS/MHzへと強化されている。クロック自体が，LPC11U24 の 48MHzからLPC1768 では 96MHzに引き上げられていることもあって，両者の性能差は小さくない。販売価格には 1000 円ほど差がある[7]が，高性能のLPC1768 を選ぶべきだろう。コストにこだわるなら，バイナリレベルで互換性のあるLPCXpresso NXP1769[8]も魅力的だが，mbedの最大の特徴であるブラウザ上の開発環境は使えない。

LPC1768 を購入すると，箱の中には mbed 本体，マイクロ B タイプの USB ケーブル，B6 サイズ 1 枚の英文セットアップガイド，そしてピン配置の書かれた名刺大のカード 2 枚（紙とプラスチック）が入っている。これ以外には，インターネットに接続できる PC を用意するだけでよい。プログラム開発は，その PC と付属の USB ケーブルで接続して行うことになる。形状は，通常より幅広の 40 ピン DIP パッケージ（54×27 ㎜ほど）で，ブレッドボードに直接挿すことも可能だ。基板上にはプログラムから制御可能の LED が 4 個装着されており，動作確認程度であれば他に回路を用意する必要もない。

2.2. Arduinoとmbed

今や最もポピュラーなマイコンボードである Arduino だが、万能というわけではない。mbed のような高性能ボードとの差を理解して使い分けるべきだろう。Table.2 は Arduino の代表としての Uno と，LPC 1768 を比較したものである。

	Arduino Uno	mbed LPC1768
CPU	ATmega328 (AVR) 8bit RISC	NXP 1768 (ARM Cortex M3) 32bit RISC
Clock Speed	16MHz	96MHz
Program area	32KB	512KB
Data area	2KB	32KB
EEPROM area	1KB	－
Digital I/O	×20	×26
Analog In	×6	×6
Analog Out	－	×1
PWM Out	×6	×6
Serial	×1	×3
SPI	×1	×2
I2C	×1	×2
CAN	－	×1
Ethernet	－	×1
USB	－	×1
Voltage	5.0V	3.30V

Table.2 Arduino Uno and LPC 1768

表からわかるように，Arduino の 8bit 16MHz に対して mbed は 32bit 96MHz であり，CPU のスペックが全く違う。詳細は後で述べるが，両者の性能差はクロック以上だ。さらに注目すべきなのは，プログラム領域とデータ領域の容量である。Arduino のデータ領域は 2KB

---

[4] System on Chip，CPU とメモリや I/O などを一体化させワンチップにしたもの。
[5] Snapdragon はチップセットで，CPU は Scorpion。
[6] Dhrystone Million Instruction Per Second
[7] 秋月電子の価格は 11U24 が 4200 円，1768 が 5200 円。
[8] 1769 ではクロックが 120MHz に引き上げられており，秋月電子の価格は 2500 円。

しかなく，2 バイトの int 型 1000 個分しかない。補助的スペースとして EEPROM が 1KB あるが，合わせても 1500 個だ。mbed ならば 4 バイトの int 型 8000 個分はあるわけで，用途によってはこの差は大きい。

入出力に関しては， mbed にアナログ出力（DAC）というアドバンスがあるくらいで，ほぼ同レベルである。

インタフェースについては，シリアル，SPI，I2Cなどの数が多いだけでなく，CAN[9]まであり，コネクタは実装されていないが，EthernetやUSBホスト機能を備えるなどmbedの充実ぶりが著しい。

なお，mbed の動作電圧は 3.3V である。入力ピンへの 5V 印加はボードにダメージを与える可能性があるので Arduino（＝5V）の経験者は注意してもらいたい。

2.3. 開発環境

mbed のプログラム開発は，従来のマイコン開発とは一線を画す画期的なものである。すべての開発環境はクラウドにあり，インターネット経由で利用するのだ。ローカルに必要なのはブラウザだけで，ソースもクラウド上に保管される。mbed の使用法については日本語による丁寧な解説がある[4]ので，全くマイコンを触った経験がない人でも，LED の点滅くらいなら 30 分もあればできるだろう。秋葉原で RJ45 コネクタを手に入れてくれば，mbed から Twitter への投稿も難しくはない。

詳細は文献[4]を参照してもらうことにして，ここでは手順だけを簡単に説明しておこう。

1. 付属の USB ケーブルで mbed を PC に接続。
2. PC からは USB メモリとして認識される。
3. mbed 内にある MBED.HTM を開く。
4. 初回は開いたページで Signup する。
   （二回目以降は登録したユーザ名で Login）
5. 右上メニューの Compiler をクリック。
6. 左上メニューの New をクリックし，作成するプログラムの名前を設定する。
7. ワークスペースにある main.cpp（ひな形としてLEDの点滅プログラムが自動的に作られるようだ）を変更して各自のプログラムを作成。
8. 中央上メニューの Compile をクリック。
9. コンパイルに成功すると実行ファイル（*.bin）が自動的にダウンロードされる。
10. ダウンロードされた*.bin を mbed にコピー。
11. mbed の中央にあるリセットボタンを押して実行。

一連の作業の中で，4 から 8 までの操作はブラウザを通してクラウド上で行われる。プログラミングそのものは C/C++言語で行うが，入出力関係のライブラリの使用法については Handbook のページに書かれている。また Cookbook のページには，様々な応用例もあるので参考になるだろう。

もちろん，クラウドを使わずに LPCXpresso の開発環境をダウンロードしてローカルだけで開発を行うことも可能なようだ。

2.4. パフォーマンス

デジタル出力を行うプログラムを作成して，Arduino と mbed の処理速度の差を調べた。

Arduino用に作成したのが、次のProgram.1 である。スペースの都合上一部省略してあるが，13 番ピンの HIGH，LOWを交互に 50 回ずつ繰り返しており，13 番ピンに直結するボード上のLEDが点滅を繰り返す[10]。合計すると，startからendまでの間には 10 メガ回のデジタル出力が行われていることになる。

```
long int i, start, end;

void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  start = millis();
  for(i=0;i<100000;i++){
    digitalWrite(13,HIGH);
    digitalWrite(13, LOW);
    //省略してあるが、HIGH と LOW を
    //交互に 49 回（計 50 回）ずつ繰り返し
  }
  end = millis();
  Serial.println(start);
  Serial.println(end);
}
```

Program.1 led.ino（Arduino）

測定を繰り返したところ、start と end の差はほぼ 39712 で一定していた。millis()の単位はミリ秒なので 39.712 秒ということになる。

同様の内容をmbedで行うのが，Program.2 である。こちらはボード上のLED1 を点滅させているが，mbed にはArduinoのシリアルモニタのようなものがない[11]ので，測定結果はI/Oピン経由でLCD（SC1602）に出力した。結線は，RSを 24 番ピン，Eを 26 番ピン，BD4 か

9 差動信号を使ったノイズに強い通信規格で，車載機器，産業用ロボットなどに使用されている。

10 点滅が早いので点灯し続けているように見える。

11 もちろん TeraTerm などを使い USB シリアル経由でモニタリングすることは可能である。

らBD7 は，27 から 30 番ピンへと接続している。LCD の制御は、Cookbook にある TextLCD ライブラリを Importして利用した。

```
#include "mbed.h"
#include "TextLCD.h"

TextLCD lcd(p24, p26, p27, p28, p29, p30);
DigitalOut led1(LED1);
Timer t;

int main() {
    int i;

    t.start();
    for(i=0;i<100000;i++){
        led1.write(1);
        led1.write(0);
        //省略してあるが 1、0 を
        //交互に 49 回（計 50 回）ずつ繰り返し
    }
    t.stop();
    lcd.printf("%d",t.read_us());

    return 0;
}
```

Program.2 main.cpp（mbed）

表示された値は，533334 であった。こちらの単位はマイクロ秒なので 0.533 秒ということになるが，これは Arduino の 70 倍以上の速さである。

メモリの容量も含めて考えれば，大量のデータを高速に処理しなければならないような用途，たとえばマルチメディアの処理などについては mbed が有利だろう。

３． 超小型コンピュータ Raspberry Pi

3.1. Raspberry Piとは

ケンブリッジ大学と米Broadcom社が支援するラズベリーパイ財団 [12]が、「プログラミングに親しめる環境を子供たちへ！」と開発したのがRaspberry Piである。

Raspberry Pi はクレジットカードサイズのボードで RAM 256MB，USB ポート×1 の model A と，RAM 512MB，USB ポート×2，さらに Ethernet ポートまで備えた model B の二種がある。ドル建て価格はそれぞれ 25 ドルと 35 ドルであり，国内では RS コンポーネンツが model B を 2950 円で販売している。そのスペックは次表のとおりである。

CPU	BCM2835（SoC） Core ARM11 700MHz（875DMIPS）
RAM	512MB
SDIO	SD/SDHC ×1
HDMI	max 1920×1200
AUDIO	3.5 ㎜ Stereo
USB	A Type×2
LAN	10Base-T/100Base-TX ×1
GPIO	×17（SPI，I2C，UART，PWM）

Table.3 Raspberry Pi Type B

表からわかるように，標準的な PC のインタフェースはすべて装備されており，マイコンボードというよりはオールインワンのマザーボードである。実際, Raspberry Pi の特徴は， SD カードにインストールした Linux が稼働することで, キーボードとマウス, そしてモニタ（＝テレビ）を接続すれば，立派にパソコンとして使うことができるのだ。動作は多少重いが X Window も動くので，これ一台ですべてをまかなうことも不可能ではない。それでいて，外部回路を接続すれば C/C++言語から Arduino 的なハードウェア制御もできるのである。しかもサイズは 85.6×56×21 ㎜，重量は 45g にすぎない。まさに手のひらサイズの万能ボードと言えるだろう。

Raspberry Piは発売当初から大人気で，世界中に大量のバックオーダーを抱えていた [13]のだが，徐々に需給も改善されてきたようだ。Linuxは敷居が高いと感じる人もいるかも知れないが、元来が子供用に作られたボードなので扱いは難しくない。OSはSDカードに導入するのだが，調子が悪くなったらすべてを消して再インストールしても，せいぜい 10 分程度で済む。興味のある人はぜひ入手して試してみてほしい。

3.2. ペリフェラル

Raspberry Pi を購入しても，箱に入っているのは一枚のボードだけで, 電源などの付属品は一切付いていない。動作させるためには，本体以外に以下のものを用意する必要がある。

電源アダプタ

電源はMicro BのUSBコネクタから供給する。出力 3.5W（5V700mA）以上のアダプタが必要だが、コネクタも含めてスマートフォン用のACアダプタを流用 [14]できる。手持ちがなくて新規に購入しても 1000 円以下で

[12] Raspberry Pi を非営利組織で販売する目的で同名の財団を設立したようである。

[13] 筆者も手に入れたのは 12 月のことである。

[14] 筆者所有のものは 5V 1.2A（1200mA）だった。

入手できるはずだ。なおPCのUSBから給電する場合は，USB 2.0 経由では不足[15]で 3.0 が必要になる。

SD（SDHC）カード

クラス 4 で 4GB 以上（最大は一応 32GB）のもの。4GB の SDHC（500 円以下で入手可能）で充分だろう。

SD カードリーダ/ライタ

ノート PC には装備されていることも多い。100 円ショップでも入手可能。

ディスプレイ

HDMI 入力のディスプレイと接続ケーブル。HDMI 入力のあるテレビがあればよい。画質は悪くなるが，コンポジットビデオでの接続も可能。

LAN ケーブル

ネットワーク接続をするなら必要。

USB キーボードとマウス

PC 用のもの。GUI を使わないならマウスは不要。

PC

OS を SD カードへ書き込むときに使用。

ディスプレイ（＝デジタルテレビ）さえあれば，残りのすべてを新規に購入したとしても，たいした金額にはならないはずだ。

3.3. インストールと設定

Linux のインストールでは苦労することも少なくないが，Raspberry Pi の場合は、イメージをダウンロードして SD カードに書き込むだけだ。公式ホームページには複数のディストリビューションが用意されているが，Raspbian と呼ばれる Debian ベースのものが一般的なようだ。ここでは，Windows PC を用いた Raspbian の書き込み手順を説明しておこう。

1. http://www.raspberrypi.org/downloads から最新の Raspbian（執筆時点では 2012-12-16-wheezy-raspbian.zip）をダウンロードし解凍する。（フォルダ内に OS のイメージ 2012-12-16-wheezy-raspbian.img ができる）
2. https://launchpad.net/win32-image-writer/+download から win32diskimager-binary.zip をダウンロードし解凍する。（フォルダ内に実行可能ファイル win32diskimager.exe ができる）
3. SD カードリーダ/ライタに SD カードをセットし PC に接続する。
4. win32diskimager を起動し右上の Device の部分が SD カードのドライブを示していることを確認。（ドライブ指定を間違うと PC のハードディスクの内容を破壊してしまうので注意）
5. image File に 2012-12-16-wheezy-raspbian.img を指定し Write。
6. 数分程度で書き込み終了。

筆者自身は試してはいないのだが Mac の場合はもっと簡単なようで，https://hotfile.com/dl/185876039/25057df/RPi-sd_card_builder_v1.1.zip.html にある RPi-sd card builder というツールを使えば，イメージのダウンロードも含めて自動的に処理してくれるらしい。なお OS の SD カードへの書き込みに関する詳しい情報は http://elinux.org/RPi_Easy_SD_Card_Setup にあるので必要に応じて参照してもらいたい。

書き込みが終わったら，SDカードを本体裏面のカードスロットに挿入する。USBにキーボードとマウス，HDMIにテレビを接続してから電源を入れれば[16]，即座に画面上をブートメッセージが流れ、数秒後には自動的にraspi-configツールが起動して，コンフィグレーション画面が表示されるはず[17]だ。ここまで到達すれば、ハードウェア本体とOSの書き込みには問題がなかったことになる。

コンフィグレーション画面が立ち上がったら，以下の 4 項目を設定しておく。

expand_rootfs

選択して Enter キーを押せばよい。しておかないと SD カードのスペースをフルに利用できない。

config_keyboard

日本語キーボードなら，Generic 105-Key（Intl） PC…Other…Japanese の順に進み，Japanese-Japanese (OADG 109A)を選択する。残りの項目はデフォルトのままで構わない。

change-locale

en_GB.UTF-8，ja_JP.EUC-JP，ja_JP.UTF-8 UTF-8 の３つを選択状態にしておく。（スペースキーを押して＊マークを付ける）日本語フォントを入れるまではメッセージが文字化けするので後回しでもよい。日本語を使うつもりがなければデフォルトのままで構わない。

ssh

Enable を選択する。これで ssh 経由でのアクセスが可能になる。ネットワークを使わないなら不要。

なお，ディスプレイの表示で画面の端が切れるときは，overscan の項目を変更すると解消できる。また，overclock では最大 1GHz までのクロックアップ設定が可能である。もちろん動作保証外ではあるが，試した範

[15] 2.0 の容量は 500mA しかない。一応立ち上がるが、LAN を使ったりするとダウンするらしい。

[16] Raspberry Pi には電源スイッチがないので、AC アダプタをコンセントに差し込むだけだ。

[17] 画面に何も表示されないときは，テレビ側の入力選択が HDMI になっていることを確認する。

囲内では問題なく作動し，X Window の操作では効果が実感できた。

以上の設定が終わったら，raspi-config ツールのメイン画面で Finish を選択すると，reboot now?と問われるので，Yes を選んで再起動する。SD カードの領域拡張でしばらく待たされるが，いずれ

```
raspberry pi login:
```

とログイン待ちになるので，デフォルトのユーザ名 pi を入力し，

```
password:
```

に続けてデフォルトのパスワード raspberry を入力すれば pi@raspberrypi~S というコマンドプロンプトが表示される。なお再びコンフィグレーション画面を呼び出したいときは，プロンプトに続けて

```
sudo raspi-config
```

と入力[18]すればよい。

OSが起動したら，日本語環境のためのソフトをいくつかインストールするが，それらのダウンロードのためにインターネットへ接続しておく。ルータ側でDHCPが動いていれば[19]，ルータとRaspberry Pi本体をLANケーブルでつなぐだけでよい。接続されたことを確認するには，以下のようにブラウザでも起動してみればいいだろう。まず，プロンプトに続けて

```
startx
```

と入力し X Window（LXDE）を起動する。GUI が表示されたら，左側の Midori と表示されているアイコンをクリックするとブラウザが起動する。まだフォントを入れていないので日本語表示はできないが，英語サイトを見ることは可能なはずだ。X Window では Windows や Mac と同じような操作ができるので，いろいろと試してみるとよいだろう。GUI から抜けるには，左下のメニューから Logout を選べばよい。

インターネットへの接続ができたら，日本語環境の構築を行う。まず，プロンプトに続けて

```
sudo apt-get install ttf-kochi-gothic xfonts-intl-japanese xfonts-intl-japanese-big xfonts-kaname
```

と入力し，フォントをインストールする。これで X Window を立ち上げれば日本語表示はできるはずだ。続けて，コマンドラインの日本語化と日本語入力のために

```
sudo apt-get install jfbterm uim uim-anthy
```

として日本語表示システム jfbterm, 入力メソッド uim, かな漢字変換システム anthy をインストールする。プロンプトから

```
jfbterm
uim-fep
```

として起動すれば，コマンドライン上のメッセージは日本語化される。FEPのON,OFFキーを設定[20]すれば，コマンドラインでの日本語入力も可能になるらしいが，その必要はないだろう。何も設定しなくても，X Window 上では半角/全角キーやShift+SpaceでFEPを起動すれば，LXTerminalを含めて日本語入力が可能である。

以上で OS のインストールと設定は終了だが，Linux なので最後はきちんとシャットダウンしなければならない。プロンプトから

```
sudo shutdown -h now
```

としてから電源コンセントを抜く。

3.4. Raspberry PiでのI/O

Raspberry Pi の右上には P1 と呼ばれる 2×13 列のピンがある。http://elinux.org/Rpi_Low-level_peripherals によるとピン配置は次の表のようになっており，17 本は GPIO（General Purpose I/O）として使用できるようだ。

3.3V	5V
GPIO 0（SDA）	5V
GPIO 1（SCL）	GND
GPIO 4（GPCLK0）	GPIO 14（TXD）
GND	GPIO 15（RXD）
GPIO 17	GPIO 18（PCM__CLK）
GPIO 21（PCM_DOUT）	GND
GPIO 22	GPIO 23
3.3V	GPIO 24
GPIO 10（MOSI）	GND
GPIO 9（MISO）	GPIO 25
GPIO 11（SCKL）	GPIO 8（CE0）
GND	GPIO 7（CE1）

Table.4 Layout of P1

ここでは，Program.1，2 と同様に LED を点滅させるプログラムを，文献 7）の方法に従って作成した。回路的には GPIO 4（左列上から 4 番目のピン）から，LED および 330Ω の抵抗を直列に経由し，右列上から 3 番目の GND へと接続した。

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <time.h>

int main(){
  int fd;
  long int i;
  clock_t start,end;
```

18 スーパユーザ権限が必要なので sudo を付ける。

19 DHCP がわからないという人なら動いているはず。

20 /usr/share/uim/generic-key-custom.scm を変更するらしい。

```
  fd=open("/sys/class/gpio/export",O_WRONL
  Y);
  write(fd, "4", 2);
  close(fd);

  fd=open("/sys/class/gpio/gpio4/direction
  ",O_WRONLY);
  write(fd, "out", 4);
  close(fd);

  fd=open("/sys/class/gpio/gpio4/value",O_
  RDWR);

  start = clock();
  for(i=0;i<100000;i++){
    write(fd,"1",2);
    write(fd,"0",2);
  }
  end = clock();

  close(fd);

  printf("%f",((double)(end-start))/CLOCKS
  _PER_SEC);

  fd=open("/sys/class/gpio/unexport",O_WRO
  NLY);
  write(fd, "4", 2);
  close(fd);

  return 0;
}
```

Program.3 led.c (Raspberry Pi)

プログラムの作成から実行までの手順は以下のとおりである。プロンプトから

```
nano led.c
```

としてエディタ nano を起動し，プログラム(led.c は任意のプログラム名)を作成する。終わったら，ctrl＋O（ctrl キーと O を同時に押す）で保存し，ctrl+X で nano を終了してプロンプトへ戻る。

```
cc led.c -o led
```

としてコンパイルすると実行可能ファイル led が作成されるので

```
sudo ./led
```

として実行[21]すればよい。

プログラムの内容についても簡単に説明しておこう。Raspberry Pi でのポートはファイルと同じ扱いだ。

最初に，/sys/class/gpio/export を開いてポート番号（"4"は GPIO4 の場合）を書き込み，使用するポートを通知する。複数のポートを使うときは write 文を書き連ねて指定すればよい。

次に/sys/class/gpio/gpio4/direction を開き，出力ポートの場合は"out"を書き込む。これで GPIO4 は出力ポートになる。入力ポートにしたい場合は"in"と書き込む。

ポートからのデータ出力は/sys/class/gpio/gpio4/value に"1"を書き込めば HIGH（＝3.3V）となり，"0"を書き込めば LOW（＝0V）となる。

使い終わったら/sys/class/gpio/unexport を開いて使用終了を通知する。

このプログラムの実行結果として表示された結果は 25.32[22]であったので処理時間は 25.32 秒ということになり，Unoの 39.7 秒はともかくmbedの 0.533 秒には遠く及ばない。直接ポートを操作する方法[7]を使えばスピードアップできるだろうが，それでもmbedにはかなわないだろう。理由はOSが相当のマシンパワーを消費していることに違いない。

Raspberry Pi の絶対的な速度は mbed に劣るし，低レベル I/O の使い勝手では Arduino に劣っている。しかし，OS 上で動く C/C++のプログラムから I/O 操作ができることに大きな価値がある。Linux の資産を利用しながら，他のマイコンボードと連携させれば，マイコンの世界に新たな可能性が開かれるに違いない。かっての PC がそうであったように，個人が自由に遊べるコンピュータが戻ってきそうな,そんな楽しみのあるボードなのである。

４． ARM ベースのアルディーノ Due

長い間，Arduino のマイクロコントローラは Atmel の AVR だったが，ついに ARM CPU を搭載した Due と呼ばれるボードが登場した。現時点（2012 年 12 月）では Due の流通量は極端に少なく，実は筆者もまだ入手できていないのだが，現在，最も注目されているマイコンボードなので www.arduino.cc にある情報を基に概要だけを紹介しておこう。

Dueは 4×2.1 インチ(101.6×53.3 ミリ)のサイズで、外観はArduino Mega 2560 とほとんど変わらない。目立った違いは，USBがマイクロBタイプになり，しかも 2 系統[23]搭載されていることくらいだろう。ピン配置は，デジタルピン 0～13 番，アナログピン 0～5 番，ICSPに

[21] ポートアクセスにはスーパーユーザ権限が必要なようだ。

[22] OS が動いているので，測定するたびに結果は違ったが，おおよそ 25～26 秒の範囲だった。

[23] 一系統はプログラミング用，もう一系統はホスト機能用として使い分ければ便利だろう。

ついてはUnoやDuemilanoveと同じで，従来のシールドとの互換性もあるようだ。但し，他のARM系のマイコンボードと同様に，I/Oが 3.3Vであることには注意しておかなければならない。

開発ツールは，Arduino1.5.1 がベータバージョンとして提供されており，既存の AVR 系のボードにも対応しているので，Uno を使って試してみた。外見的には、選択できるマイコンボードに ARM 系のメニューが加わっており Due に関する項目もあったが，それ以外の違いはなく，使用法などは従来のものと差がなかった。また付属しているサンプルプログラムも従来のままであり，ソフトウェア的な使い勝手はこれまでの Arduino とほとんど変わらないものと考えられる。

スペックを比較するため，mbed のものと並べて表にしたのが次の Table.5 である。

	Arduino Due	mbed LPC1768
CPU	SAM3X8E (ARM Cortex M3) 32bit RISC	NXP 1768 (ARM Cortex M3) 32bit RISC
Clock Speed	84MHz	96MHz
Program area	512KB	512KB
Data area	96KB	32KB
Digital I/O	×54	×26
Analog In	×12	×6
Analog Out	×2	×1
PWM Out	×12	×6
Serial	×4	×3
SPI	×1	×2
I2C	×2	×2
CAN	×1	×1
Ethernet	－	×1
USB	×2	×1
Voltage	3.3V	3.3V

Table.5 Arduino Due and mbed

CPU のベースは両者ともに Cortex M3 であるが，Due はクロックが 12.5%ダウンしているので，純粋な演算能力も同程度に低下しているはずだ。メモリ関係は，プログラム領域のサイズは同じだが，Due のデータ領域は mbed の 3 倍で int 型にして 2 万 4000 個分ある。I/O は内容的にはほぼ同じだが，本数では Due が圧倒している。唯一 Due には Ethernet がないが，これはシールドで簡単に追加できるはずだ。

これだけの内容で，スイッチサイエンスでの販売価格が 4980 円なのだから人気があるのも無理はない。速度やメモリの量，あるいはデジタル I/O の本数などで Uno に不満があるのなら，検討してみるべき Arduino である。

５． 終わりに

ラズベリー財団のページには，“to find a platform that, like those old home computers, could boot into a programming environment”と書かれている。一昔前のパソコンは，GUI などなかったが，ちょっとしたプログラムを書いたり，入出力をしたりするのは今よりも遥かに簡単だった。わずか数十行のプログラムを書くのに，Visual Studio や Eclipse の使い方を覚えたり，モータを一つ回すのに，PCI Express の I/O カードを引っ張り出したりする必要などなかった。今やそういう目的には，PC よりもマイコンボードが向いている。PC からマイコンというと，何だか先祖返りのようでもあるが，Intel から ARM へという時代の流れの中では，最先端でもあるのだ。

ところで，本号は，齋藤雄志先生と，佐藤創先生の退職記念号である。お二人は経営学部情報管理学科の時代から，学部の先頭に立って活躍されてきた。能力だけでなく，お人柄も心から尊敬できるお二人がご退職されるのは寂しいばかりだが，「集まり散じて人が変わる」のが大学なのだから，それも仕方のないことなのだろう。

お二人がそうであったかはわからないが，今の六～七十代には，少年時代に工作好きでラジオやアンプを作っていたなどという人が少なくない。およそ技術とは関係のない分野の人と話をしていても，「50 年前の秋葉原はねえ…」などという話題が出てきたりするのだ。最近の少年たちは忙しく，なかなかモノ作りをしたりする機会はないだろう。もちろん，プログラムやコンテンツ作りだってモノ作りには違いない。しかし，たまには形のあるモノを作ってみるのも悪くない。きっとマイコンはそのためのよい道具になるはずだ。

参考文献

[1]IC Insights, Extreme Results in Top 25 2011 Semiconductor Sales Ranking, http://www.icinsights.com/news/bulletins/Extreme-Results-In-Top-25-2011-Semiconductor-Sales-Ranking

[2] 湯之上，Intel とイノベーションのジレンマ，Electronic Journal 2012 No12(p42-45)

[3]IC Insights，Standard PCs Will Lose Status as Largest IC Application in 2013,http://www.icinsights.com/news/bulletins/Standard-PCs-Will-Lose-Status-As-Largest-IC-Application-In-2013

[4]NXP fan，mbed を始めましょう，http://mbed.org//users/nxpfan/notebook/lets_get_started_jp/

[5]ARM/Cortex マイコン徹底研究，CQ 出版社(2010)

[6]Raspberry Pi official page，http://www.raspberrypi.org/

[7]桑野, ARM コンピュータで I/O, インタフェース 2012 No.12（p73-82）