1. マイコンを活用すると

「誰でもマイコンを活用すると役に立つ！」

インターネットやスマートフォンなどの普及によって，誰もが毎日 IT を基盤とするサービスを受けて生活している．最近では、様々な「モノ」がインターネットに接続され、情報交換することにより相互に制御し合い，人々に多彩なサービスを提供する概念である IoT（Internet of Things，モノのインターネット）が大きく期待されている．IoT を支える主な技術の一つはマイコンであり，スマホも高性能なマイコンによってできているものである．IT とマイコンを活用すると IoT だけではなく，これまでなかった新しい技術を生み出すことも，既存のものをより便利なものにすることもできる．

マイコンとは、マイクロコントローラ（Micro Controller）またはマイクロコンピュータ（Micro Computer）の略称であるが，厳密には両者は異なるものである．マイクロコントローラはマイクロプロセッサとも言い，コンピュータの頭に該当する部品のことを指す．一方，マイクロコンピュータは非常に小さいコンピュータのことを指す．昔はマイクロコントローラに様々な周辺部品を組み合わせてマイクロコンピュータを作ったが，近年の組み込みマイクロコントローラは，一つの部品の中にほとんどの周辺部品の機能が入っているため，マイクロコントローラとマイクロコンピュータを区別することは非専門家にとってあまり意味がない．本書でも，特に二つを区別せずにマイコンと呼ぶこととする．

マイコンは，ユーザのプログラムによって任意の機能を行うことができる，ほぼ万能なものである．一般的な電子回路とは異なり、回路を作り直したり、部品を取り替えたりしなくても，プログラムを変更するだけで機能の変更が可能であるため開発が楽である．10年前までは，マイコンは専門家またはマニアだけが用いる特殊機器であった．しかし今では，技術の進展により高い専門知識がなくても，誰もが少ない努力でマイコンを活用して高度なことができるようになっている．スキルが必要なハンダ付け作業や複雑な回路設計をしなくてもプログラミングの基礎的な知識とコンピュータさえあれば，誰でも自分の目的に合わせてマイコンを活用することができる。たとえば、家の様々な場所の温度を 24 時間計測して比較する自由研究の場合、マイコンと温度センサを組み合わせるだけで簡単にできてしまう．大学生が卒業研究で指向性アンテナを設計・試作してその精度を測る場合でも，マイコンとモータを組み合わせれば手間をかけずに自動的に大量の測定実験結果を得られる．趣味で周辺の明るさにあわせて光る飾りを作るのも，マイコンを用いれば簡単である．

以上の例は，あくまでもマイコンを利用したことのない人にその可能性を見せるためのものであり，アイディアと材料次第でどんなものでも作ることができる．本書中で紹介されたものを自分の手で製作してみることで，マイコンの基礎とその応用方法が理解でき，自らのアイディアを自由に実現できるようになるはずである．

2. OpenCM とは

本書では OpenCM というマイコンボードを用いる．OpenCM は ROBOTIS Inc.が開発したオープンソースの組み込みマイコンである．オープンソースとは，プログラムのソースコードなどが皆に公開されているという意味であり，OpenCM の場合は，回路図や開発環境のソースコードなどが公開されている．

組み込みマイコンとは，一般的な計算機とは異なり特定の家電製品や機械，装置に組み込まれるコンピュータシステムを意味する．組み込みマイコンの特徴は，LED やモータなどのハードウェア部品を制御するための関連回路が内蔵されていることが特徴である．OpenCM の特徴を簡単に列挙すると以下の通りである．

・専門家でなくても，容易にハードウェアを開発できる
・OpenCM とコンピュータさえあればプログラミングできる
・Windows，Mac など様々な環境で開発できる

・ハードウェアとソフトウェアが全て公開されている
・コードが簡単であり，容易に実装できる
・スマートアクチュエータを容易に利用できる
・様々なセンサモジュールが容易されている

最後の特徴として挙げた，スマートアクチュエータを容易に利用できる点と様々なセンサモジュールが容易されている点が，本書で OpenCM を用いる主な理由である．一般的に良く利用されるマイコンとしては Arduino があげられる．OpenCM よりもユーザは多く，これまでに蓄積された情報が多い．ただし，OpenCM の開発環境は，Arduino の開発環境をベースとしており，プログラミングにおいて文法や命令はほとんど同じである．そのため，OpenCM を利用することで，これまでに蓄積された Arduino の情報も利用しながら，スマートアクチュエータを利用することができる． OpenCM で利用できるスマートアクチュエータは，ROBOTIS Inc.のスマートサーボモータであるダイナミクセルシリーズである．OpenCM では，ハードウェアとしてダイナミクセル用のコネクタが用意されており，ソフトウェアとしてダイナミクセルを制御するための API が用意されている．そのため，スマートサーボモータの使い方を完璧に把握していなくても簡単に利用できる．さらに OpenCM では，ROBOTIS Inc.が生産しているセンサモジュールも使用できるため，電気電子回路の知識がない人でも簡単にセンサを用いた組み込みシステムが開発できる．

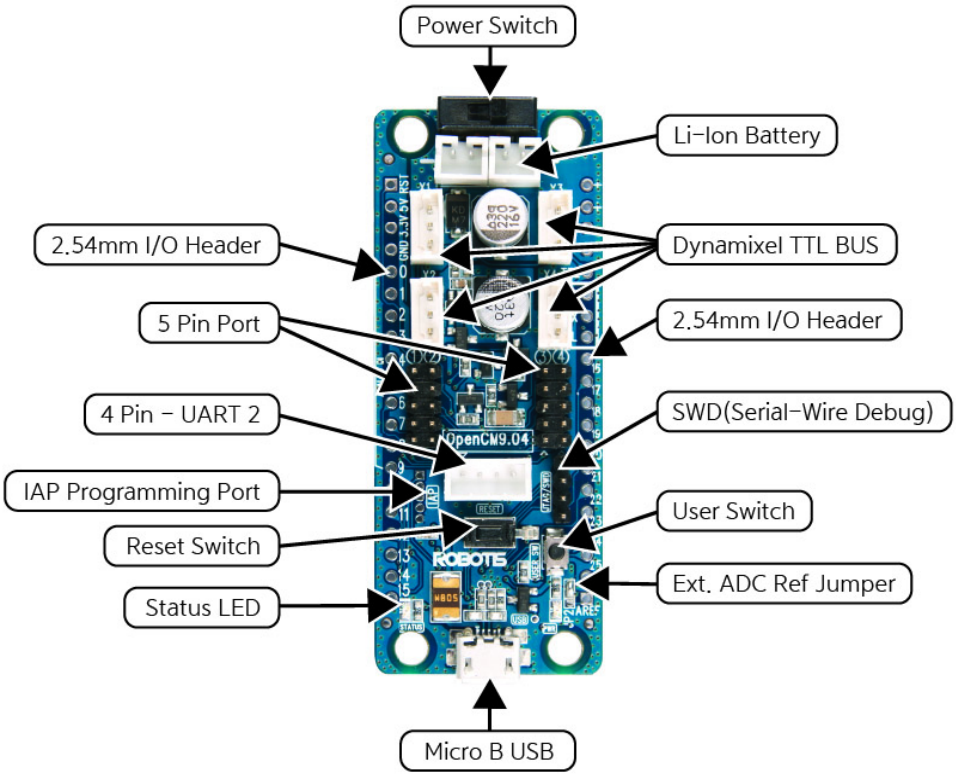


図 1．OpenCM9.04

図 1 は OpenCM の外観と各部分の名称である．”Power Switch”は電池からの電源を On/Off するトグルスイッチ，”Reset Switch”はマイコンボードをリセットしプログラムを最初から実行させるプッシュボタンスイッチ，”User Switch”はユーザからの入力を受け入れるプッシュボタンスイッチである．”Status LED”はプログラムによって点灯／消灯できる緑色 LED である．”Micro B USB”は Micro B type のコネクタであり，USB ケーブルでコンピュータと接続し作成したプログラムをマイコンにダウンロードするためのものである．”2.54mm I/O Header”は穴の間隔が 2.54mm でマイコンと外部とを繋ぐためのコネクタである．”2.54mm I/O Header”については，図 2 を用いてさらに詳しく説明する．他の部分に関しては必要に応じて改めて説明する．

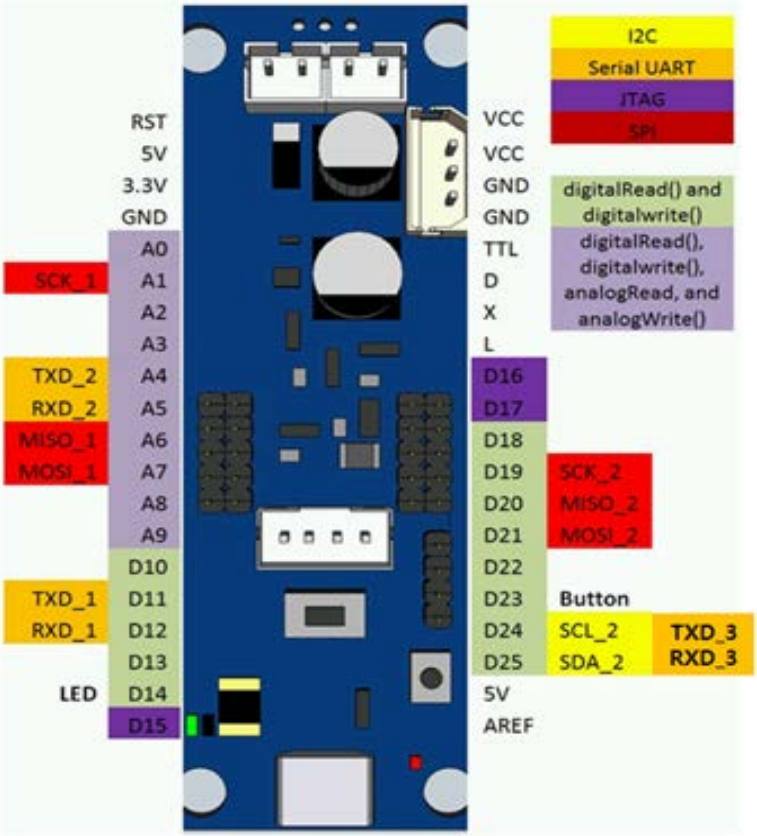


図 2. 汎用入出力（General Purpose Input/Output, GPIO）の配置図

図 2 に示す入出力信号ピン及び一部の電源ピンに関して左上から順に簡単に説明する.

RST	CPU のリセットピンと繋がっている.
5V	バッテリーをマイコンボードに繋いだときに 5V が出力される. （バッテリーがつながっていない時には 5V は出力されない）
3.3V	3.3V を出力する.
GND	グラウンド（グランドとも呼ぶ）. 0V の場所. 他の GND ピンともつながっている
A0～A1	デジタル入力またはデジタル出力またはアナログ入力
A2～A9	デジタル入力またはデジタル出力またはアナログ入力または PWM 出力
D10～D13	デジタル入力またはデジタル出力または PWM 出力
D14	デジタル入力またはデジタル出力または PWM 出力 （マイコンボード上の LED がつながっている）
D15	デジタル入力またはデジタル出力
VCC	バッテリーの”＋”とつながっている
GND	グラウンド. バッテリーの”－”とつながっている.
D16～D22	デジタル入力またはデジタル出力
D23	デジタル入力またはデジタル出力 （”User Switch”ボタンとつながっている）
D24～D25	デジタル入力またはデジタル出力
5V	5V が出力される.
AREF	ADC の基準

図 3 は OpenCM のハードウェアブロック図である。

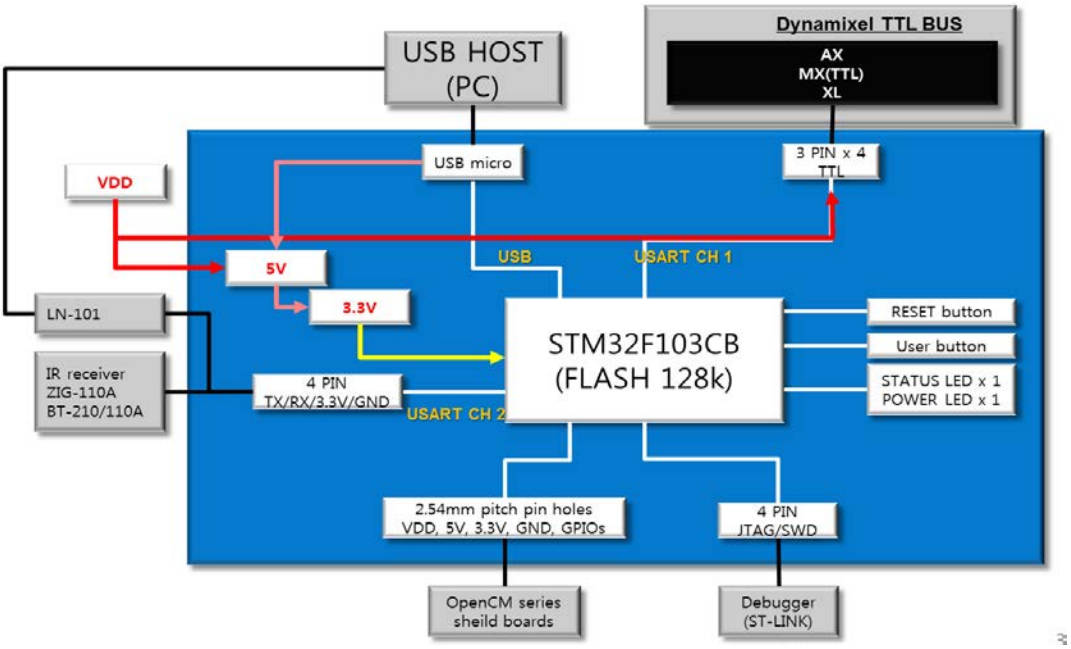


図 3. OpenCM のハードウェアブロック図

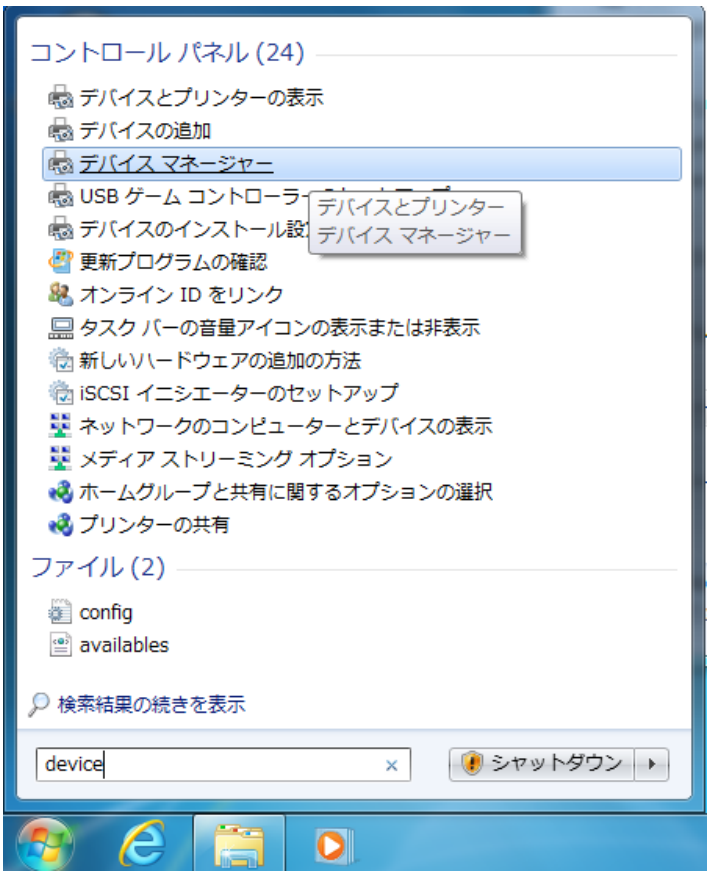
3. とにかく使ってみよう

3.1 OpenCM の認識からサンプルコードの実行まで

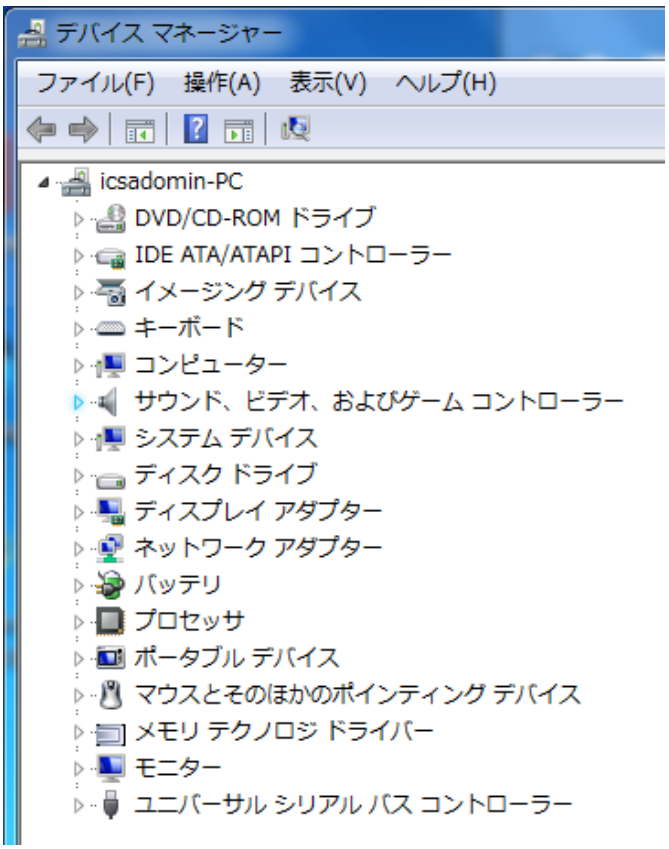
今回の創成2では OpenCM を中心に構成されたマイコントレーニングキットを用いる(図 4). OpenCM と様々な実験部品が入っている.

図 4 マイコントレーニングキット

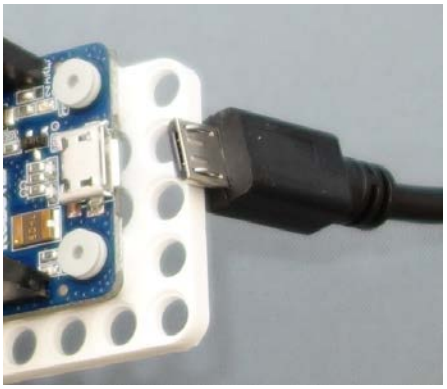
まずは Windows のスタートボタンを押して検索欄に device と入力すると，上に「デバイスマネージャー」というタイトルが現れる．それを実行しよう．

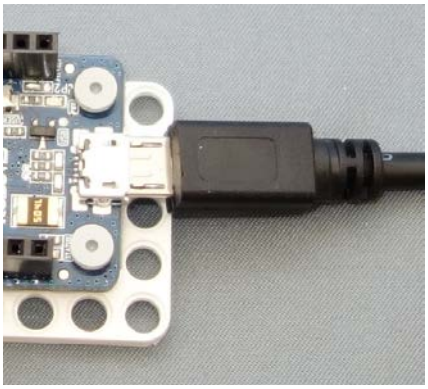


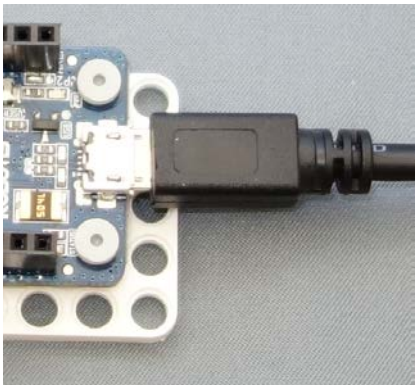
すると下のようなリストが現れる．



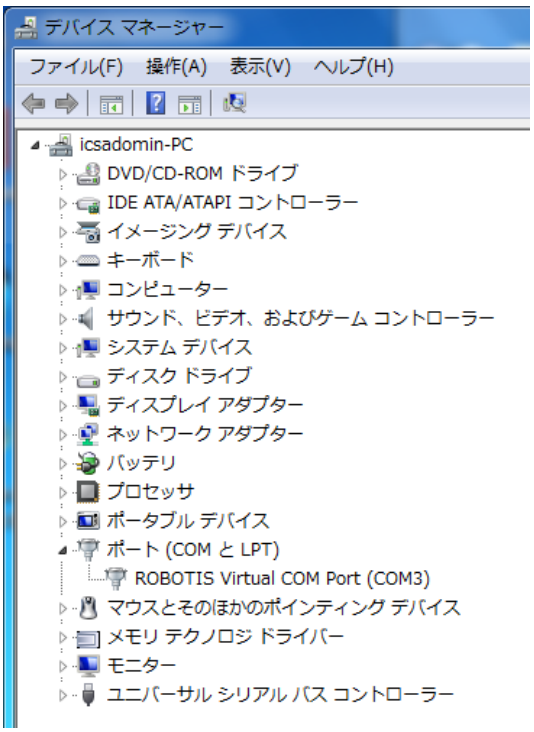
この状態で付属の USB ケーブルを OpenCM ボードに挿してみよう。差し込む際は USB ケーブルとボード側の形状をよく確認し，下図(a)のように正しい向きにして差し込もう。
(b)は，完全に差し込めていない状態である．(c)のようにしっかり奥まで差し込もう．その後，USB ケーブルの反対側を USB ハブに挿し，USB ハブのケーブルを挿したポートの横にあるボタンを押して OpenCM とコンピュータを接続しよう．

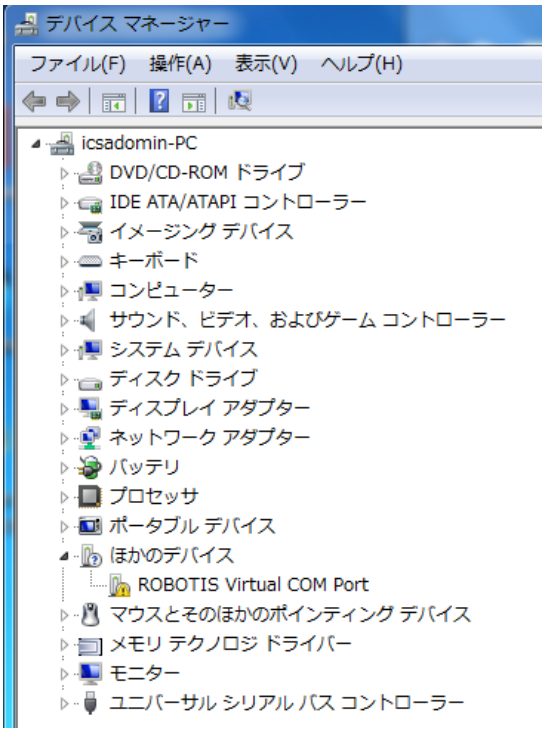

(a)


(b)

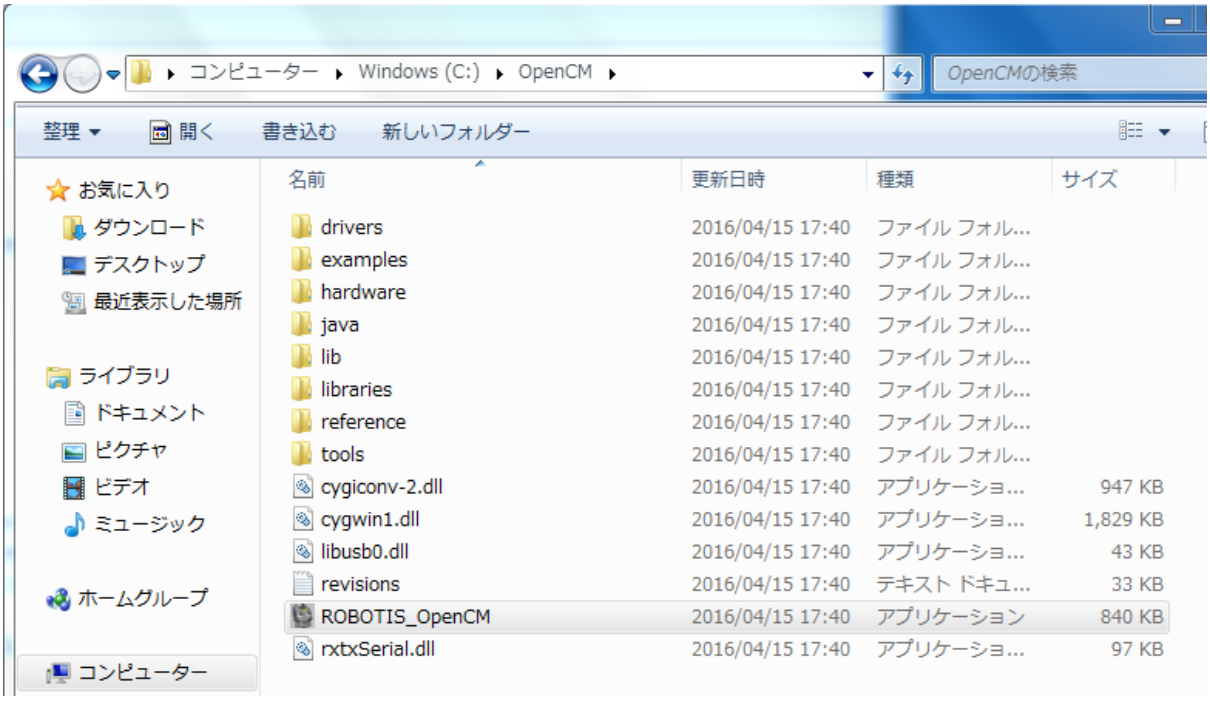

(c)

しばらく（数秒から数十秒）経って，ドライバーが正しく認識されるとデバイスマネージャーのポート内に「ROBOTIS Virtual COM Port」ができる．これが下の写真の左のように表示されればコンピュータが OpenCM を正しく認識しており，プログラミングの準備が完了したことになる．もし下の図の右のように，ほかのデバイス内に現れてしばらく時間がたっても変化がない場合には OpenCM を正しく認識していないことを意味する．この場合，ケーブルを抜いて数十秒程度待ってから挿し直してみる。もしこれを数回繰り返しても正常に認識しない場合には，コンピュータを再起動し，再度同じ手順で試してみるとよい．

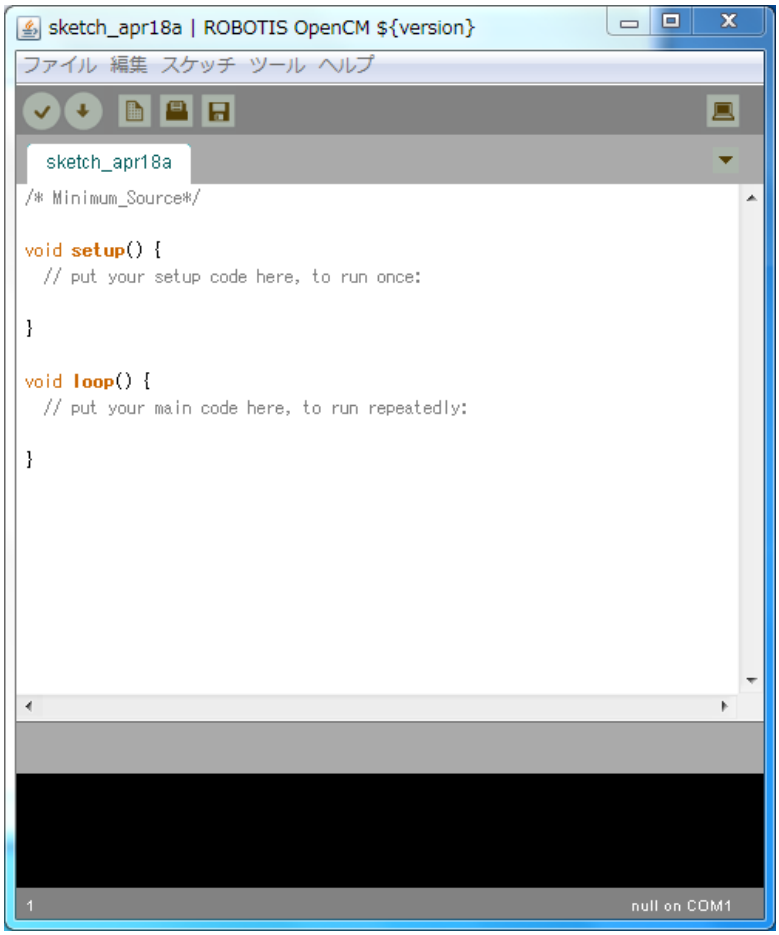




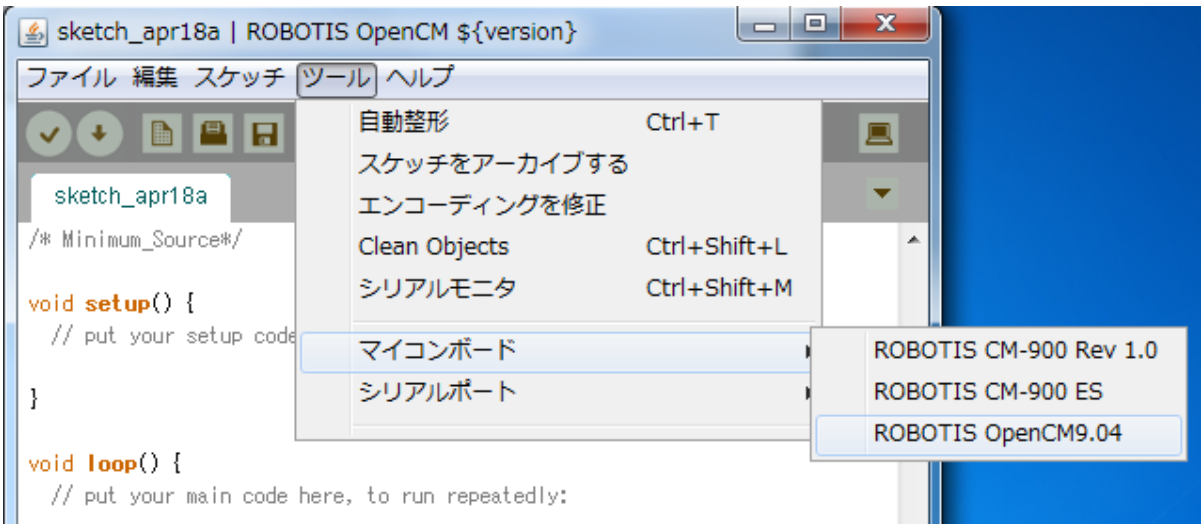
OpenCM を正しく認識したら，ソフトウェア開発環境である ROBOTIS_OpenCM を実行する．



下のような画面が現れたら，無事開発環境が実行されたことを意味する．

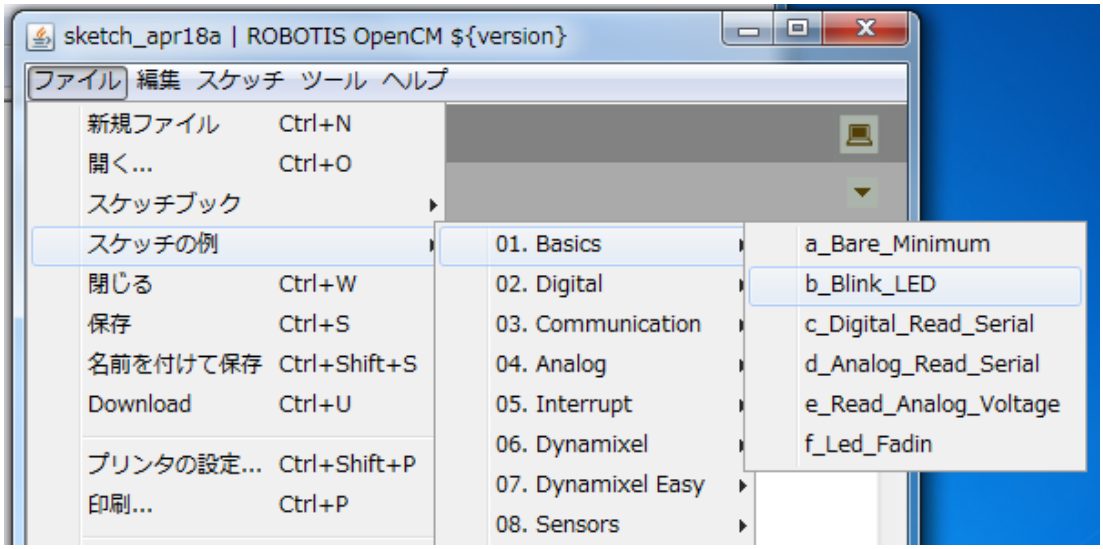


まずはターゲットのマイコンボードを選択しよう．「ツール」-「マイコンボード」内にある ROBOTIS OpenCM9.04 を選択する．



次に通信用のポートを選択しよう．「ツール」-「シリアルポート」内にある正しいポートを選択する．先ほどのデバイスマネージャーの画面にポート番号が出ているので，その番号と同じ番号を選べばよい．この資料の例では COM3 となっているが，実行環境や状況によって変化するため必ずしも COM3 とは限らない．自分の環境でポート番号を確認して，その番号を選択する．

次に，サンプルプログラムを実行してみよう．サンプルプログラムはスケッチの例というメニュー内に入っており，「ファイル」-「スケッチの例」から選択できる．ここでは，例の中から「01. Basics」-「b_Blink_LED」を選択する．



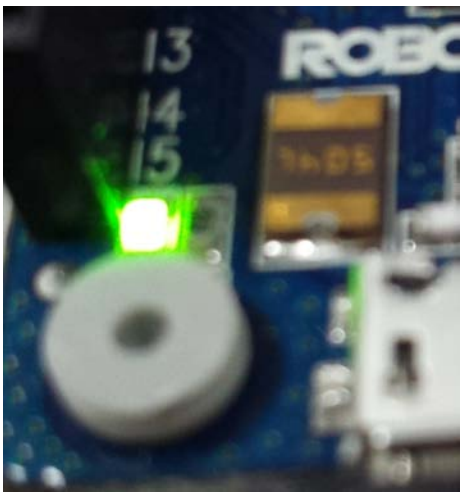
下のようなコードが読み込まれたかを確認してみよう。

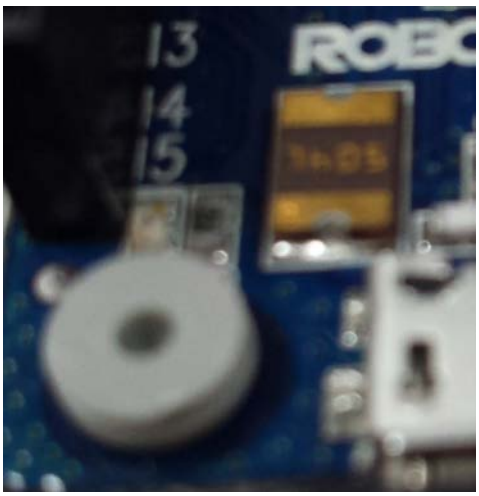
```
void setup() {
  // Set up the built-in LED pin as an output:
  pinMode(BOARD_LED_PIN, OUTPUT);
}

void loop() {
  digitalWrite(BOARD_LED_PIN, HIGH); // set to as HIGH LED is turn-off
  delay(100);          // Wait for 0.1 second
  digitalWrite(BOARD_LED_PIN, LOW);  // set to as LOW LED is turn-on
  delay(100);          // Wait for 0.1 second
}
```

正しくソースコード読み込まれたら，コンパイル，ダウンロードしてみよう．下の図を参考に、まずコンパイルボタンを押してコンパイルが完了するのを待つ．エラーがなくコンパイルができたら，次はコンパイルしたコードを OpenCM にダウンロードしよう．プログラムの実行はダウンロードが完了したら自動的に行われる．

正しくダウンロードが完了していれば下の写真のようにボード上の緑 LED が点滅する。





3.2 サンプルコードを理解しよう

動作確認ができたら OpenCM をコンピュータからはずして，ソースコードの内容を理解してみよう.

```
void setup() {
  // Set up the built-in LED pin as an output:
  pinMode(BOARD_LED_PIN, OUTPUT);
}

void loop() {
  digitalWrite(BOARD_LED_PIN, HIGH); // set to as HIGH LED is turn-off
  delay(100);          // Wait for 0.1 second
  digitalWrite(BOARD_LED_PIN, LOW);  // set to as LOW LED is turn-on
  delay(100);          // Wait for 0.1 second
}
```

OpenCM のコードは大きく2つに分かれており，入出力ピン(GPIO)の使い方等を設定する設定部（setup()関数）とプログラムの動作を記述するループ部（loop()関数）に分けられる．リセットボタンを押したり，電源を入れなおしたりすると OpenCM はまず setup()関数を 1 度だけ実行してから，loop()関数を繰り返し実行する.

まず setup()関数内について説明する．pinMode()という関数は GPIO の使い方を決める関数であり，

pinmode(BOARD_LED_PIN, OUTPUT)

とは BOARD_LED_PIN をデジタル出力ピンとして使用する，という意味である．BOARD_LED_PIN については¥hardware¥robotis¥cores¥robotis¥CM-904.h 内で

#define BOARD_LED_PIN　　14

と宣言されている．これで 14 番ピンがデジタル出力ピンになったので，そのピンに 1(HIGH)を出力したり，0(LOW)を出力したりできる．

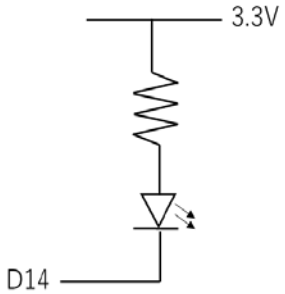
digiralWrite()は指定のピンに 1(HIGH)または 0(LOW)を出力する関数である。例えば，

digitalWrite(BOARD_LED_PIN, HIGH)

とは BOARD_LED_PIN(14 番ピン)に 1(HIGH)を出力するという意味であり，すなわち高い電圧（3.3V）を出力するという意味である．反対に

digitalWrite(BOARD_LED_PIN, LOW)

は 14 番ピンに 0 を出力する，すなわち低い電圧を出力するという意味である．OpenCM では右図のように緑色 LED がVccと 14 ピンの間に接続されているため，14 番ピンに 1 を出力すると LED が消えて，0 を出力すると LED が点灯する．

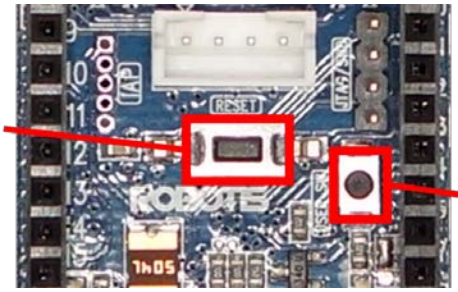


delay()は引数で指定される時間（単位 ms）だけ待ち時間を作る関数である．従って上のプログラムは，LED を消して 0.1 秒待つ，LED を点けて 0.1 秒を待つ，を繰り返すプログラムとなっている．

4. スイッチ入力を受けてみよう

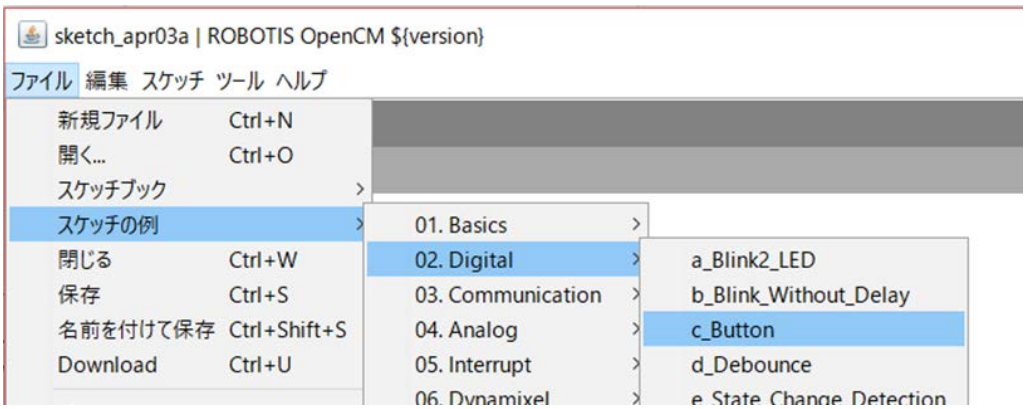
4.1 サンプルコードの実行

スイッチは日常生活でもよく使われているデバイスである．照明をつけたり消したりする際のスイッチ，テレビやエアコンのリモコンについているボタンスイッチなど，家電製品など様々な装置を操作するためにスイッチが使われている．もっとも簡単な形で人の意図を機械に伝えるためのデバイスといえる．コンピュータでも，キーボードやマウスなどに，ユーザの入力を受けるための多くのスイッチが使われており，マイコンでも最も簡単なユーザからの命令入力手段としてスイッチが使われる．OpenCM にはボード上に二つのボタンスイッチが存在する．



上の写真に，リセットボタンとユーザボタンを示す．中央のリセットボタンは，OpenCM ボードをリセットし再起動するときに使用する．右側のユーザボタンは，ユーザが自由に利用できるボタンスイッチであり，汎用入出力 D23（23 番ピン）につながっている．

このボタンスイッチのサンプルプログラムを実行して動作とプログラムを確認してみよう．「ファイル」-「スケッチの例」から「02. Digital」-「c_Button」を選択し，サンプルプログラムを開く．



このサンプルコードを 3 章の説明と同様にコンパイルし，OpenCM にダウンロードしてみよう．その後，OpenCM 上のユーザボタンスイッチを押したり離したりしてみよう．正しくコンパイル・ダウンロードできていれば，スイッチの操作に合わせて OpenCM 上の LED が点滅するのが確認できるはずである．

4.2 サンプルコードを理解しよう

このサンプルコードでは，GPIO の使い方等を設定する setup()関数において，pinMode()関数を用いて BOARD_BUTTON_PIN と BOARD_LED_PIN をそれぞれデジタル入力とデジタル出力ピンとして使用すると設定している．BOARD_BUTTON_PIN と BOARD_LED_PIN は¥hardware¥robotis¥cores¥robotis¥CM-904.h 内で

```
#define BOARD_BUTTON_PIN        23
#define BOARD_LED_PIN           14
```

と宣言おり，23 番ピンがデジタル入力ピンに、14 番ピンがデジタル出力ピンに設定されている．デジタル入力ピンに設定するということは，外部からの入力電圧を 1 または 0 の入力として受けることを意味する．

このサンプルでは，

pinMode(BOARD_BUTTON_PIN, INPUT_PULLDOWN);

と INPUT_PULLDOWN となっているが，これは指定のピンをデジタル入力ピンに設定し，さらに電圧の入力がない場合には 0V の入力とみなす，という意味である。反対に INPUT_PULLUP と設定すると，電圧の入力がない場合には 3.3V の入力とみなす，ということになる．OpenCM のデジタル入力ピンでは，0V に近い電圧はデジタルで「0」，3.3V に近い電圧をデジタルで「1」として扱うように設計されている．

次に，loop()関数では，

buttonState = digitalRead(BOARD_BUTTON_PIN);

によって D23 の入力された内容を buttonStaten に代入する．上ではユーザボタンスイッチは D23 に接続されていると述べたが，正確には D23 はボタンスイッチを経由して 3.3V に接続されている．つまり，ボタンスイッチが押されている間は，D23 は 3.3V と接続され，3.3V が D23 に入力される．押されていない時には D23 は開放され，何も入力されない．ただし，INPUT_PULLDOWN として設定したことにより，0V が入力されたとみなされる．従って、buttonState には、ユーザボタンが押されていれば「1」，押されていないと「0」が代入される．HIGH は 1，LOW は 0 として宣言されているため，ユーザボタンが押されていると OpenCM 上の LED が点灯し，ボタンを離すと消灯する．

```
void setup(){
  /*
   BOARD_BUTTON_PIN is needed to pull-down circuit
   for operationg as digital switch fully.
   */
  pinMode(BOARD_BUTTON_PIN, INPUT_PULLDOWN);
  pinMode(BOARD_LED_PIN, OUTPUT);
}

void loop(){
  // read the state of the pushbutton value:
  int buttonState = digitalRead(BOARD_BUTTON_PIN);

  if(buttonState==HIGH){ //if button is pushed, means 3.3V(HI
    digitalWrite(BOARD_LED_PIN, LOW);
  }
  if(buttonState==LOW){// if button is released, means GND(LO
    digitalWrite(BOARD_LED_PIN, HIGH);
  }

}
```