Information technology researching society
of the Gunmer, by the Gunmer, for the Gunmer

Vol. 01
Comic Market 87 版

群馬大学電子計算機研究会

Lollipop

IGGG
http://www.iggg.org/

群馬大学電子計算機研究会　著

群馬大学電子計算機研究会 IGGG "Lollipop" Vol. 01 Comic Market 87 版

目次

Conoha VPS に Gentoo を入れた

擬音 @gion_U

1　この記事について

『ConoHa VPS に Gentoo を入れたよ』情報をあまり見ないので, 部誌に使える適当なネタが浮かばなかったわたしはこのことについて書くことにした.

1.1　なぜ ConoHa VPS か

元々**さくらの VPS** を使っていたわたしが, PyCon 2014 に行ったときに ConoHa ブースでクーポンをもらったりしたのを機に ConoHa も使ってみようか, と思ったのがきっかけ.

1.2　なぜ Gentoo か

さくらの VPS で Gentoo を使っていたけれど, ある種酔狂で使ってるようなところがあったので, ここで CentOS あたりに乗りかえるのもアリか, と思いインストーラを立ちあげたはいいが, ディスク構成を決めるところで BtrFS を使いたいように使えなさそうだった（subvolume を切って `/` とか `/home` にマウントしたかった）し, RH 系列に疎く `debootstrap` 的なプログラムがあることを知らなかったので

そういうことになった.

1.3　なぜ BtrFS か

subvolume 切ってその単位でスナップショット取れたり透過圧縮できたり, イカしてるじゃん, という程度の理由で選んでるのでそのうち他のひとみたいに FS 破損して涙目になるんだと思う.

2　#gentooinstallbattle

といっても別段特別なことはしなくてもいいはず. Gentoo をインストールしたことがあるなら多分どうにかなる.

2.1　下準備

SFTP で ISO イメージをアップロード. Gentoo livecd でもいいんだけれど, わたしは SystemRescueCD のイメージが手元にあったのでそれを使った（SRCD は Gentoo ベース）.

ConoHa VPS は割り当てられてるディスク領域を 2 つに分割できるらしいけど, 面倒臭いので 100GB 1 本

にした．そもそもパーティションのことを考えたくない（あとで拡縮する必要が出てくるかもしれない, と考えはじめて夜も眠れなくなるタイプの人間）から BtrFS を使ってるというのもあるし．

次に ssh でログインできるようにした．ConoHa VPS のコンソールにはペースト機能もついてたりして便利な感じはあるけれど, やっぱり慣れてる端末から作業したかった．

```
# mkdir -m 700 ~/.ssh; cd ~/.ssh
# wget https://github.com/gion-xy.keys -O authorized_keys
# chmod 600 authorized_keys
```

これで外から root@[conoha-vps-ipaddr] に ssh 接続できるはず．

2.2 ディスクの整備

まずは単一パーティションを BtrFS でフォーマット．コマンド叩いてもいいし startx して GParted を立ちあげてもいい．

次に subvolume を切ってマウント．

```
# mkdir /btrfs
# mount /dev/vda1 /btrfs; cd /btrfs
# btrfs subv cre @
# btrfs subv cre home
# btrfs subv cre portage
# mount -o compress=lzo,subvol=@ /dev/vda1  /mnt/gentoo
% mkdir -p /mnt/gentoo/{home, var/portage}
% mount -o compress=lzo,subvol=home /dev/vda1 /mnt/gentoo/home
% mount -o compress=lzo,subvol=portage /dev/vda1 /mnt/gentoo/var/portage
```

2.3 stage3 tarball の展開

念のために日時を合わせてから stage3 tarball を展開．

```
# date
# cd /mnt/gentoo
# wget ${stage3_tarball_uri} -O - | tar xvjp
```

最後の行は wget で取得してきたものを直接 tar の標準入力に渡してアーカイブを残さないようにしている．

2.4 ベースシステムのインストール

ミラーリストから日本のサーバーを使うように指定．

```
# mirrorlist -i -o >> etc/portage/make.conf
# mirrorlist -i -r -o >> etc/portage/make.conf
```

gcc の-march=native の展開結果をもとに make.conf の CHOST 変数を編集．

```
# gcc -march=native -E -v - </dev/null 2>&1 | sed -n 's/.* -v - //p'
```

の結果を make.conf の CHOST=""の中にコピペすればいい．

新環境へ chroot して, portage tree を取得・更新．

```
# cp -L /etc/resolv.conf mnt/gentoo/etc/
# mount -t proc none /mnt/gentoo/proc
# mount --rbind /sys /mnt/gentoo/sys
# mount --rbind /dev /mnt/gentoo/dev
# chroot /mnt/gentoo /bin/bash
# source /etc/profile
# emerge-webrsync
# emerge --sync --quiet
# eselect news read
```

プロファイルの設定. たぶん desktop プロファイルは使わないと思うのでそのままでいいと思う……

```
# eselect profile list
# eselect profile set 1
```

タイムゾーンを設定しておく. 終わったら念のため時間を確認.

```
# rm /etc/localtime
# ln -s /usr/share/zoneinfo/Asia/Tokyo /etc/localtime
# echo "Asia/Tokyo" > /etc/timezone
# date
```

2.5 たのしいカーネルビルド

とりあえずカーネルソースを持ってくる.

```
# emerge gentoo-sources
# cd /usr/src/linux
```

カーネルコンフィグで失敗すると起動しなくて悲しい思いをしたりするので, `localyesconfig` ターゲットと `lspci` コマンドに助けてもらう.

`make localyesconfig` を実行すると, 現在のシステムで読みこまれてるモジュールがカーネル組み込みに設定される. また,

```
# lspci -k | grep Kernel
```

とすると, デバイスが利用しているモジュールの名前が出力されるので, これらを `make nconfig` の [F8](SymSearch) で検索して有効にしたりすればいい. 今回は BtrFS root の構成なので `CONFIG_BTRFS_FS` を組み込みにした.

ここで突然の読者プレゼント

起動するカーネルのコンフィグファイルを用意しました: `http://pastebin.com/QtmNa85a`
カーネルを LZ4 圧縮するように設定してるので, `app-arch/lz4` を `emerge` しないといけないと思う.

コンフィグを終わらせたらビルドする.

```
# make && make modules_install && make install
```

`make` に`-jn` とか`-ln` とかつけて並列ビルドさせてもいいと思う. わたしは`-j4` を指定した.

2.6 システム設定

`/etc/fstab` ファイルの編集. わたしの環境ではこんなふうにしました:

```
/dev/vda1               /               btrfs           noatime,compress=lzo,subvol=@           0 1
/dev/vda1               /home           btrfs           noatime,compress=lzo,subvol=home        0 1
/dev/vda1               /var/portage    btrfs           noatime,compress=lzo,subvol=portage     0 1
```

BtrFS では dump/pass は 0/1 にする.

`/etc/conf.d/hostname` ファイルの編集.

```
% nano -w /etc/conf.d/hostname
hostname="myhost"
```

ネットワークの設定. まず `net-misc/netifrc` を `emerge` してから, `/etc/conf.d/net` ファイルを編集する. VPS に割り当てられた IP アドレスを `x.x.x.x` だとして,

```
config_eth0="x.x.x.x/24 brd x.x.x.255"
routes_eth0="default via x.x.x.1"
```

として変更を保存したあと, サービスとして登録すれば起動時にネットワークが設定されるはず.

```
# cd /etc/init.d
# ln -s net.lo net.eth0
# rc-update add net.eth0 default
```

ロケールの設定. /etc/locale.gen を開き, 必要なロケールについて先頭の#を外して保存してから

```
# locale-gen
```

GRUB のインストールと/dev/vda への書き込み.

```
# emerge grub
# grep -v rootfs /proc/mounts > /etc/mtab
# grub2-install /dev/vda
# grub2-mkconfig -o /boot/grub/grub.cfg
```

root のパスワード設定とユーザーの作成. このとき sudo もインストールしてしまい, 普段使いのユーザーに sudo の実行権限をつける.

```
# emerge sudo
# passwd
# useradd -m -G users,wheel [username]
# passwd [username]
# su [username]
```

su/sudo でユーザーを切り替えてから, 2.1 節でした公開鍵の設置をもう一度行い, いま作成した一般ユーザーに ssh ログインできるようにする. ただ, そのままログインしようとするとホスト鍵認証で失敗するので, インストール作業が終わってからクライアント側で

```
% ssh-keygen -R [vps_ipaddr]
```

などとして, ホスト鍵を一旦削除しないといけない.

ここまでできたら無事起動するのを祈りながら後始末と再起動.

```
# exit
# cd
# umount -l /mnt/gentoo/dev{/shm,/pts,}
# umount -l /mnt/gentoo{/boot,/proc,}
# halt
```

ISO マウントを解除してから VPS を起動し, ConoHa VPS の VNC コンソールを開き, GRUB のメニューや Linux のブート処理ののちログインができれば成功. おめでとうございます.

2.7 これから

各自で使いたいものを入れましょう :-)

参考資料

- 新しいマシンを入手したので Gentoo をインストールしました - Emacs ひきこもり生活
  http://d.hatena.ne.jp/meech/20120301/1330626616
- Installing Gentoo Linux (amd64) in Sakura VPS
  https://gist.github.com/rkmathi/7762190

HOG 特徴を用いた乳首修正パッチシステム

群馬大学電子計算機研究会 IGGG

ism67ch

1　はじめに

まず，本研究の背景について述べる。2010 年 12 月 15 日に東京都青少年の健全な育成に関する条例の改正が可決され採択された。この一連の内容において，当時都知事の二次元創作に対する発言に対し，様々な非難があった。

私個人の意見としては，性的表現も表現の自由ではないかと考えるが，趣向がない人にとっては不快に感じることもあるのだろうとも思う。しかし，純粋にその内容を楽しんで欲しいのである。もし，不快に感じる部分がその性的描画にあるのであれば，そこを隠せば内容に集中してもらえる人も居るのではないかと考えた。よって，本研究では性的描画の一例として乳首を隠すことによって，それを実現する方法を提案する。

具体的には，Histograms of Oriented Gradients 特徴を用い，乳首の特徴ベクトルを算出し，それを用いてテスト画像に置いて乳首を検出，パッチを当てるといった流れである。尚，今回は様々な理由から二次元画像に対してのみ行うこととする。

2　HOG 特徴について

HOG とは Histograms of Oriented Gradients の略であり，N.Dalal '05 で発表された特徴量である。局所領域（セル）の輝度の勾配方向をヒストグラム化して得られる特徴量であり，特徴としては以下が上げられる。

- 幾何学的変換に強い
- 照明の変動に強い

以上のような特徴を活かし，HOG 特徴は人検出や車検出に利用されている。 算出の手順は以下の 3 ステップである。

1. 輝度の勾配強度と勾配方向の算出
2. ヒストグラムの作成
3. ブロック領域での正規化

この 3 ステップに基づいて HOG の原理を説明していく

2.1　輝度の勾配強度と勾配方向の算出

まず，ある入力画像におけるセルについて定義する。セルとは，あるピクセルに置いて，そのピクセルを中心とした正方ピクセルの集まりである。例えば図 1 の例であれば，ある画像に対して，5×5 ピクセルのセルに分割をしている。

次にある 1 つのセルに対して考える。あるセルの中心ピクセルを (x, y) とする。このとき，このセルの 4 近傍のピクセルは図 2 のように，$(x, y-1)$，$(x-1, y)$，$(x, y+1)$，$(x+1, y)$ と表されることができ，各座標に置ける輝度値を関数 L を用いて表現する（(x, y) 座標の輝度値は $L(x, y)$ と示す)。中心ピクセル (x, y) における x 軸方向，y 軸方向の輝度勾配は以下のようにして計算することが出来る。

$$f_x(x, y) = L(x+1, y) - L(x-1, y) \tag{1}$$

$$f_y(x, y) = L(x, y+1) - L(x, y-1) \tag{2}$$

(1)，(2) 式を用い，勾配強度（magnitude）$m(x, y)$ と勾配方向（direction）$\theta(x, y)$ を以下のように定義する。

$$m(x, y) = \sqrt{f_x(x, y)^2 + f_y(x, y)^2} \tag{3}$$

$$\theta(x, y) = \tan^{-1} \frac{f_x(x, y)}{f_y(x, y)} \tag{4}$$

以上の動作を繰り返すことで，各ピクセルにおける輝度の勾配強度と勾配方向を求めることができる。

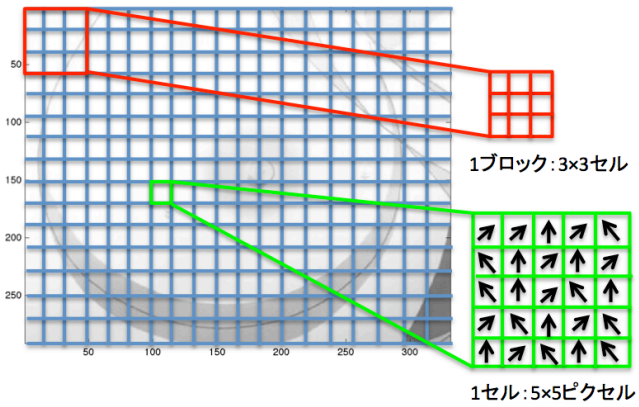


Figure 1: セルとブロック

2.2　ヒストグラムの作成

ここまでで，各ピクセルにおける勾配強度と勾配方向が求まったはずである。次のステップでは，先ほど定めたセル毎に，勾配強度と勾配方向におけるヒストグラムを作成する。このとき，勾配方向は 0° から 180° の角度を 20° ずつ 9 方向に分割してヒストグラムを作成する。

2.3　ブロック領域での正規化

最後に，各セルにおいてヒストグラムが作成する。次のステップでは，作成したセルをブロックと呼ばれる領域単位

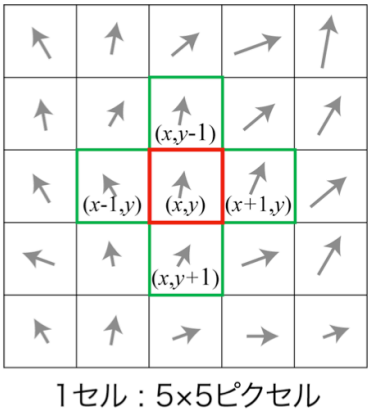


Figure 2: セル座標

で正規化する。ある n 番目の HOG 特徴量についての正規化について考える。ブロック単位で正規化していくのだが，様々な手法があるためここでの言及は避けるが，今回の実験に際しては，VL-Feat の関数を用いているため，正規化法もそれに準じている。詳しくはそちらを参照されたい。

3 内積計算を行って求めたスコアによるマッチング

内積スコアによるマッチングについて解説していく。先までの内容で，輝度画像における HOG 特徴を求めることが出来る。この HOG 特徴を用い，複数の訓練画像から，あるテスト画像において対応するパーツが求められるかどうかを実験する。 実験の簡略な手順は以下である。

1. 訓練画像それぞれにおける乳首の HOG 特徴を算出する
2. 求めた HOG 特徴の平均を求める（以降これを特徴ベクトルと呼ぶ）
3. テスト画像に対し，訓練画像と同じピクセルの検査窓を走査する
4. それぞれの検索窓に対し，HOG 特徴を求めておく
5. 4. で求めたそれぞれの HOG 特徴に対し，2. で求めた特徴ベクトルとの内積を求める
6. スコアをマップ化して，値が高いところにパッチを当てる

尚，今回 HOG を求めるための関数には VL-Feat の関数 vlhog を用いている。

4 実験

【実験条件】
訓練画像：ToLove るから図 4 の画像内における乳首
テスト画像：乳首露出している 10 枚の画像
画像サイズ：250 × 250 ピクセル
検出対象：乳首（42 × 42 ピクセル）
検索窓の移動：3 ピクセルずつ移動
HOG の設定：セルサイズ 15×10
9 方向に分割
ブロックサイズ 3×3
実装環境：Matlab，VL-feat
性能評価：隠れているか否かで Accuracy を求める

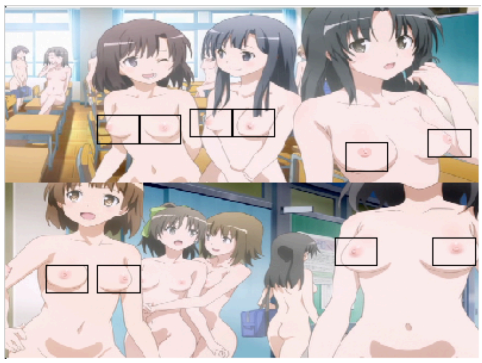


Figure 3: ToLove るの教室シーンにおける画像

今回利用した画像の採用理由としては，Google 検索で割と上位にヒットしたためであり，恣意的に乳首を選択した訳ではない。また実際に HOG 特徴を求める際には，図 4 の画像中の短形部分において，それぞれ HOG 特徴を算出している。セルサイズ，ブロックサイズにおいては，予備実験において最も精度が良かった数値を採用している。

5 実験結果

今回の実験結果としては，正答率 90 パーセントをマークした。入力結果に対する，出力結果は図 4 のような感じである。

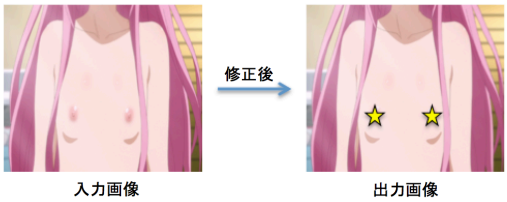


Figure 4: 入力と出力一例

図 4 から，乳首に修正が入っていることが分かる。

6 終わりに

本論文では，性的描画を規制する方法の一例を取り上げた。今後の課題としては，動画においてリアルタイムにパッチを当てられるようにするなどが上げられる (他にもいくらでもある)。 事後談であるが，テスト画像を男性キャラクターにした場合も修正はされた。しかしながら，当てたパッチのせいか，逆にエロくなってしまった。該当部位にどんなパッチを当てていくかも考慮の余地があるのかもしれない。 最後に，本論文は決して原作を否定するものではなく，むしろ多くの人に原作を知ってもらいたいというモチベーションであることを強調しておく。

C 言語ポインタを理解するための第一歩

1.　初めに

私は、学校で初めて習ったプログラミング言語が C 言語であった。講義で入門書を一通り読み、入門書レベルのプログラムを書くことができるようにはなりました。しかし、行列の足し算、引き算、掛け算の処理を行う関数を作成する際、2 次元配列の変数を関数に渡して、処理を行わせていたが、これらの関数を完成させるのに 3 日ほど時間がかかりました。特に時間がかかったのは、デバッグ作業でした。main 関数で上記の関数群を呼ぶ際に渡す実引数や関数の仮引数にどんなものを入れればよいかわからず、総当たりでデバッグ作業をしていました。その後、プログラムはうまく動作しましたが、なぜこの方法でうまくいったのか私は理解できませんでした。そこで、「C 言語ポインタ完全制覇」の本を同級生の 4 人と一緒に輪読し、C 言語について深く勉強しました。

```
/* 関数Multiply(行列の掛け算を行う関数)の定義 */
void Multiply(Matrix *psa,Matrix *psb,Matrix *psc){
    int i,j,k;

    /* 掛け算が可能かどうか判定する処理 */
    if((psa->col)!=(psb->row)){
        printf("掛け算することができません.\n");
        exit(0);
    }

    for(i=0;i<(psa->row);i++){
        for(j=0;j<(psb->col);j++){
            for(k=0;k<(psa->col);k++){
            psc->mtrix[i][j]+=psa->mtrix[i][k]*psb->mtrix[k][j];
            }
        }
    }
}
```

図１：行列の掛け算処理を行う関数

2.　対象読者と目標

主に C 言語の入門書を一通り読んだ方ぐらいのレベルの内容です。記述する内容は、「C 言語ポインタ完全制覇」を読んで、簡単に配列とポインタについてのことをまとめました。ここの内容を読んでいただいて、C 言語のポインタってこんな感じなのかという程度を知っていただければ幸いです。

3.　ポインタと配列の関係

C 言語の配列というと、入門書を読んだ方は、次のような図 2 をイメージすると思います。入門書等では、ポインタと配列について説明するとき、次のような説明がよく使われます。

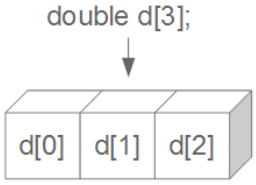


図２：配列のイメージ

「配列名は、配列の先頭要素へのアドレスを格納しているポインタと同じ働きを持つ。」－①

実は、この説明は正確ではありません。厳密なことを言えば間違っています。ここが C 言語のポインタにおいて重要な点だと私は考えています。解説をする前に知識として、「ポインタ変数 p に n 加算すると、そのポインタが指す型のサイズ×n」－② だけ進むことを覚えておいてください。

例えば、図３のようなコードがあったとします。

①より、次は成り立ちます。(⇔は同値を表す。)

p = &array[0];

⇔ p = array;

②の性質と上記の同値関係からℓ15～ℓ17 を次のように変更できます。

```
p = array;
for ( i = 0; i < 5; i++) {
      printf(“%d¥n”, *(p + i));
}
```

ところで、*(p + i)は p[i]と書くことができます。

```
p = array;
for ( i = 0; i < 5; i++) {
      printf(“%d¥n”, p[i]);
}
```

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      int array[5];
6      int *p;
7      int i;
8
9      /* 配列arrayに値を設定 */
10     for (i = 0; i < 5; i++) {
11         array[i] = i;
12     }
13
14     /* その内容を表示(ポインタ版) */
15     for (p = &array[0]; p != &array[5]; p++) {
16         printf("%d\n", *p);
17     }
18
19     return 0;
20 }
21 [EOF]
```

図３ ：array2.c

このことから、*(p + i)は p[i]は同じ意味になる。p = array;という代入が行われていますが、p の中身は一度も変わっていません。それなので、直接 array と書いてしまうと、次のようになります。

```
for ( i = 0; i < 5; i++) {
      printf(“%d¥n”, array[i]);
}
```

となり、p[i]は*(p + i)のシンタックスシュガー(簡便記法)であり、それ以外の意味は全くないのです。配列名に[ ]をつけても、配列名は「配列の先頭要素へのポインタ」と読み替えられています。ゆえに、②の意味ではなく、正しくは「[ ]がついてもつかなくても、**式中**では、配列はポインタに読みかえられる。」ということです。配列宣言時の[ ]は別の意味を持つので、その点を注意してください。

4.　ポインタ演算と[ ]のどちらがよいか

昔では、ポインタ演算を使用することで高速であるので推奨されていたらしいが、現在のコンパイラは最適化が進んでいるので、ポインタ演算を使用しなくても全く差はつかないそうです。

3 節の説明から、p[i]は*(p + i)のシンタックスシュガーです。どちらもコンパイル時には同様の処理が施されますが、人間側からみておそらく p[i]のほうが読みやすく感じと思われます。もともとシンタックスシュガーは人間にわかりやすくするために導入された機能なので、先人の方々も*(p + i)は読みにくいので、p[i]を生みだしたのだと思われます。

以上から、私個人としては大抵の場合、p[i]というコードを書いておいたほうが読みやす

く、この書き方を推奨します。

5. まとめ

今回は、本で学んだ配列とポインタについての簡単な説明でしたが、いかがでしょうか？非常に大まかな説明をしたので、納得された方、伝わったけどいまいちイメージができない方、まったく理解できないという方もいらっしゃると思います。もし、後者の方ならば、ごめんなさい、私がうまく伝えることができなかったからだと思います。次にこのような機会を与えていただけるならば、このことを反省して次の機会でよりよく伝えることができるように努力します。

ここまで読んでいただき、ありがとうございました。

By Sakuragi Yuuto.

6. 参考文献

[1] C 言語ポインタ完全制覇, 前橋和弥, 株式会社技術評論社,
2013 年 10 月 10 日(第 15 刷), p53～p65

[2] 図 2 , http://c-lang.sevendays-study.com/day5.html, 2014/12/01 に確認

Raspberry Piでドライブレコーダー作ってみた

Zakky (@ZakkyR)

このまえ OSC2014 Tokyo/Fall に参加して、Raspberry Pi の話があって興味が出てきたので買った。

適当にいじってたらドライブレコーダーができた。
◯静止画機能(1920×1080)
◯10 秒動画(320×240 FPS=60)
◯長時間動画(320×240 FPS=15)
っていう簡単なもの。
ホントはデジカメにしたかったけど、お金なくてモニターが買えなかったのと UI をつくる技術がなかったので時間的に断念。

まぁドライブレコーダーならモニターないのもあるよね！

ってことで名称はドライブレコーダー

使ったもの
◯Raspberry Pi B+(OS：Rasbian)
◯Raspberry Pi カメラモジュール
◯タクトスイッチ（青、黄、赤、白、黒）
◯ブレッドボード
◯ジャンパーワイヤー

操作はボタンだけで動かしたかったので、ラズパイの GPIO 端子を使う。

しかし残念なことに**電子工作ド初心者**なもんでつなぎかたがわからねぇ。

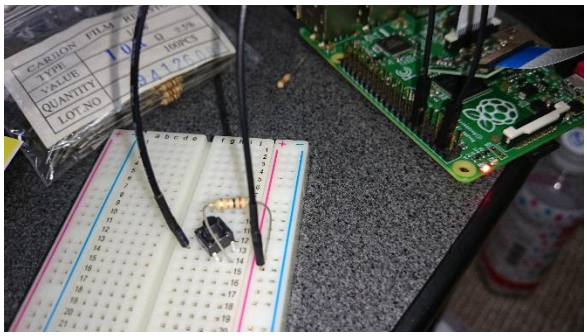
というわけでひたすらググる。

調べてたらこんなページにありついた

RaspberryPi #9 シャットダウンボタンをつくろう maigo@sekai
【http://maigo-lalala.blogspot.jp/2013/07/raspberrypi-9.html】

どうやら Raspberry Pi に wiringPi をインストールすると C 言語で制御ができるようになるらしい。

さっそくインストール。
カレントディレクトリに wiringPi ディレクトリがあるのでそこでビルド

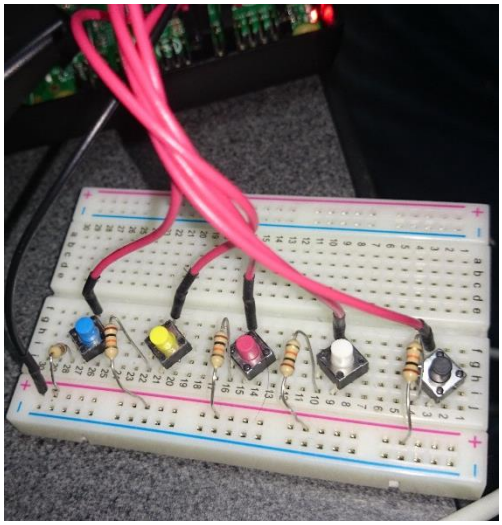


ビルドすると gpio コマンドが使えるのでネット上にあった写真を元につないでみた。
GPIO18 番 → タクトスイッチ → 抵抗(10KΩ) → 5V

テスト。
gpio –g read 18
ボタンを押している間は 1、押していないときは 0 と表示される。

ボタンのつなぎ方がわかったので今度は上記サイトのようにシャットダウンボタンを作ってみる。
C 言語で GPIO 制御を行うには wiringPi.h を include すればいい。
実行するときには管理者権限が必要なので、必ず *sudo* をつけること。

次にカメラについて。
カメラは Raspberry Pi カメラモジュールを使用。本体に専用のポートがあるのでさして、*raspi-config* でカメラの使用を Enable にすれば OK。
静止画撮影は *raspistill*、動画撮影は *raspivid* を実行する。それぞれ引数で撮影方式等色々指定できる。



左から静止画、10 秒動画、
長時間(2 時間)動画、撮影停止、シャットダウン
ニッパがないから抵抗の足が長い

うまくいったのでそれを元に静止画ボタン、10 秒動画ボタン、長時間動画ボタンを作成。
撮影時のファイル名は現在時刻になるようにした。

長時間用の動画停止ボタンを作成するが、ラズパイカメラの動画撮影コマンドには時間指定はあるが、中断するコマンドはないみたい。しかたがないので *ps* で撮影のプロセスの PID を調べ、

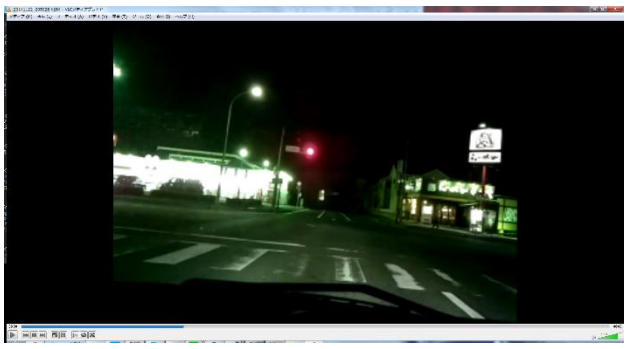
***kill* コマンドで強制終了する**という荒業で対応。

Raspberry Pi の電源をつけたら勝手にプログラムが実行できるようにする。
電源をつけると本来はユーザー名とパスワードを入力しなければならないが、ボタンだけで操作したいので自動ログインにする。
あとはプログラムを自動起動に設定すれば OK。
/etc/rc.local に実行ファイルの場所を記述しておけば電源つけたら勝手に起動する。

というわけで完成～♪

残念ながらケースが小さくてジャンパーワイヤーつなぐとふたが閉まらない。

今度レゴかなんかでケースつくるかなぁ



実際に撮影してみたところ明るい道なら夜でも撮れた。
動画は H264 ファイルで保存される。
解像度の関係で 10 秒動画よりも静止画のほうがサイズがデカかったりする。
長時間の方は 1 時間以上撮影してもデータサイズが 400MB もいかないすぐれもの。

現状の問題としては、Raspberry Pi は RTC モジュールがついていないため時刻の同期は NTP を使用して行われている。そのため、インターネットに接続されていないと同期されない。ファイル名をタイムスタンプでつけているので**撮影したファイルの日付が 3 日前とかになったりする。**(前回シャットダウンした時の時刻の続きになるそうだ)
今度時間があるときにキャラクタ LCD モジュールと RTC モジュールをつけたり、解像度とかの設定をもっと簡単にできるように改造しようとかなんとか。

まぁ今回は電子工作ド初心者が Raspberry Pi で遊んでみたらこうなったというところで。
次回があったらもう少し人に見せられるようなものをつくりたい。

iModela であそんでみる

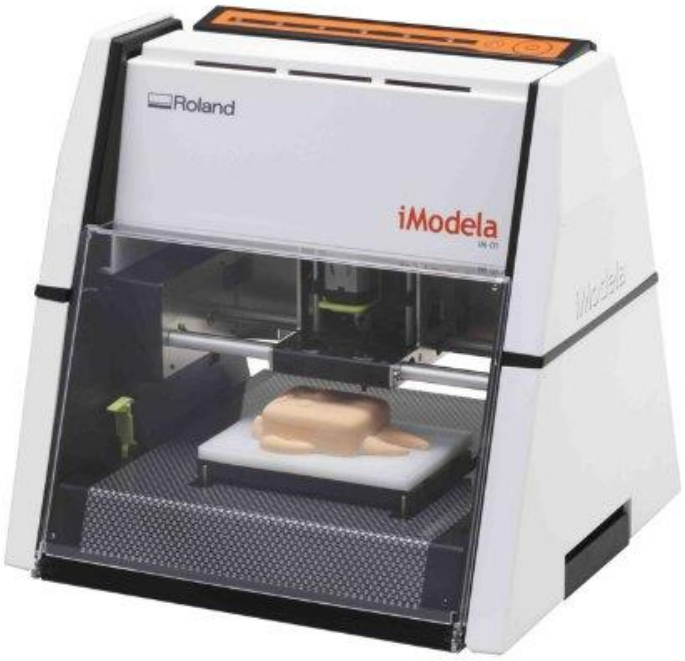
桐生川 (@kuriuzu)

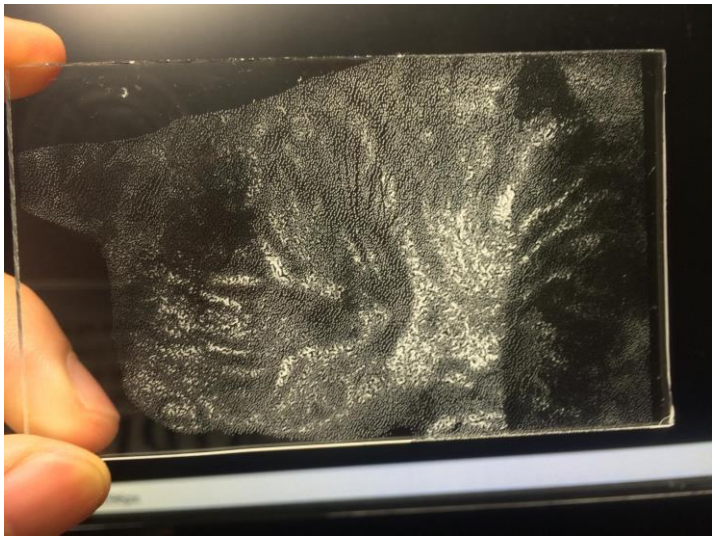
iModela って何

小型 3D 切削加工機。回転する刃物がコンピューター制御で動いて材料を切削加工してくれる。CNC フライス盤を小型化して個人でも入手できるようにしたみたいな感じ。

お値段 81000 円と切削加工機としては最安機種と思われる。

ただし、加工範囲が X,Y,Z:86x55x26mm とかなり小さい。





とりあえず使ってみる

アクリル板とか iPhone ケースをステージに取り付けて、彫刻とかをやってみた。

手順はこんな感じ。

1. Inkscape、Illustrator などのベクター画像編集ソフトで、素材画像をベクター化。
2. iModela 付属ソフト「iModela Creator」に読み込ませる。

    Inkscape の場合、Inkscape の編集画面上でオブジェクトを選択・右クリックして「コピー」、iModelaCreator の編集画面上で「Ctrl+V」で貼り付けるとイケる。
3. 「iModela Creator」で修正した後、iModela に出力。

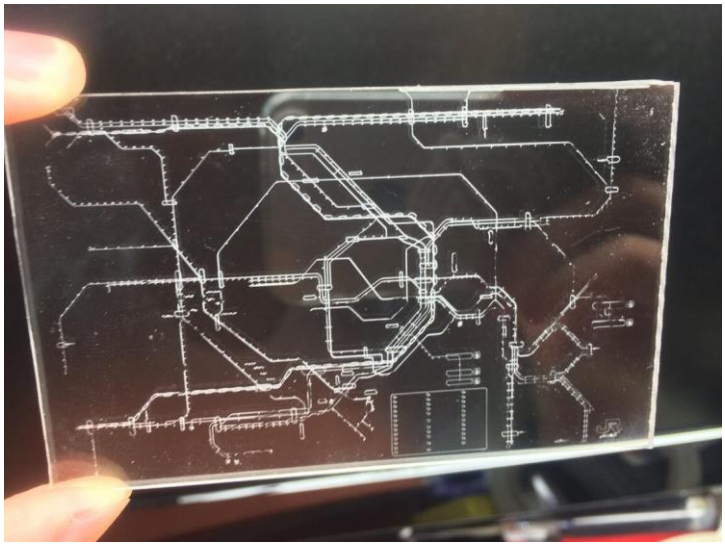
紙面では伝わりにくいが、どちらもきれいに切削できている。細かい模様も再現できている。

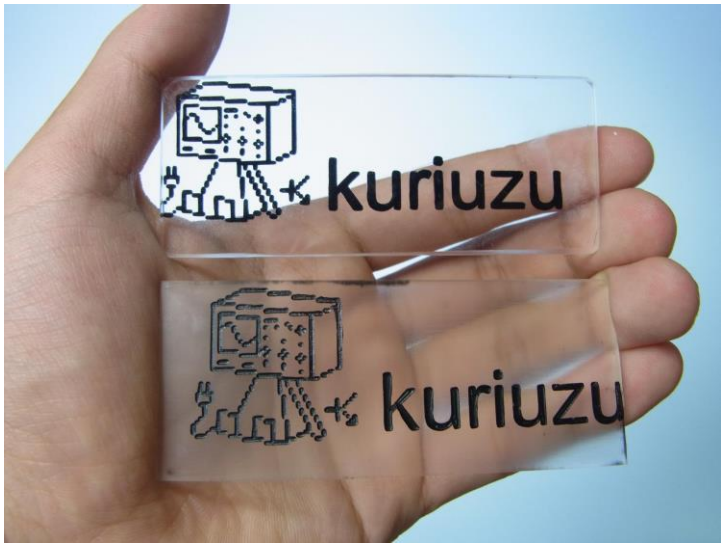
まとめ

工作が好きな人はハマりそう。工作の幅が広がる。

ただ、切削に時間がかかったり加工に手間がかかったりと使うのは面倒くさい。あと作動音がすごくうるさい。

まぁなかなか面白いから興味を持ったら Google 検索したりしてみよう。





有機電界効果トランジスタの特徴と構造

Y.S

Organic semiconductors have attracted much attention as promising materials for electronics devices such as organic field-effect transistor (OFET). The present paper describes recent advances, charactaristics and structure of OFET briefly.

Keywords: Organic Field-Effect Transistors 有機電界効果トランジスタ, Organic Semiconductors 有機半導体, Conductive polymers 導電性高分子, Electronics devices 電子デバイス, Thin film transistor 薄膜トランジスタ, Organic Electronics 有機エレクトロニクス

1 序論

通常、有機物は電気を通しづらい性質を持つ。しかし 1977 年に白川英樹らが有機物であるポリアセチレンにおいて電気が流れることを発見した [1]。これをきっかけに有機エレクトロニクスが発達し、これまでに有機電界効果トランジスタ (OFET)[2]、有機 EL(OLED)[3]、有機太陽電池 (OPV)[4] への応用に向けた研究が活発に行われている。有機物は比較的低温で膜形成することができ、低コストでデバイスが作製できることや、軽量であること、物質が柔らかいためフレキシブルなデバイスへの応用が期待できることがその理由としてあげられる。とくに OFET では電子ペーパー、フレキシブルディスプレイ、人工皮膚、無線自動識別 (RFID) タグといった、これまでにない製品への利用が検討されている [5][6]。次項からは、これまでのデバイスと OFET の違い、OFET の中枢となる有機半導体の特徴、OFET の駆動メカニズムについて簡単に説明する。

2 電界効果トランジスタ (FET)

電界効果トランジスタ (FET) というのはトランジスタの一種であり、その FET に有機半導体を用いているものが OFET である。トランジスタというのは 3 端子からなる素子であり、1 端子によって残りの 2 端子間の電流を制御できる仕組みになっており、スイッチングや増幅動作に用いられる。特に電気的にスイッチを ON、OFF することができることから、いくつもの素子を組み合わせて数値計算に応用され、現在では情報処理において必要不可欠なものとなっている。当然パソコンにも用いられており、今日の中央演算処理装置 (CPU) では数十億以上のトランジスタが集積化されている。

トランジスタの中でも FET はその構造上、比較的消費電力が小さく集積化しやすいという特徴を持つ重要な素子である。また FET の中でも、絶縁体を持つ MOSFET と持たないジャンクション FET(jFET) の 2 種類がある。現在主流となっている OFET は、MOSFET に構造が近いものとなっている。

3 有機半導体

有機物は炭素原子によってその骨格が形成されており、さまざまな構造のものが存在する。この特徴から任意の分子構造を設計することができ、溶媒に溶けやすい、耐熱性がある、電気が流れやすいといった、いろいろな性質を付与することが望める。また無機物に比べて低温で構造が変化するため成形加工が比較的容易であり、溶液状態にして広げる、蒸着するというような方法により、低コストで大面積の膜が作製可能である。それに加えて軽量であること、柔軟で曲げることが可能なものが多いことも有機物の利点である。

有機物の中で半導体の性質をもつものを有機半導体という。ある条件において電気を流すようになる物質が半導体である。一般的な有機物は絶縁体として働くが、キャリアとなる電子または正孔が分子内、分子間を容易

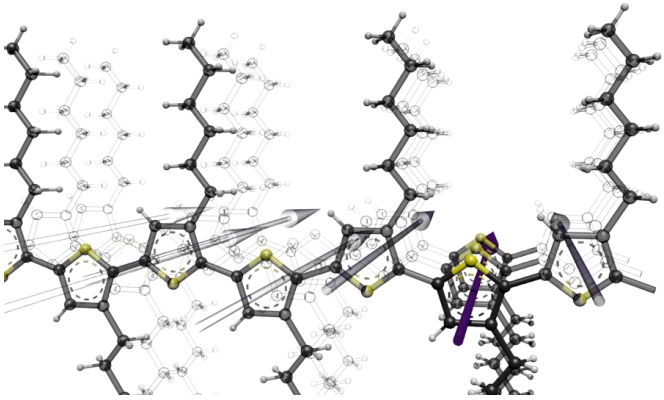


Fig. 1: Electric conduction in π-conjugated polymers. Purple arrows indicate electric current.

に移動できるような構造を持つ一部の有機物では電気を流す性質を持つ。その条件を満たす多くの分子ではπ共役系と呼ばれる構造をもち、ほとんどの場合、単結合と二重結合が交互になってできたリング状の構造を持っている。大抵このようなリング中ではキャリアが移動しやすく、また Fig.1 のように別の分子同士のリングが重なっているときにはリング間でのキャリアの移動が容易である。

有機半導体にはペンタセン (Fig.2(a)) 等の低分子から、ポリ-3-ヘキシルチオフェン (P3HT)(b) のような繰り返し単位の構造が 100 個以上連なった鎖状の高分子半導体も存在する。低分子の方が比較的綺麗な分子配列をとり、電気伝導の面では高分子よりも優れた性質になりやすい。一方で高分子は低分子に比べて柔軟性があり、曲げても平気なフレキシブルな製品をつくる上では重要となる。以上のような特徴を生かし、シリコンを中心とした無機半導体にはない性質を持つデバイスへの応用が期待される。

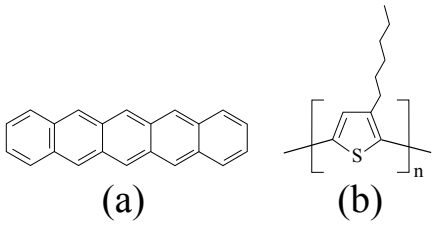


Fig. 2: Example of organic semiconductors. (a)pentacene(b)P3HT

4 OFET の仕組み

OFET は MOSFET と構造が非常によく似ている。MOSFET はソース電極とドレイン電極に加えて、ソース-ドレイン間に流れる電流を制御するゲート電極の 3 端子を持った素子である。ソース、ドレイン電極とゲート電極は絶縁膜によって隔てられており、ソース-ドレイン電極間に半導体が存在している。この半導体に有機半導体を用いたものが現在主流の OFET である。ソースとドレインは構造上同一であり、印加する電圧によってその呼び名が決まる。また層の順番によって構造が変化するが、ここでは絶縁膜の下にゲート電極があり、半導体層の下にソース-ドレイン電極が存在するボトムゲートボトムコンタクト型と呼ばれる構造の OFET を示す (Fig.3)。ここで簡単にデバイスの動作原理について触れる。通常、ただソース-ドレイン間に電圧をかけただけでは半導体層に電気は流れない。しかしプラス電荷とマイナス電荷がお互い引き合うように、ゲート電極に電圧をかけると絶縁膜を隔てた反対側の半導体中にキャリア（電子または正孔）が引き寄せられる。この状態でソース-ドレイン間に電圧をかけると、今度は集められたキャリアが流れソース-ドレイン間に電気が流れる。このようにして、ゲートに電圧をかけることによってソース-ドレイン間の電流を制御し、トランジスタとしての機能を果たしている。

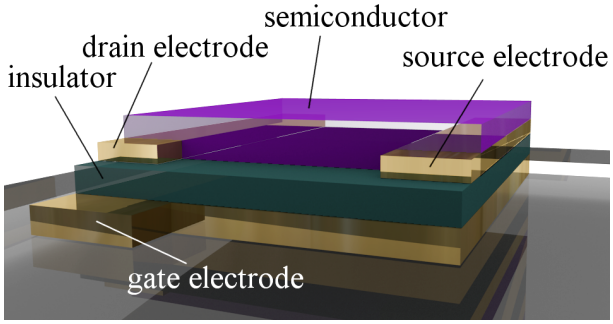


Fig. 3: Structure of bottom gate/bottom contact OFET.

5 OFET の現状

これまでに述べたように、OFET はかつてない機能を持ったデバイスへと応用できるとして期待されている。しかし実用化に向けての問題点がいくつか存在する。まずあげられるのはその導電性の低さである。有機半導体は無機半導体に比べて電気が流れにくく、これを改善するための分子設計やデバイス製作プロセスに関した研究が多数行われている。またデバイスの安定性も欠点となっている。有機物は比較的構造が変化しやすいため、空気中の分子と相互作用したり、電圧印加による影響を受けやすく、安定したデバイス動作が課題となる。しかし外部からの影響を受けやすいためセンサーに応用する動きもある。

しかし、半導体層だけでなく基板や絶縁膜等も有機高分子にすることによって折り曲げ可能なフレキシブルなデバイスとなること、軽量であること、低コスト生産が可能であることにより、次世代のエレクトロニクスを担うものとして有機エレクトロニクスが注目されている。

参考文献

[1] Chiang, C. K., C. R. Fincher, Jr., Y. W. Park, A. J. Heeger, H. Shirakawa, E. J. Louis, S. C. Gau, A. G. MacDiarmid. *Phys.Rev.Lett.* **39**,1098-1101 (1977).

[2] Wang, C., H. Dong, W. Hu, Y. Liu and D. Zhu. *Chem.Rev.* **112**,2208-2267 (2012).

[3] Reineke, S., F. Lindne, G. Schwartz, N. Seidler, K. Walzer, B. Lssem, K. Leo. *Nature* **459**,234-238 (2009).

[4] Graetzel, M. *Acc.Chem.Res* **42**,1788-1798 (2009).

[5] Martins, R., A. Nathan, R. Barros, L. Pereira, P. Barquinha, N. Correia, R. Costa, A. Ahnood, I. Ferreira, E. Fortunato. *Adv.Mater.* **23**,4491-4496 (2011).

[6] Usta, H., A. Facchetti, T. J. Marks. *Acc.Chem.Res* **44**,501-510 (2011).

1 個 300 円の Arduino で生活を便利にする話

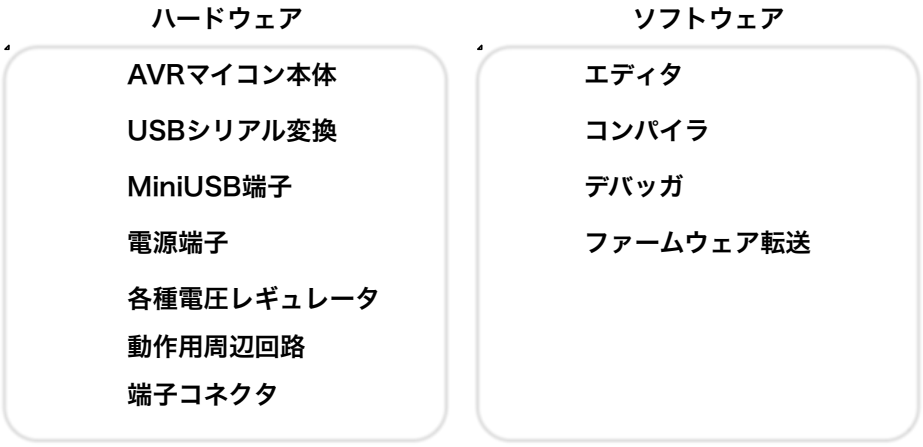
はじめに

この本を買って頂いてありがとうございます。私はレポート用紙(@gokinaka)といいます。
今回はみんな大好き Arduino のお値段を安く仕上げることで、あちこちにばらまいて、ちょっと生活を便利にするための一つの方法について書きたいと思います。なお、ハードウェアは標準的に使われている Arduino Uno を前提に話しています。

Arduino はここがすごい

この本を読むような方々なら Arduino は多分聞いたことがあるし、触った事がある方も沢山いると思いますが、簡単な説明と Arduino のすごいところを話します。

-Arduino はマイコンではない
ってのは言い過ぎかもしれません。Arduino はマイコンのハードウェア的要素と、プログラミングのソフトウェアな要素を含んだ統合開発環境です。今までのマイコン開発はマイコン本体をはじめ、書き込み器やコンパイラなど沢山用意するものがありました。相性問題などもありました。
Arduino はこれらを全てパッケージングしました。下の図のようにハードウェアと PC 側のソフトウェアの世界でまとめて一つにしました。



-オープンソースと数の暴力
Arduino はイタリア生まれのオープンソースなハードウェアです。利用する際に殆ど縛り的な要素がありません。勝手に互換機を作ってもいいですし(ただし Arduino と名付けていいのは本家のみ)、Arduino に取りつけるセンサなどのモジュールも勝手に作って売っても構いません。
それにくわえて、車輪の再生産をしなくても(勿論勉強のためにしてもいいと思う)、自分の欲しい物が簡単につくれます。
また利用者が世界に沢山いるので、もし分からないことがあったり、詰まってしまってもググレカスの精神で解決する事が多いです。

-L チカまで 10 分

L チカは LED をチカチカさせるという意味です。プログラミングな世界でいう、Hello World 的なやつだと思ってください。これくらいできてあたりまえに見えますが、今までのマイコンはここまで行き着くのが非常に大変でした。

Arduino は違います。PC があれば開発環境をインストールして、USB で繋いで、サンプルプログラムから Blink を選択して、コネクタに LED を差し込むだけです。

快適なネット環境なら 10 分でここまで動かせます。最初のハードルが低いので、初心者やちょいと電飾を制御したいデザイナーさんなどにも気軽に使ってもらえます。

ここがどうにかしたい

ここまで Arduino についてだいぶ褒めちぎりましたが、私が個人的に Arduino に不満に思っていることがあります。お値段が高いことです。大手通販なスイッチサイエンスにて 3024 円で販売しています。(2014/11 現在)

正直お財布にはしんどい価格です。一つならともかく、何個も揃えて事を成したいと考えた時、ちょっと見逃しがたい値段設定です。

Arduino のマイコン本体である AVR ATmega328P は、秋月電子にて 1 つ 250 円で売っています。まとめ買いで更にお得です。つまりその周辺回路で結構な金額が掛かっていると考えられます。

その部分、本当に必要?

本題である Arduino の要らない部分を省いてお値段をお安くする話です。

あなたが Arduino である物を作りたいと考えました。例えば、それが PC から制御したいとか考えるならば、普通にお手持ちの Uno を使えばいいと思います。

しかし時として、作りたいものが、PC と Arduino を接続しないで動作するような使い方しか行わない、3.3V の電圧なんて要らないよ! という場合があります。

この際、UNO の周辺回路でも比較的高価な USB と AVR を繋ぐシリアル変換回路は最初にスケッチ(プログラム)を書き込んでお役御免です。もったいないと思うのです。

そこで Arduino の要素の内、必要な部分だけ集めて、安く、コンパクトに Arduino を使う方法を解説します。

必要なもの

今回は Arduino UNO の ISP(In-System Programmer)を利用して、新しい AVR マイコンにブートローダ(Arduino の OS みたいなもの)とスケッチを書き込みます。
動作させるのに最低限必要な物は以下のとおり。この部品表はスケッチ書き込み用回路とその後の動作確認に必要な部品を記しています。

部品名	型番、容量	個数
Arduino UNO		1
AVR	ATmega328P	1
抵抗	1k	1
	10k	1
セラミックコンデンサ	22pF	2
電解コンデンサ	10uF	1
ブレッドボード	お好きなもの	1
LED	赤	1

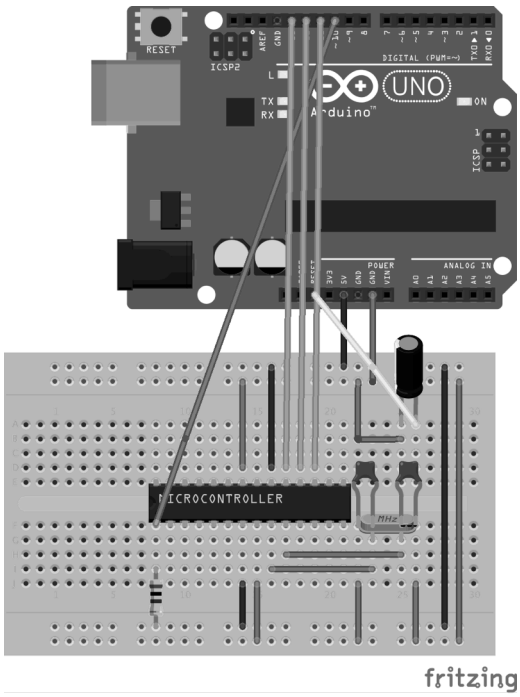
ブートローダとスケッチの書き込み

Arduino に使われているマイコンは AVR といいます。しかし、そのままでは Arduino のスケッチを書き込むことはできません。その前に ブートローダを呼ばれるプログラムを書き込む必要があります。

-ブートローダの書き込み
先でも少し触れましたが、Arduino UNO とソフトウェアには Arduino ISP という、AVR への書き込み器として使える機能が標準的に搭載されています。これを使ってブートローダを書き込んでいきます。PC と書き込み先の新しい AVR との間で UNO に橋渡しをお願いする形です。
構成する回路図、及びブートローダを書き込みする動作手順は Arduino 公式 (http://arduino.cc/en/Tutorial/ArduinoISP)　に載っています。Arduino ISP でググると出て来ると思います。今回は分かりづらいところもあるので細かく解説します。

1.UNO へ Arduino ISP の書込
ファイル＞スケッチの例＞Arduino ISP を開いて ISP スケッチを取り出します。その後、Arduino にスケッチを書き込みます。

2.回路構成とブートローダの書込
下のような回路を構成します。電解コンデンサの方向にご注意ください。RESET 端子と電解コンデンサの＋を結線します。また、ここで使用している抵抗は 10k です。



次に PC 側で、ツール＞書込装置＞Arduino as ISP を選択します。
最後にツール＞ブートローダを書き込む を選択すると新しい AVR にブートローダが書き込まれます。
また、16MHz のセラロックをお持ちの方は水晶振動子と 22pF の部分を置き換えても大丈夫です。セラロックではちょっとコンデンサの値が違いますが、書き込み段階では正常に動作しました。

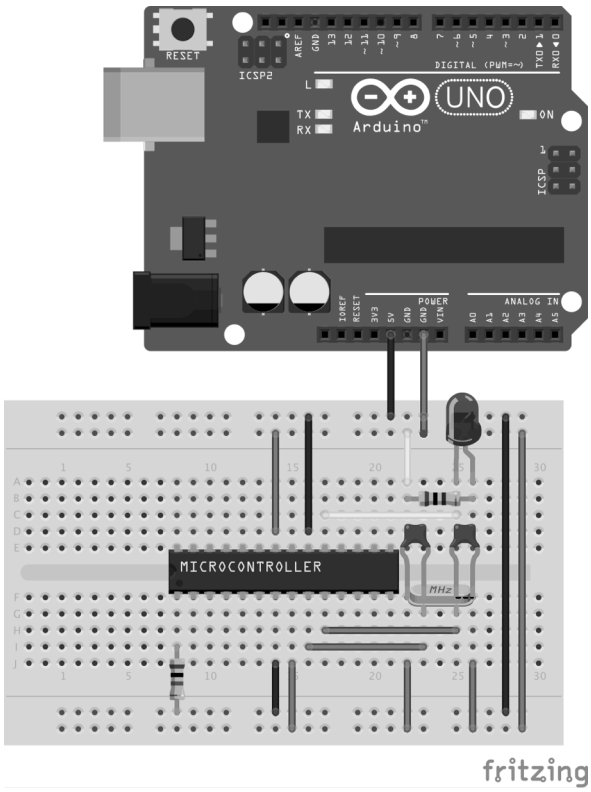
-スケッチの書き込み
次にスケッチを書き込みます。ここでは開発環境のサンプルに含まれている、LED を点滅させる Blink というスケッチを書き込みます。

1.回路構成の変更
基本的に回路構成は同じですが、ブレッドボード上の電解コンデンサを取り除きます。

2.スケッチの書込
まず、ファイル＞スケッチの例＞01.Basics＞Blink を開きます。
次にファイル＞書込装置を使って書き込む を選択すると、スケッチが書き込めます。

単体での動作確認

上記の手順で無事 Blink スケッチが書き込めたとします。単体での動作を確認するためにブレッドボード上の AVR の digital pin13 と GND の間に LED と 1k 抵抗を直列に接続してみてください。
ここでも Arduino が出てきますが、単純に 5V の電源を取り出しているだけです。お好みの電源をお使いください。
うまく行っている場合、LED が点滅します。



ここまでが書込及び動作確認の手順です。うまくいかないようならば、回路構成を見直す、エラーの内容を調べてみるなどしてみてください。きっと他の誰かが同じ内容で詰まっているはずです。

ここから、スケッチを書いて周辺機器と連携させれば安価に色々な事ができるようになります。

どうやって使うか

安い Arduino が作れたはいいけど、何に使えばいいの?という時の話。

Arduino が得意とするのはセンサなどとの連携や照明などアナログな器具の制御です。また消費電力が小さいので、24 時間動きっぱなしや、バッテリーで動作させる時に力を発揮します。

私は外付けの冷蔵庫開けっ放し防止装置などを作りました。扉にリードスイッチを付けて、扉の開閉を検知。20 秒たっても閉まらないとブザーが鳴る仕組みです。電源は冷蔵庫近くの USB 充電器から引っ張ってきています。この一人暮らし用の冷蔵庫には地味についてない機能で、一晩中開けっ放しで冷凍庫の中身が全滅という憂き目に合わないように。

工作例やアイデアが他にも沢山出てきているので、是非調べてみてください。

おわりに

買うほどの物ではないけども、あると便利な物。Arduino はそのような生活を少し便利にするアイテムを作るのに向いているシステムだと私は考えています。

この時、自作して価格が安く収まれば、作る時のハードルも低いし、壊れても作りなおせばいい。いいことずくめ！ と思って今回の記事を書きました。

こういう記事を書くのは初めてで、わかりづらいところも沢山あると思いますが、少しでも自作に対するハードルを下げることができたら幸いです。

人間と酔い

風土@fuudo_food_0309

1. 人間と酔い

人間は酔う。酔いには、アルコールによるものだけではなく、乗り物に乗ることによる「乗り物酔い」、３D 映像を見ることによる「３D 酔い」などがある。特に乗り物酔いは人間だけでなく、動物にも起こる。

また、この 50 年ほどで現れた比較的新しい種類の酔いも存在する。「宇宙酔い」である。人類が地上を離れ、宇宙空間へと飛び出すまで決して経験することのなかった「宇宙酔い」が、宇宙飛行士たちを苦しめる。

「酒酔い」「乗り物酔い」「3D 酔い」「宇宙酔い」・・・など、人間は様々な酔いというものに直面するが、「酒酔い」と「それ以外の酔い」には明確な違いがある。それは、「酒酔いを好む人はいるが、その他の酔いを好む人はまずいない」という点である。私は、「酒酔い」と「その他の酔い」の差異と、「その他の酔い」の防止法についてまとめていこうと思う。

2. 酔いの種類

2.1　酒酔い

飲酒をすると人間は酔う。もちろん程度の差こそあれ、全く酔わない人はいないだろう。

人間がアルコールを摂取すると、脳内では「ドーパミン」と呼ばれる神経伝達物質の分泌が促進される。これは「心地よさ」や「楽しさ」と言った感情を生み出す作用がある物質とされており、飲酒をすることで気分が高揚したり、朗らかになったりするのはこのためである。だが、酒の酔い方は人それぞれである。例えば、私は非常に酒に弱く、酒を飲んだところで「心地よさ」を感じることができず、逆に悪心を感じてしまう。酒が強い、弱いというのは遺伝的影響が強く、慣れることによって改善されるということはないというのが一般的である。

2.2　乗り物酔い

車酔い、飛行機酔い、船酔いなどをまとめて「乗り物酔い」とする。これは酒酔いとは全く異なるメカニズムで起こる「酔い」である。

動物の耳内には「三半規管」と呼ばれる臓器があり、体の傾きを検知する。ロボットで言うとジャイロセンサーがこれに当たる。これと、目による視覚情報との動きのずれ（視覚情報では景色が右に動いているのに、三半規管では左に動いているなど）が発生すると、ひとによっては酔ってしまう。これは、「二つの情報のズレは、自然状態ではあってはならない

現象である」→「体が誤作動を起こしている」→「危険信号としての酔いを起こす」
というプロセスを踏んでいるとされている。

2.3 3D 酔い（ゲーム酔い）

基本的なメカニズムは乗り物酔いに準ずる。3D 映像などを視聴した際、体は静止したままなのにも関わらず視覚情報は「体が動いている」と認識してしまうと、乗り物酔いに似た症状を起こすことがある。これは、ＦＰＳ視点（First Person Shooting）の動画で顕著に起こるとされている。日本人は特にＦＰＳ動画に酔いやすいと言われており、ＦＰＳゲームがあまり流行しない要因の一つと考えられている。

2.4　宇宙酔い

宇宙飛行士が微小重力環境において、身体のバランスが保てなくなり、吐き気やめまいを起こす。我々が発症ことはまずないであろうが、宇宙飛行士にとっては死活問題である。日本人宇宙飛行士古川聡氏曰く「毎日が船酔い」「死にそう」だという。また微小重力下で嘔吐した場合、気管に吐しゃ物が詰まって窒息する可能性がある。宇宙酔い改善を目指す研究は、地上の乗り物酔いにおける対策にもつながると期待されている。

上記の通り、酔いには様々な種類があり原因もそれぞれであるが、どれも「悪化すると吐き気や嘔吐をもたらす」という点では一致している。
では、これらの症状がなぜ起こるのかをまとめることにする。

3. 酔いと自律神経

動物の体には「自律神経」と呼ばれるものがある。これは、自らの意思では動かすことができない生理現象（心臓の鼓動、食物の消化など）を司っており、これが停止することは生物としての死を意味する。

先述のとおり、乗り物酔いは目からの視覚情報と三半規管からの傾き情報との祖語が原因であり、言い換えれば「センサーから得られたデータの不自然な干渉」である。つまり、センサーから得られたパラメータに不具合がある場合、それらの情報をまとめて処理する脳も混乱が起きる。得られた情報は自律神経からの出力データにも当然影響を与えるため、自律神経によって動かされている内臓は影響を受けやすい。特に、強い消化液を分泌し続ける胃は自律神経からの影響を強く受けるため、吐き気を催すのである。

また、吐き気とは「体内に遺物が混入した際の危険信号」と同じであり、ありもしない異物を体外に排出しようとして嘔吐するのである。食中毒などで嘔吐を発症するのも、毒物を排出しようとする人体のメカニズムである。

まとめ

・酔いとは人体を守ろうとする反応のひとつである

参考文献：

なぜ楽しくなるのか～脳とアルコールのメカニズム～

http://www.health.ne.jp/library/3000/w3000753.html

酒の飲み過ぎに要注意　練習すれば強くなるは気のせい

http://www.zakzak.co.jp/economy/ecn-news/news/20140618/ecn1406180830007-n1.htm?view=pc

酔うってどういうこと？

http://www.sapporobeer.jp/tekisei/shikumi/yoi.html

乗り物酔いのメカニズム

http://www.ssp.co.jp/aneron/mechanism/

信州大学医学部耳鼻咽喉科学講座 NASA が採択「宇宙酔い解明」への道

http://www.shinshu-u.ac.jp/special/research/2011/12/45596.html

日本語音声合成エンジンとしての初音ミクを使用した研究・開発の紹介

michele_lap†

1. はじめに

本稿は初音ミク[1]の日本語音声合成エンジン論やメディア論でなく，日本語読み上げソフトの音声として「初音ミク」を使用し研究・開発の紹介である．

2. 初音ミクとは

初音ミクについては，情報処理学会会誌でも特集[2]を組まれており，説明の必要が無いと思うのでここでは省く．

3. 日本語音声合成エンジンとは

日本語音声合成エンジンについては誌面の都合上省くが，ここではテキスト読み上げ（TTS:text-to-speech）として利用している．

初めは Microsoft Speech Platform の日本語音声合成エンジン「Microsoft Server Speech Text to Speech Voice(ja-JP, Haruka)」を使用していた．読み上げた音声に不満は全くなかったが，インパクトがほしかったため初音ミクに変更した．変更後は，初音ミクの知名度もあり，開発そのものにより注目してもらえるようになった．変更後の音声を初音ミクとして認識する割合は 100%である．

4. 研究・開発物

研究・開発物の一部を紹介する．

4.1. ESS ロボットチャレンジ（MDD ロボットチャレンジ）

ESS ロボットチャレンジ[3]に，モデルベースアプローチ(MBA)による組込みソフトウエア開発 PBL(Project Based Learning)の実践を行い，得られた知見を研究にフィードバックする自己研鑽の場として活動している．飛行船制御組込みソフトウエアを商業ソフトウエア開発を意識し UI 作成し，航行状況をテキストとグラフィックで表示する機能飛行船の状況を初音ミクの音声で知らせる機能を実装した．

4.2. Leap Motion を用いた子ども向けゲーム

学園祭の展示物として「Leap Motion」を使用した画面のぐんまちゃんをなでるように手をかざすと，ぐんまちゃんが動いて初音ミクの音声で話しかけてくれる子ども向けゲームを作成した（図 1）[4].

図 1　Leap Motion を用いた子ども向けゲーム

4.3. ET ロボコン

ET ロボコン[5]にも ESS ロボットチャレンジと同じ理由で参加している，アーキテクト部門の PV とパフォーマンスの演出に「女性キャラクターの声」として使用した．

4.4. 群馬大学工学部歌「関東八州」

今までは TTS としてしか使用していなかったが，初音ミク本来の使い方である，歌わせてみたものである[6].

5. まとめ

初音ミクに歌ってもらうことだけでない，TTS の音声として使用した研究・開発の紹介した．初音ミクには，このように幅広い利用方法がある．

参考文献

[1] http://www.crypton.co.jp/mp/pages/prod/vocaloid/cv01.jsp
[2] 情報処理 2012 年 05 月号別刷「《特集》CGM の現在と未来: 初音ミク，ニコニコ動画，ピアプロの切り拓いた世界」 https://www.ipsj.or.jp/magazine/5305.html
[3] Ehttp://www.sigemb.jp/rc/2012/
[4] http://www.iggg.org/news/meia-2014/
[5] http://www.etrobo.jp/2013/
[6] http://ftp.iggg.org/public/hasshu.mp3

†群馬大学電子計算機研究会 (IGGG)

SCADE Suite（モデルベース開発ツール）による Arduino 入門

michele_lap†

1. はじめに

本稿はモデル開発ツール SCADE Suite [1]による Arduino[2]入門である. SCADE Suite による Arduino のいわゆる L チカの基本的な操作を述べ, モデルベース開発ツール SCADE Suite による Arduino 開発に興味をもってもらえればと考える. なお, ここで述べる方法は, 筆者の環境で動作できたものであり, リファレンスがないため, 正式な方法かどうか不明である.

2. Arduino とは

Arduino とは, フィジカルコンピューティングのためのオープンプラットフォームである[2]. ここでは Arduino Uno や Arduino Mega 2560 といった基本的な Arduino ではなく, Intel Edison Kit for Arduino と Intel Galileo Gen 2 を使用する（図 1）.

図 1 Intel Edison Kit for Arduino と Intel Galileo Gen 2

使い方やインストール方法などは, Intel Galileo をはじめよう[3]などに詳しく書かれているのでそちらを参考にしてほしい.

3. SCADE Suite（モデルベース開発ツール）とは

SCADE Suite とは, モデルベース開発によるセーフティクリティカルな組込み制御ソフトウェア開発支援ツールで, 厳格なセマンティクスを持つグラフィカル言語を使用して, 上流から一貫したソフトウェアの開発工程を、1 つのツール環境で実現する[1].

4. SCADE Suite の設定

4.1. SCADE Suite の基本的な使い方

SCADE Suite の基本的な使い方は Esterel Technologies 社より提供されている“Getting Started with SCADE Suite” のテキスト（英語）や“SCADE Suite E-Learning”の動画（英語）などを参考にしてほしい. E-Learning の一部が YouTube に公開されているため, ユーザではなくても視聴することができる[4]. また, ここではモデルの検証は単純なシミュレーションしか行っていない.

4.2. SCADE Suite のモデル

SCADE Suite のモデルを図 2 に示す. これは, サンプルプログラムの「fade」スケッチを SCADE Suite に変換したものである. LED をフェードインさせるものであり, LED の明るさを led_val として出力するものである. led_val の Last は 0 とする. led_val の Last と 1 を加え, 250 より小さいか比較して, 小さければその値を led_val に代入し, 大きければ led_val に 0 を代入する.

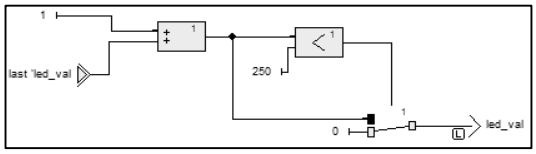


図 2 SCADE Suite のモデル

4.3. SCADE Suite のシミュレーション

SCADE Suite モデルを作成したらまず,「チェック」を行いエラーがないか確認する. 問題が無ければ, シミュレーションを行い, テストを実行する. 図 3 に 10 ステップ毎に実行したシミュレーション結果である. led_val が 255 を超えると 0 になっており, こちらの意図通り動作していることが分かる.

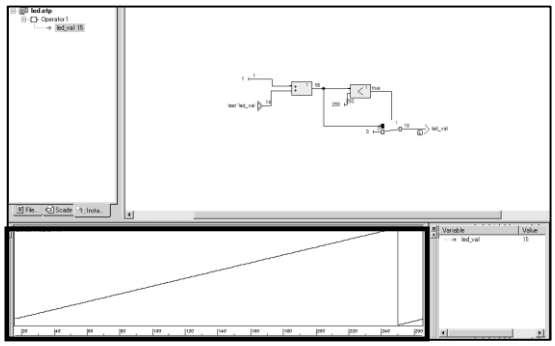


図 3 SCADE Suite のシミュレーション結果

†群馬大学電子計算機研究会 (IGGG)

4.4. C コード生成

「自動生成」を行うとSCADE Suite モデルからCコードが自動生成される. 自動生成されたCコード(図4, 図5)は, セーフクリティカルな業界からの認証取得済みである.

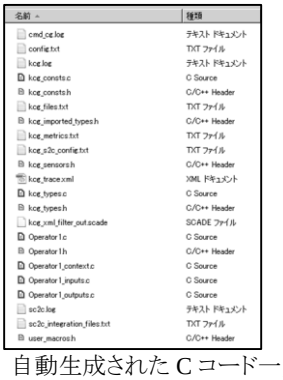


図 4　自動生成された C コード一覧

```
/* $************** KCG Version 6.1.3 (build i6) **************↓
** Command: s2c613 -config C:/Users/pro/Desktop/led/KCG¥kcg_s2c_config.txt↓
** Generation date: 2014-12-02T01:11:12↓
*****************************************************************$ */↓
↓
#include "kcg_consts.h"↓
#include "kcg_sensors.h"↓
#include "Operator1.h"↓
↓
void Operator1_reset(void)↓
{↓
  Ctxt_Operator1.init = kcg_true;↓
}↓
↓
/* Operator1 */↓
void Operator1(void)↓
{↓
  /* Operator1::_L7 */ kcg_int _L7;↓
  ↓
  if (Ctxt_Operator1.init) {↓
    Ctxt_Operator1.init = kcg_false;↓
    _L7 = 1 + 0;↓
  }↓
  else {↓
    _L7 = 1 + led_val;↓
  }↓
  if (_L7 < 250) {↓
    led_val = _L7;↓
  }↓
  else {↓
    led_val = 0;↓
  }↓
}↓
↓
/* $************** KCG Version 6.1.3 (build i6) **************↓
** Operator1.c↓
** Generation date: 2014-12-02T01:11:12↓
*****************************************************************$ */↓
↓
[EOF]
```

図 5　自動生成された C コード

5. Arduino の設定

Arduino では C 言語を呼び出せるため, プログラム本体は SCADE Suite で行い, ピンへの入出力のみ ArduinoIDE で行う. Arduino のスケッチを図 6 に示し, 一部内容を紹介する.

```
extern "C" {        //C言語の呼び出し
    #include "Operator1.h"
}

int ledPin = 9;       // LEDをピン9に接続

void setup() {
  // put your setup code here, to run once:
    Serial.begin(9800);       //デバッグ用にシリアル通信を行うための設定
}

void loop(void) {
  // put your main code here, to run repeatedly:
    Operator1();          //演算をSCADEで行う

    Serial.println(led_val);        //led_valをシリアルポートへ出力して改行

    analogWrite(ledPin, led_val);       //led_valをアナログ出力

}
```

図 6　Arduino のスケッチ

C 言語の呼び出しを図 7 に示す. 後述する SCADE Suite が C 言語を自動生成するため, C 言語の呼び出しをする必要がある.「Operator1」は SCADE Suite のプログラム本体である.

```
extern "C" {              // C 言語の呼び出し
            #include "Operator1.h"
    }
```

図 7　C 言語の呼び出し

loop 関数を図 8 に示す. まず, SCADE Suite のプログラムを「Operator1」関数として呼び出している. 次に led_val の値をアナログ出力している.

```
Operator1();              //SCADE プログラム本体

analogWrite(ledPin, led_val);
            //led_val をアナログ出力
```

図 8　loop 関数

6. Arduino への実装

6.1. 修正

自動生成されたファイルは KCG というフォルダに全て有り, これらのファイルを Arduino スケッチがあるファイルに全てコピーする.

先ほどの図 6 の Arduino スケッチを ArdinoIDE で検証するとエラーが出る. このエラーは, ArdinoIDE がインクルードすべきファイルを参照できないためである. h ファイルと c ファイルの全てのファイルを開き, インクルードするファイルとディレクトリの確認を行い, 修正が必要な場合は行う. 今回の内容では, 2 つのファイルを追加し, 参照するディレクトリを変更した.

6.2. 実装

修正後,「マイコンボードに書き込む」と LED がフェードインを行う.

7. まとめ

SCADE Suite による基礎的な Arduino の開発の方法を述べた. SCADE Suite は, セーフティクリティカルなモデルベース組込み制御ソフトウェア開発支援ツールのため, Arduino の開発ではメリットを享受できないかもしれないが, PBL(Project-Based Learning 課題解決型学習)として最適ではないかと考える.

参考文献

[1] http://www.esterel-technologies.com/products/scade-suite/

[2] http://www.arduino.cc/

[3] Intel Galileo をはじめよう: Matt Richardson 著 がじぇぴん訳:オライリージャパン, 2014

[4] https://www.youtube.com/playlist?list=PLazQFi--CxNOCjXrBKvoeGXLPpupoN-ru

あとがき

擬音（@gion_u)

ほんとは Python ネタで書きたかったけど、課題に追われてあるものから引っ張り出すしかなかった。じかんだいじに。

Sakuragi Yuuto

言語勉強会を通してポインタを理解したので、近頃他の言語も使ってみようかなと思っています。

Zakky (@ZakkyR)

部室にあまり行かず、Twitter でしか顔を出さない Zakky です。C87 では売り子としてもいますよ。

次回部誌があるなら今回のドライブレコーダーの製作期間が 3 日だったのでもうちょっと時間かけてまた Raspberry Pi でなにか作りたいです。

レポート用紙（@gokinaka)

この本を買っていただきましてありがとうございます。普段、殆ど活動していない幽霊部員ですが少し書かしてもらいました。拙い文章ですが、楽しんでもらえると嬉しいです

風土（@fuudo_food_0309)

もとは 3D 酔いについてまとめるはずでしたが、だんだん脱線して船酔いなどのまとめにもなってしまいました。計画性のなさを実感…うぅ……

群馬大学「電子計算機研究会」と言っておきながら、計算機とは全く関係ないものになってしまった……

桐生川（@kuriuzu)

「IGGG の部誌を作ってコミケで売ろう」と言い出したのはわたしです。言い出しっぺの法則でこの本の編集もやることになりました。まぁ、綺麗にまとまったのでよかったなーと。次回のコミケでも新しい部誌を作りたいと考えています。お楽しみに！！

発行日　2014/12/30

発行元　群馬大学電子計算機研究会

連絡先　infotech.gunmau@gmail.com

印刷　　街の印刷屋　ぱれっと

IGGG Journal "Lollipop" Vol.01, Comic Market 87 Edition

群馬大学電子計算機研究会 "Lollipop" Vol. 01 C87 版

2014 年 5 月に突如立ち上がったサークル、「群馬大学電子計算機研究会 (IGGG)」のメンバーが、ときには「進捗どうですか？」→「進捗ダメです！」と自問自答し、ときには某飲料に翼を授かりながら執筆した部誌第 1 号です。IGGG は Information technology researching society of the Gunmer, by the Gunmer, for the Gunmer の略称の模様。グンマーの、グンマーによる、グンマーのための IT 研究会、なのですが…今回、IT 系だけにとどまらず様々な分野を取り扱った記事を凝縮した結果が本書です。普段あまり読む機会のない専門外の内容なども見つかるかもしれませんので、その辺りもぜひお楽しみください。

IGGG

発行／群馬大学電子計算機研究会
@IGGGorg http://www.iggg.org/