



mbed ライブラリ

作成から公開まで

「トラ技_mbedライブラリの作成方法の勉強会」 2014年11月7日向け資料

<https://atnd.org/events/57766>

version 1.1, 08-Nov.-2014



Tedd OKANO 作『mbed ライブラリ -作成から公開まで-』は
クリエイティブ・コモンズ 表示 4.0 国際 ライセンスで提供されています

注意： この資料は、いかなる団体の公式な見解を述べたものではありません。
全くの個人的な意見を述べたに過ぎません (^_^;

こんにちは

TEDD OKANO

https://twitter.com/tedd_okano

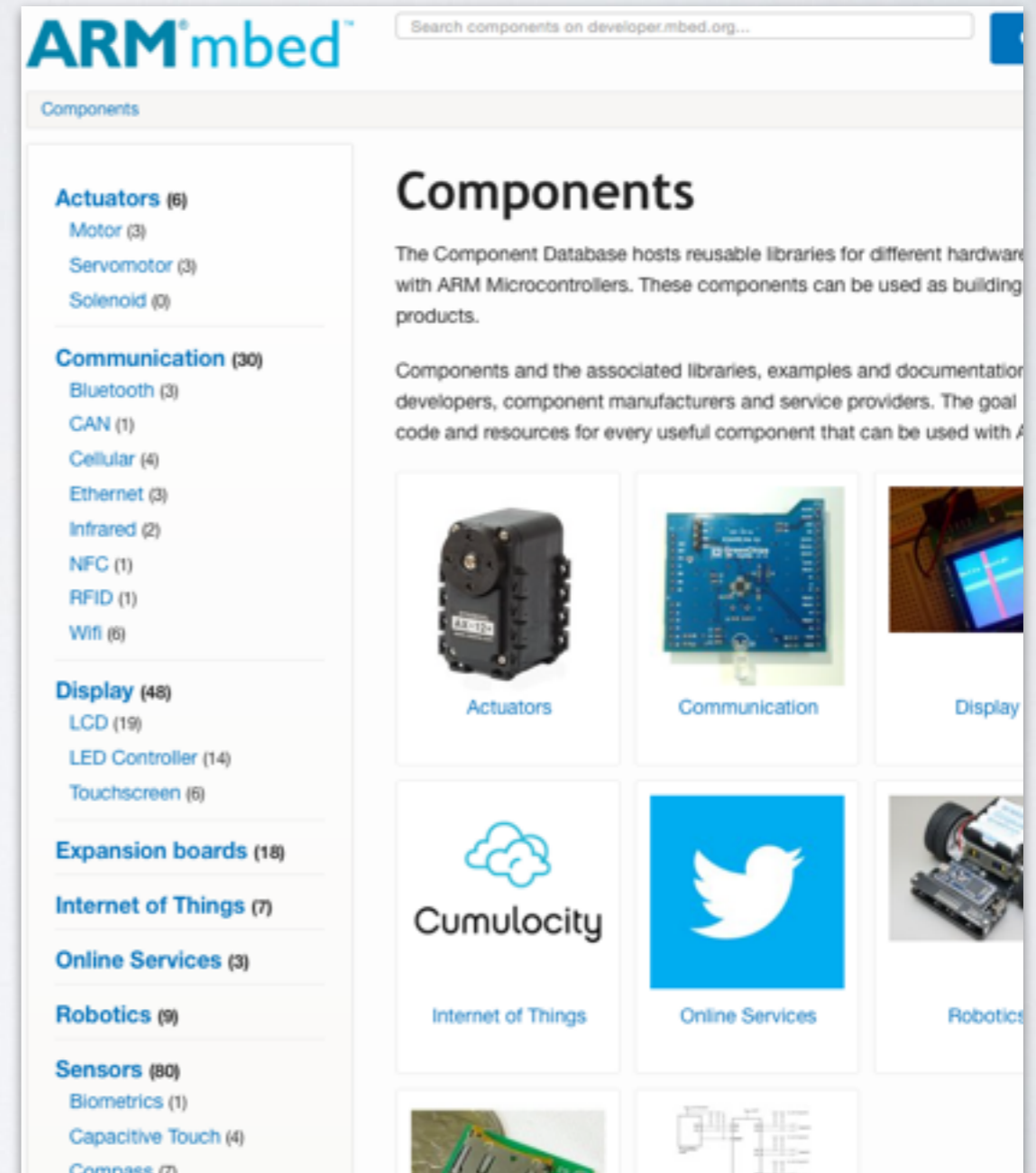
ペリフェラル系チップ・アプリケーション担当
トラ技10月号のI²Cの記事

<http://developer.mbed.org/users/okano/>
<http://developer.mbed.org/users/nxpfan/>
http://developer.mbed.org/users/nxp_ip/

The screenshot shows the user profile for Tedd OKANO on the ARM mbed developer website. The profile includes a cartoon avatar of a bear wearing glasses, the name 'Tedd OKANO', location '日本', and a Twitter link. A bio reads: 'Hello! I'm tedd. I'm an engineer for chips. Enjoy development with mbed! こんにちは!'. Below the bio are tabs for Profile, Activity, Notebook, and Code. The 'Activity' section shows recent updates to 'TextLCD_SB1602E wiki', 'SB1602E_test wiki', and 'SB1602E_Hello wiki'. The 'Public repositories' section lists several projects: 'SB1602E_test', 'SB1602E_Hello', 'SB1602E', and 'ika_shouyu_poppoyaki'. A map of Japan is also visible on the right side of the profile.

mbedのライブラリ

- 各ユーザが自由に公開
- mbed.org サイトで共有
- 簡単インポート→使用
- 部品箱の部品みたいなもの
- 『コピペで簡単』のカギ



```
#include "mbed.h"
#include "SB1602E.h"

SB1602E lcd( p28, p27 ); // SDA, SCL

int main()
{
    lcd.printf( 0, "Hello world!\r" );
    lcd.printf( 1, "pi = %.6f\r", 3.14159265 );
}
```

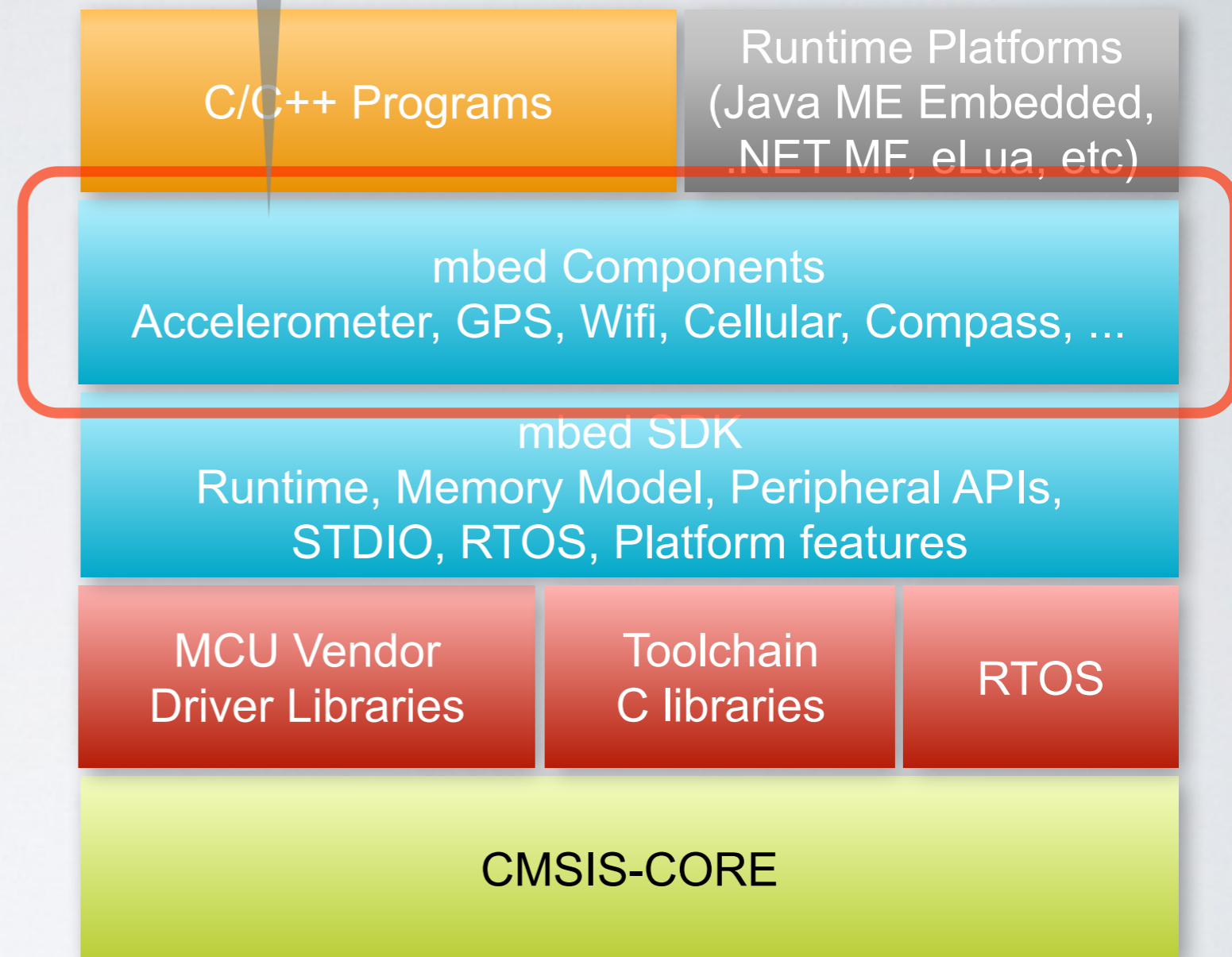
たとえば



ハードウェアばかりではなく、
ネットワーク・プロトコルなども

ライブラリ

- mbed-SDK (mbedライブラリ)
ハードウェアの高度な抽象化
- コミュニティでサポートされる
高機能ライブラリ群
- 必要があればレジスタにアクセス。
最下層レベルでの最適化



<http://developer.mbed.org/handbook/mbed-SDK>

<http://developer.mbed.org/components/>

developer.mbed.org

Platforms Components Handbook Cookbook Code Questions Forum Compiler

Dashboard (4)

ARM mbed

Search components on developer.mbed.org... Go

Hi, okano Logout

Components » Internet of Things

Internet of Things Add a component

Actuators (6)
 Motor (3)
 Servomotor (3)
 Solenoid (0)

Communication (33)
 Bluetooth (4)
 CAN (1)
 Cellular (5)
 Ethernet (3)
 Infrared (2)
 NFC (1)
 RFID (2)
 Wifi (6)

Display (51)
 LCD (20)
 LED Controller (14)
 Touchscreen (6)

Expansion boards (20)

Internet of Things (7)

Online Services (3)

Robotics (9)

Internet of Things

Internet Of Things

Xively

Nanoservice

HTML5 Websockets

MIMic Project

Axeda GO Kit for ARM mbed

AT&T M2X

Cumulocity

ハードウェアばかりではなく、ネットワーク・プロトコルなどもライブラリ/コンポーネントの対象

<http://developer.mbed.org/components/>

developer.mbed.org

net news weather computer mac man bike tv & radio life work 時刻表 HTMLカラーコード 顔文字屋 Pin It >> +

Platforms Components Handbook Cookbook Code Questions Forum Dashboard Compiler

ARM mbed Search mbed.org... Go Hi, okano Logout

Users - okano - Notebook - ライブラリ・コンポーネントの作りかた

ライブラリ・コンポーネントの作りかた Edit page Delete page

Page last updated 19 days ago, by Tedd OKANO. post a reply components, library

mbedのライブラリは非常に便利です。利用可能なライブラリはmbed.orgサイトに公開されていて、その使用法の単純なサンプル「HelloWorld」プログラムをそのままインポートして動作を試したり、あるいは自作のプログラム内にライブラリだけをインポートしてすぐに使うことができるようになっています。これらのコードは利用するだけでなく、自分で書いて公開することでコミュニティに貢献できます。

このノートページでは作成したライブラリを他の人にも使ってもらえるようにする方法を説明します。

ライブラリを作ったら、まず「公開(Publish)」します。もちろんこれだけでも、部品の型番やキーワードとなる単語でmbedページ内の検索に引っかかるようになりますから、使ってもらえるようになります。しかしより多くの人の目に触れるように「コンポーネンツ・データベース」に登録しておく、さらに使ってもらえるチャンスが増えるでしょう。

Information

このページはまだ書きかけです。このページは日本語でしか書かれていません。This page is available in Japanese only.

Information

英語版ではクラスライブラリの作りかた～公開の方法がこちらに紹介されています。Writing a Library

1. mbedライブラリの公開方法

1.1 ライブラリを作る

ここからいくつかの段階を経て、ライブラリを作る方法を説明します。ここに書いてある方法は、その1例と考えて下さい。他にも別の環境からコードを移植したり、違った手法で開発することも可能です。

1.1.1 テストプログラムを書いてみる ～まずは部品が動くことを確かめる～

簡単なデバイスを例に

ここでは非常に単純なI²Cチップ「PCA9547」を例に、ライブラリの作成手順を説明します。PCA9547は18のI²Cマルチプレクサで、8系統に分岐したI²Cのそれぞれの枝を有効化したり、あるいは全部の枝を無効化することができます。

多くのI²Cのスレーブを接続する場合のスレーブアドレスの重複回避や、I²Cバス容量負荷を軽減する目的で使うことができます。またマルチプレクサの前段でレベル変換(ロジック電圧レベルの変換)も行うことができます。

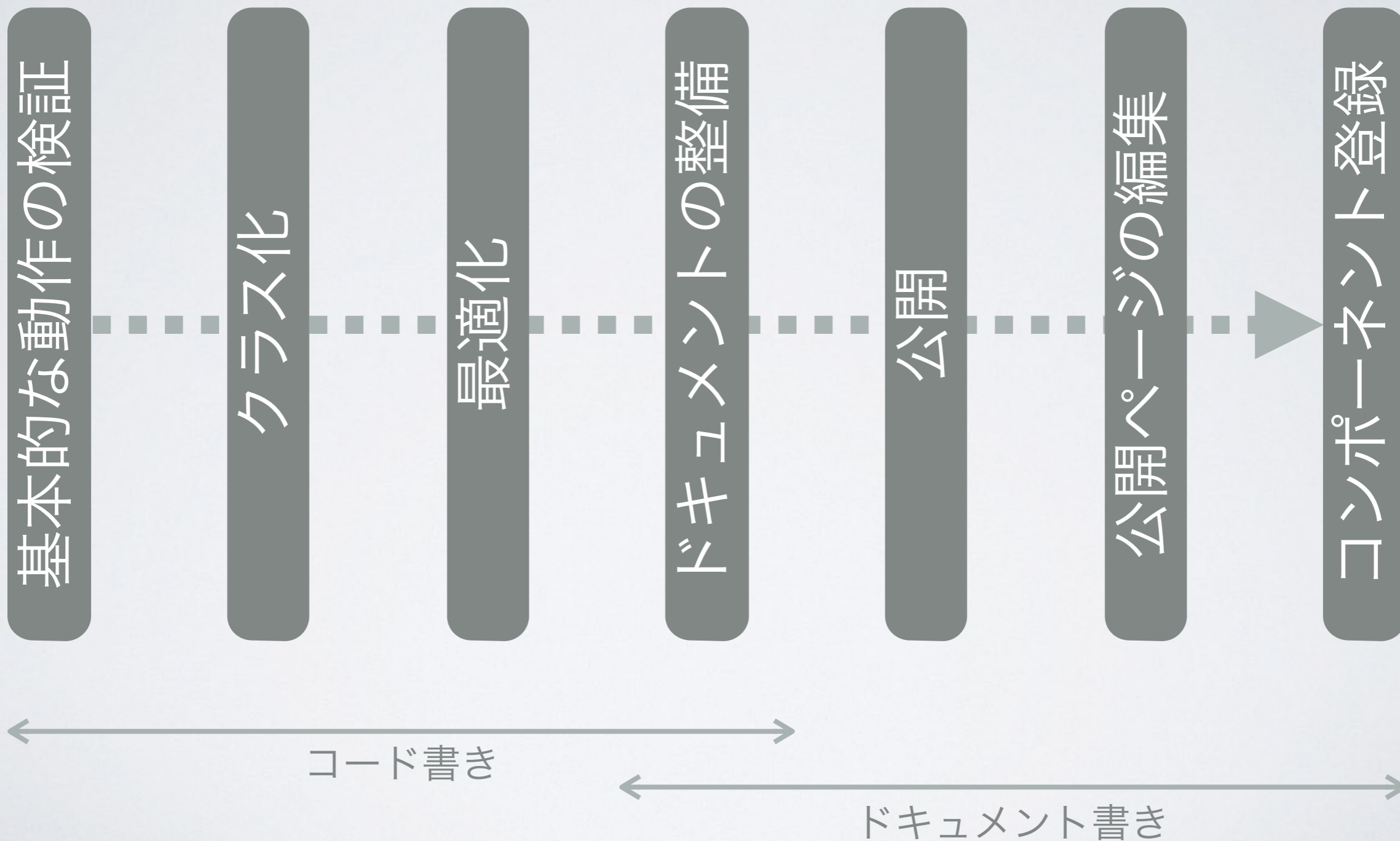


今回の話の内容は、こちらにも例があります。細かいGUIインターフェースの操作法などは、こちらをご参照ください

「ライブラリ・コンポーネントの作りかた」のページ

http://developer.mbed.org/users/okano/notebook/how_to_make_library_and_components_jp/

ライブラリ作成の流れ



公開するもの

ライブラリを使用例
サンプルコード

HelloWorld
プログラム

ライブラリ本体

ライブラリ

解説ページ

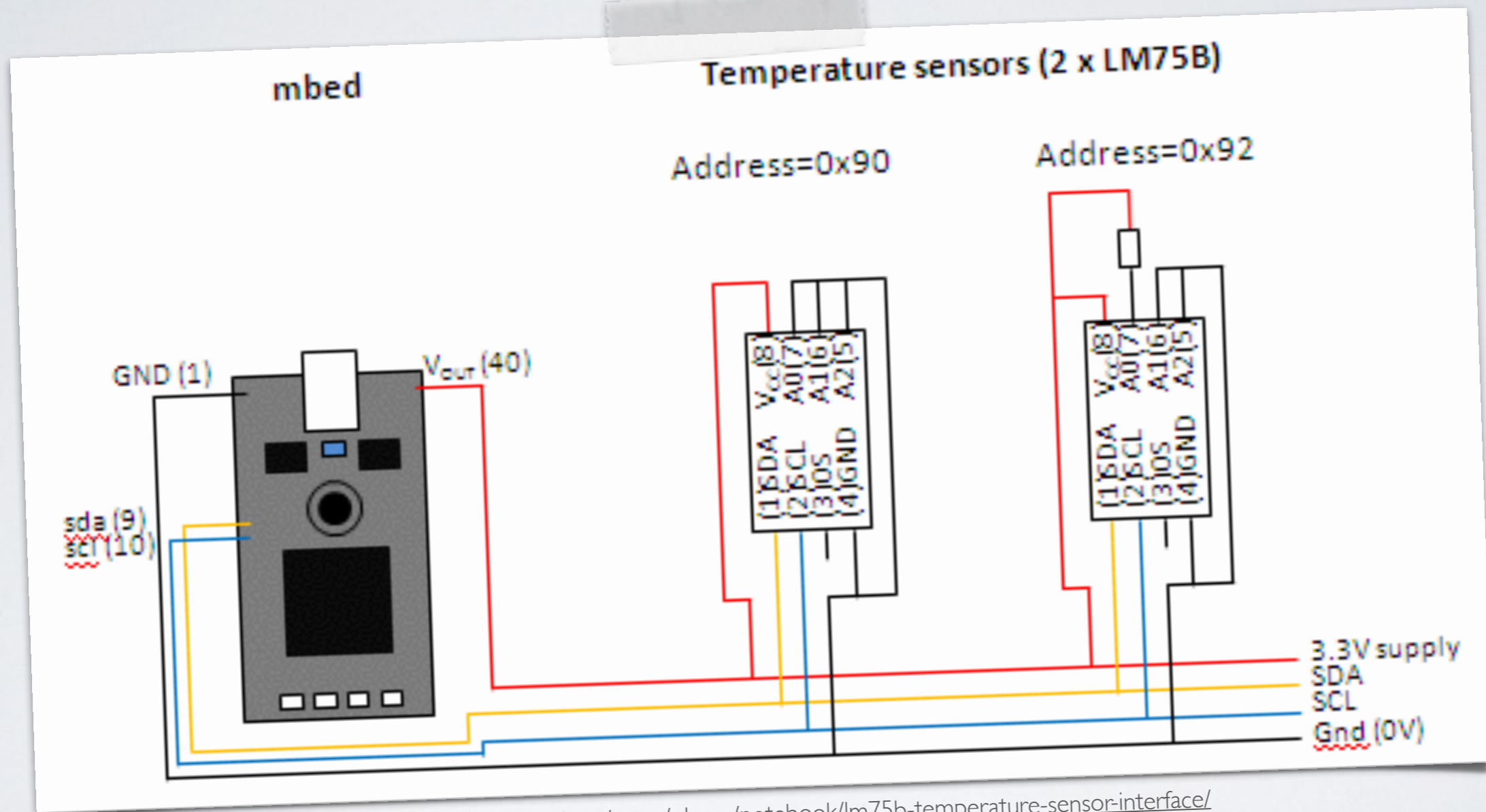
何をするライブラリか？
接続の方法
データシートへのリンクなど

たいへんそう..？

- 使ってもらいやすくするためには、いろいろ大変
- でも一度やってみれば、大したことは無い
- 必ず全部揃ってないと、いけないわけでもない
- できることまでやって、公開するのも方法
- そのあとでバージョンアップや、公開ページの記述追加も
- コードを共有して、他の人からの助けも

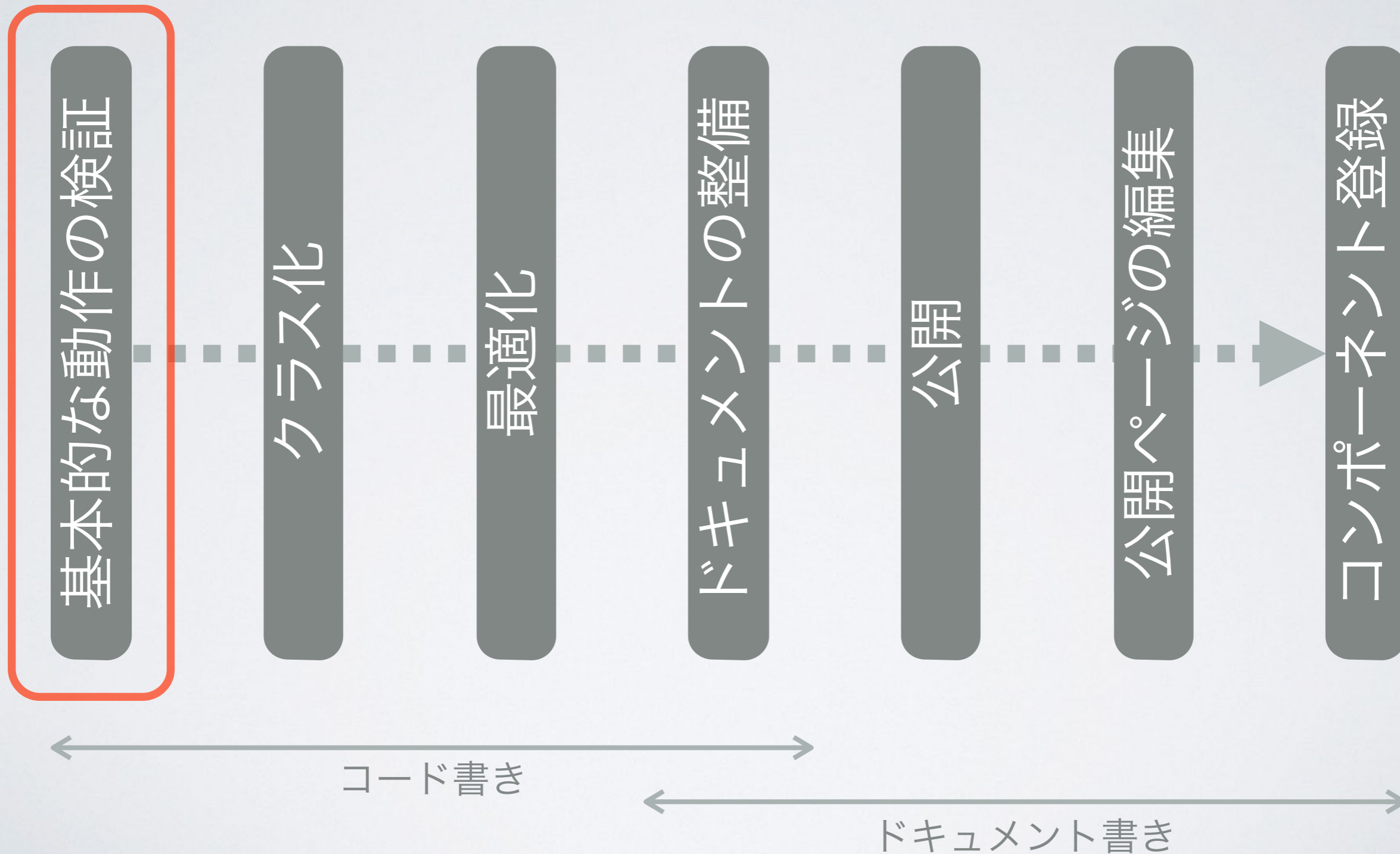
今回は簡単なハードウェアを例に説明します

例で順を追って解説



LM75B : I²Cインターフェースの温度センサ

ライブラリ作成の流れ



example project to explain how to write a class library

Dependencies: mbed test_LM75B

Home History Graph API Documentation Wiki Pull Requests Admin settings

Information

Sample code to explain the class library development. Each revisions of code in repository shows steps from hardware verification to publishing.

クラスライブラリ開発の手順を示したサンプルです。リポジトリ内のコードを追うことによって、ハードウェアの検証から公開までの流れを説明しています。

Download repository: zip gz

Repository toolbox

Import this program

Export to desktop IDE

Build repository

Send Pull Request from here

Make featured

Following

このプログラムをインポート

開発の手順の例として、コードを公開しました

これ以降のステップ・バイ・ステップで進む解説の各段階のコードを履歴から復元、試してみることができます

Edit repository homepage

Files at revision 8:d01c24c1223c

Name	Size	Actions
[up]		
main.cpp	412	Revisions Annotate
mbed.bld	65	Revisions Annotate
test_LM75B.lib	68	Revisions Annotate

Ask a question Start a discussion

Repository details

Type: Program

Created: 16 minutes ago

Imports: 0

Forks: 0

Commits: 9

Dependents: 0

Dependencies: 2

Followers: 1

This repository is Public (Unlisted).

ところで皆さん、このボタンって使われてますか？

「Commit (Commit Changes)」は現時点でのプログラムの状態 (全てのソースコード、リンクされてるライブラリ)を保存してくれるボタンです

「Revisions (Revision History)」はこれまでにコミットした履歴を表示してくれるボタン。このボタンを押してリストを表示、プログラムを選択した過去の状態に戻せたりします

「Format Code」はソースを自動的に整形してくれます。公開前には必ず押すようにしましょう

The screenshot shows the mbed IDE interface. The top toolbar contains several buttons: 'New', 'Import', 'Save', 'Save All', 'Compile', 'Commit', 'Revisions', 'Format Code', and 'Help'. The 'Commit', 'Revisions', and 'Format Code' buttons are highlighted with red circles. The 'Commit' button has a green checkmark icon, 'Revisions' has a clock icon, and 'Format Code' has a lightning bolt icon. The main workspace shows a file named 'main.cpp' with the following code:

```
1 #include "mbed.h"
2 #include "test_LM75B.h"
3
4 test_LM75B temp0( p28, p27 );
5
6 I2C i2c( p28, p27 );
7 test_LM75B temp1( i2c );
8
9
10 int main()
11 {
12     float t0;
13     float t1;
14
```

developer.mbed.org

mbed

Revision History

① プログラムを選択

② Revisionsボタンを押す

④ Switchボタンを押す

③ 切り替えたいリビジョンを選択して

コード例をインポートしたら..
↓
履歴からの復元のしかた

Program Workspace

- My Programs
- test_LM75B_Hello
- test_LM75B
- main.cpp
- mbed

Revisions of program "test_LM75B_Hello"

Showing revisions of program "test_LM75B_Hello" and public repository at [okano/test_LM75B_Hello](#).

Commit Discard Changes Compare Switch Revert Merge

Graph	Revisor	When	Who	Comment	
	<input checked="" type="radio"/>	8	21 minutes ago	okano	default tip to include published version library
	<input type="radio"/>	7	39 minutes ago	okano	comment added
	<input type="radio"/>	6	43 minutes ago	okano	two constructors version
	<input type="radio"/>	5	46 minutes ago	okano	(temporary)
	<input type="radio"/>	4	51 minutes ago	okano	chabged to referencing I2C object
	<input type="radio"/>	3	53 minutes ago	okano	modification on constructor : taking an I2C object
	<input type="radio"/>	2	58 minutes ago	okano	added : slave address change capability & operator o
	<input type="radio"/>	1	1 hour, 1 minute ago	okano	device access is made as a class
	<input checked="" type="radio"/>	0	1 hour, 1 minute ago	okano	very basic code for hardware verification

Revision 0 (f947ed831c67)

Comment: very basic code for hardw
When: 1 hour, 1 minute ago
2014-11-03 00:03:55
Files changed: 2
Lines changed: 49

Revision log

All Changes

- main.cpp
- mbed.bld

Remote changes for [okano/test_LM75B_Hello](#) Incoming: 0 Outgoing: 0

Compare With ... Publish Changes

Ready. INS


```

#include "mbed.h"

// LM75B I2C slave address
#define ADDRESS_LM75B 0x90

// LM75B registers
#define LM75B_Conf 0x01
#define LM75B_Temp 0x00
#define LM75B_Tos 0x03
#define LM75B_Thyst 0x02

I2C i2c( p28, p27 );

void init( void );
float read_temp( void );

int main()
{
    init();

    while(1) {
        printf( "temp = %7.3f\r\n", read_temp() );
        wait( 1 );
    }
}

void init( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Conf;
    command[ 1 ] = 0x00;

    i2c.write( ADDRESS_LM75B, command, 2 );
}

float read_temp( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Temp;

    i2c.write( ADDRESS_LM75B, command, 1 ); // Send command string
    i2c.read( ADDRESS_LM75B, command, 2 ); // read two bytes data

    return ( (float)( (command[ 0 ] << 8) | command[1] ) / 256.0 );
}

```

ハードウェアの動作を確認したコード

初期化

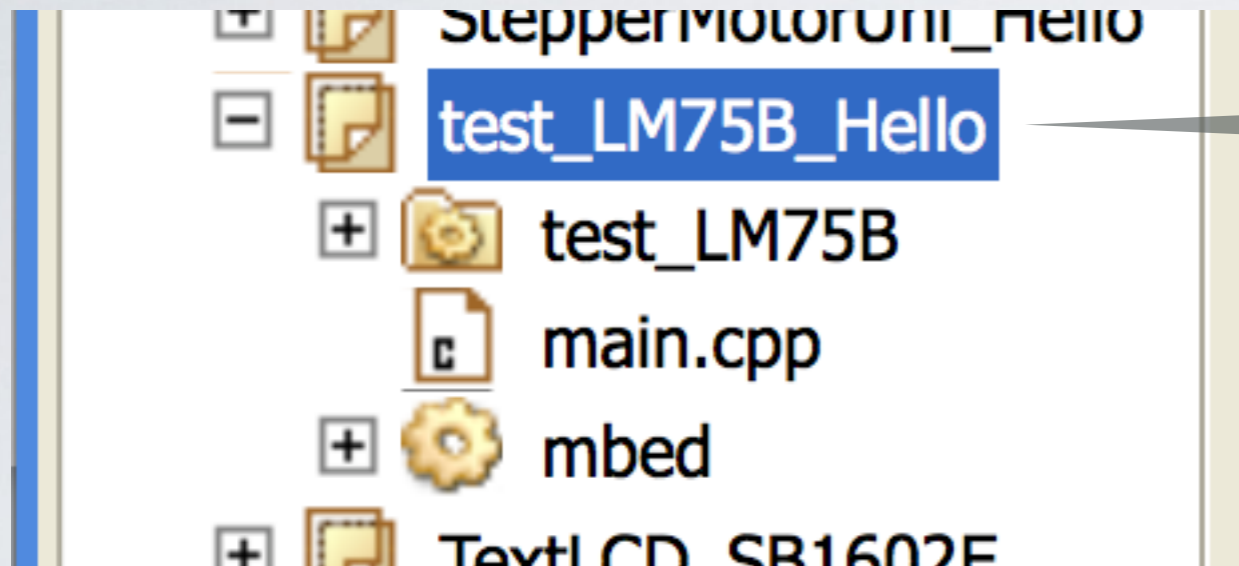
read_temp()を呼ぶと温度の値(摂氏)が返されるので、それを標準出力へ

スレーブアドレス0x90のデバイスに初期化レジスタ0x01に0x00を書く

スレーブアドレス0x90のデバイスのレジスタ0x00から2バイト読み出す

読み出したデータを摂氏に直して返す

プログラム名は「○○○_Hello」(○○○はライブラリ名、ライブラリはデバイスの型番など)のようしておくのが、わかりやすいと思います

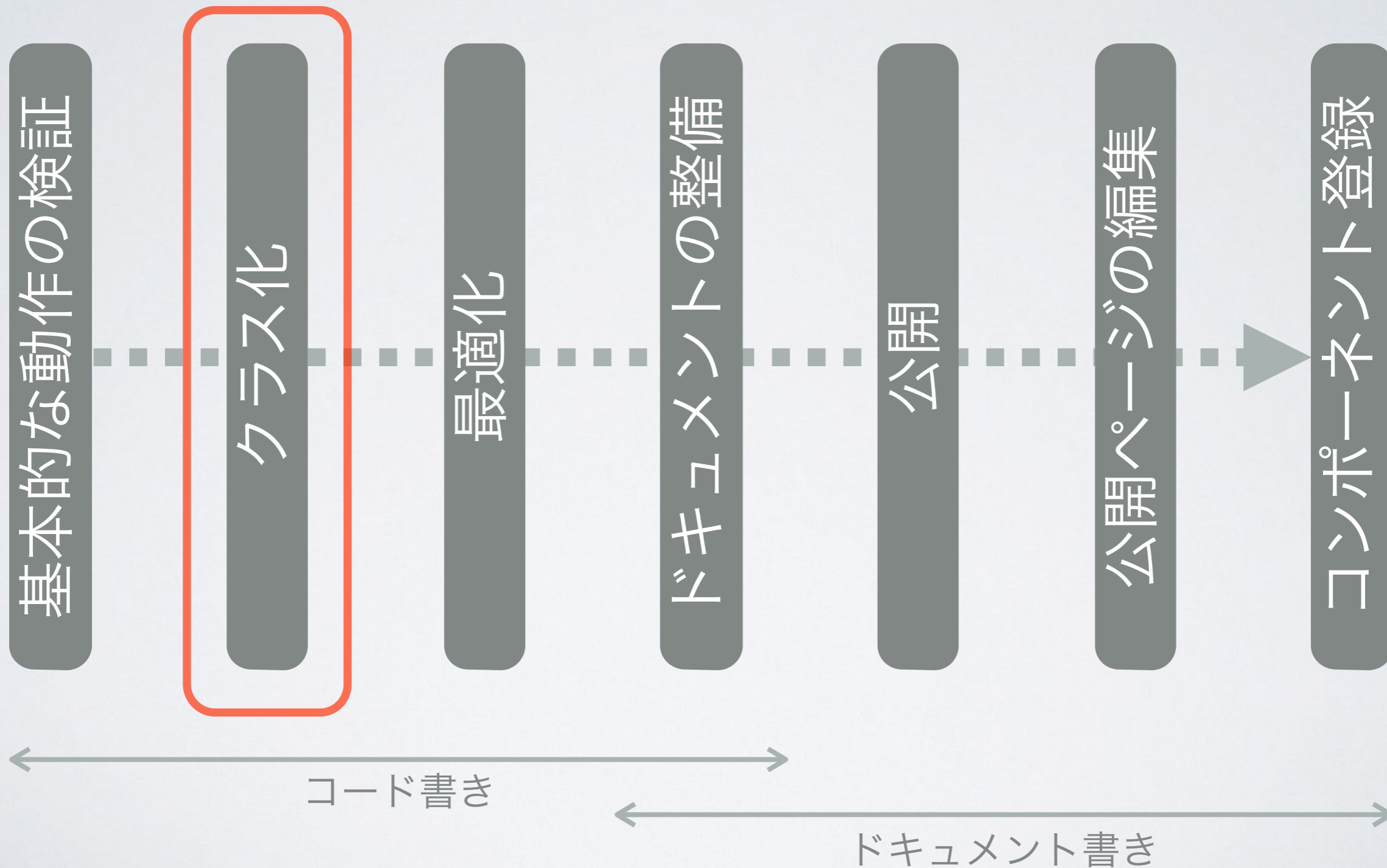


この例では、後で「test_LM75B」という名のライブラリを作るので「test_LM75B_Hello」というプログラム名にしました

プログラムもライブラリもいつでも変更することは可能です。
公開名は公開時点で固定されるので、それまでに調整すれば問題ありません。

今回のライブラリは既に公開されているLM75Bのライブラリと重複しないように「test_LM75B」としました。(公開者が違えば同じ名前で公開することは可能です)

ライブラリ作成の流れ

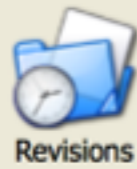


New Import Save Save All Compile Commit Revisions Help

Program Workspace

- My Programs
- test_LM75B_Hello
 - test_LM75B
 - main.cpp
 - mbed

Revision History



Revisions of program "test_LM75B_Hello"

Showing revisions of program "test_LM75B_Hello" and public repository at okano/test_LM75B_Hello.

Commit Discard Changes Compare Switch Revert Merge

Graph	Revisor	When	Who	Comment	
	<input checked="" type="radio"/>	8	21 minutes ago	okano	default tip to include published version library
	<input type="radio"/>	7	39 minutes ago	okano	comment added
	<input type="radio"/>	6	43 minutes ago	okano	two constructors version
	<input type="radio"/>	5	46 minutes ago	okano	(temporary)
	<input type="radio"/>	4	51 minutes ago	okano	chabged to referencing I2C object
	<input type="radio"/>	3	53 minutes ago	okano	modification on constructor : taking an I2C object
	<input type="radio"/>	2	58 minutes ago	okano	added : slave address change capability & operator o
	<input checked="" type="radio"/>	1	1 hour, 1 minute ago	okano	device access is made as a class
	<input type="radio"/>	0	1 hour, 1 minute ago	okano	very basic code for hardware verification

Revision 1 (3c29c04cf2)

Comment device access is made as .
 When 1 hour, 1 minute ago
 Date 2014-11-03 00:06:59
 Files changed 2
 Lines changed 42

Revision log

All Changes

- main.cpp
- test_LM75B.lib

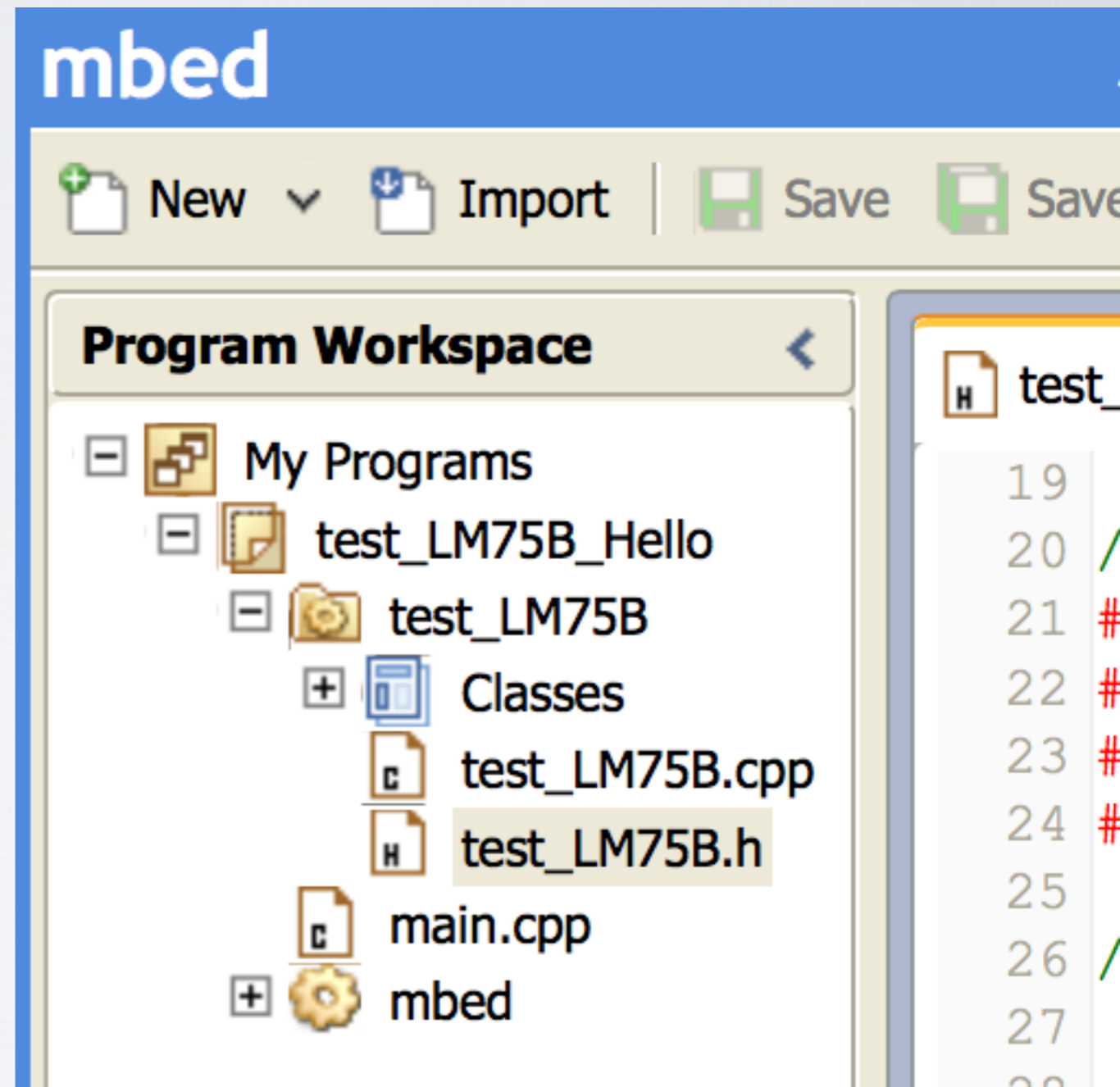
Remote changes for okano/test_LM75B_Hello Incoming: 0 Outgoing: 0

Update Update From... Compare With ... Publish Changes

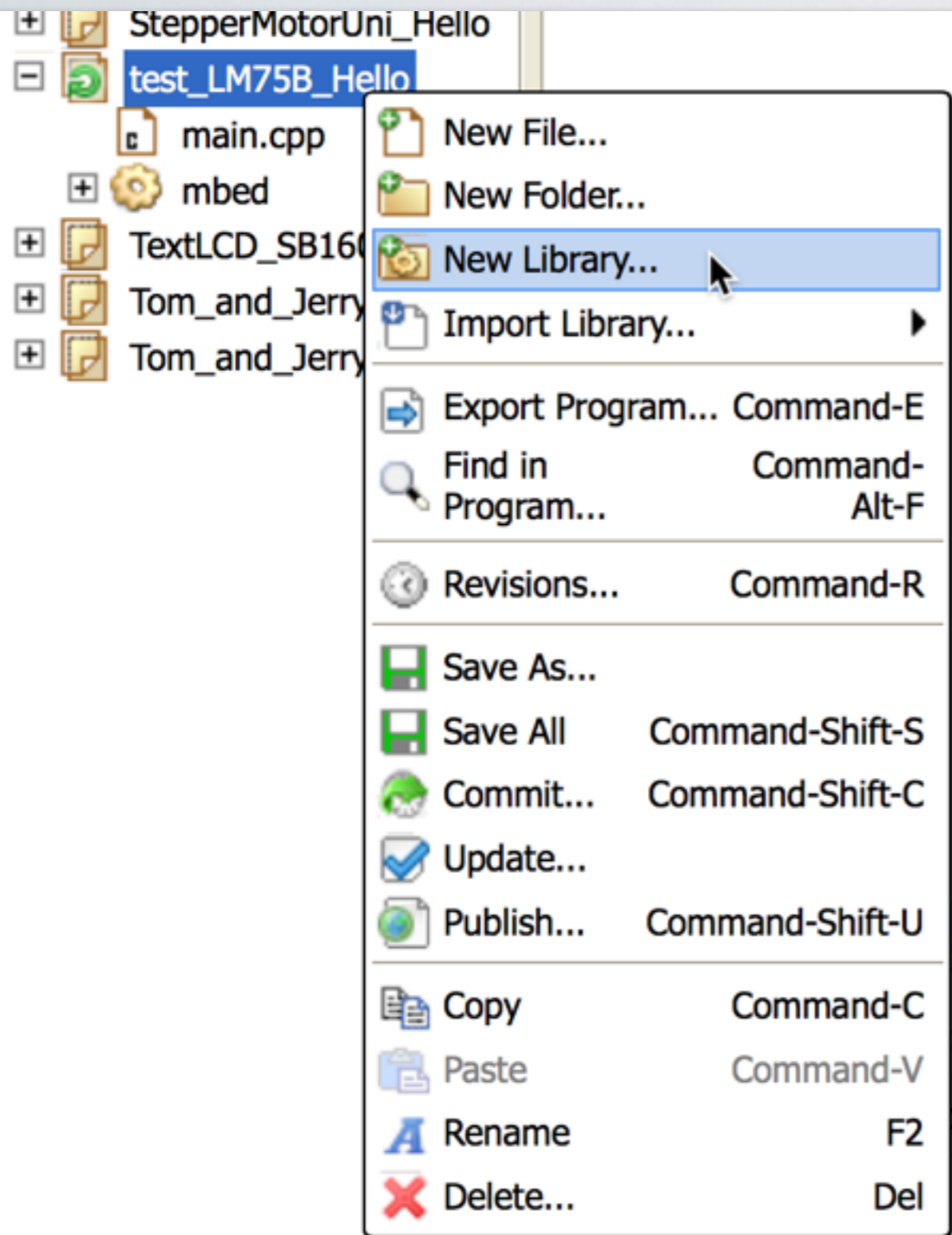
Graph	Revisor	When	Who	Comment
-------	---------	------	-----	---------

クラス化

- ライブラリ・フォルダを作って、デバイス依存コードとアプリケーションを分離
- クラスは「.cpp」と「.h」のファイルに



ちなみに..



• 各プログラムやフォルダ, ライブラリの操作は右クリックで

• 新規追加 / 消去

• Export(書き出し)

• 公開

• コピー / ペースト

• 名称変更

などができます

新main.cpp

```
#include "mbed.h"

// LM75B I2C slave address
#define ADDRESS_LM75B 0x90

// LM75B registers
#define LM75B_Conf 0x01
#define LM75B_Temp 0x00
#define LM75B_Tos 0x03
#define LM75B_Thyst 0x02
```

main.cpp

```
I2C i2c( p28, p27 );

void init( void );
float read_temp( void );

int main()
{
    init();

    while(1) {
        printf( "temp = %7.3f\r\n", read_temp() );
        wait( 1 );
    }
}
```

```
void init( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Conf;
    command[ 1 ] = 0x00;

    i2c.write( ADDRESS_LM75B, command, 2 );
}

float read_temp( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Temp;

    i2c.write( ADDRESS_LM75B, command, 1 ); // Send command string
    i2c.read( ADDRESS_LM75B, command, 2 ); // read two bytes data

    return ( (float)( (command[ 0 ] << 8) | command[1] ) / 256.0 );
}
```

```
#include "mbed.h"
#include "test_LM75B.h"

test_LM75B temp( p28, p27 );

int main()
{
    while(1) {
        printf( "temp = %7.3f\r\n", temp.read() );
        wait( 1 );
    }
}
```

```
#include "mbed.h"
// LM75B I2C slave address
#define ADDRESS_LM75B 0x90

// LM75B registers
#define LM75B_Conf 0x01
#define LM75B_Temp 0x00
#define LM75B_Tos 0x03
#define LM75B_Thyst 0x02

class test_LM75B
{
public:
    test_LM75B( PinName sda, PinName scl );
    ~test_LM75B();
    void init( void );
    float read( void );
private:
    I2C i2c;
};
```

```
#include "test_LM75B.h"

test_LM75B::test_LM75B( PinName sda, PinName scl ) : i2c( sda, scl )
{
    init();
}

test_LM75B::~~test_LM75B()
{
}

void test_LM75B::init( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Conf;
    command[ 1 ] = 0x00;

    i2c.write( ADDRESS_LM75B, command, 2 );
}

float test_LM75B::read( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Temp;

    i2c.write( ADDRESS_LM75B, command, 1 ); // Send command string
    i2c.read( ADDRESS_LM75B, command, 2 ); // read two bytes data

    return ( (float)( (command[ 0 ] << 8) | command[1] ) / 256.0 );
}
```

test_LM75B.cpp

- test_LM75B>Hello
- test_LM75B
- Classes
- test_LM75B.cpp
- test_LM75B.h
- main.cpp
- mbed

- test_LM75B>Hello
- main.cpp
- mbed

```

#include "mbed.h"
// LM75B I2C slave address
#define ADDRESS_LM75B 0x90

// LM75B registers
#define LM75B_Conf 0x01
#define LM75B_Temp 0x00
#define LM75B_Tos 0x03
#define LM75B_Thyst 0x02

class test_LM75B
{
public:
    test_LM75B( PinName sda, PinName scl );
    ~test_LM75B();
    void init( void );
    float read( void );
private:
    I2C i2c;
};

```

test_LM75B.h

定数の宣言

関数のプロトタイプ

インスタンス変数

I²C

test_LM75B.cpp

```

#include "test_LM75B.h"

test_LM75B::test_LM75B( PinName sda, PinName scl ) : i2c( sda, scl )
{
    init();
}

test_LM75B::~test_LM75B()
{
}

void test_LM75B::init( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Conf;
    command[ 1 ] = 0x00;

    i2c.write( ADDRESS_LM75B, command, 2 );
}

float test_LM75B::read( void )
{
    char command[ 2 ];

    command[ 0 ] = LM75B_Temp;

    i2c.write( ADDRESS_LM75B, command, 1 ); // Send command string
    i2c.read( ADDRESS_LM75B, command, 2 ); // read two bytes data

    return ( (float)( (command[ 0 ] << 8) | command[1] ) / 256.0 );
}

```

コンストラクタ

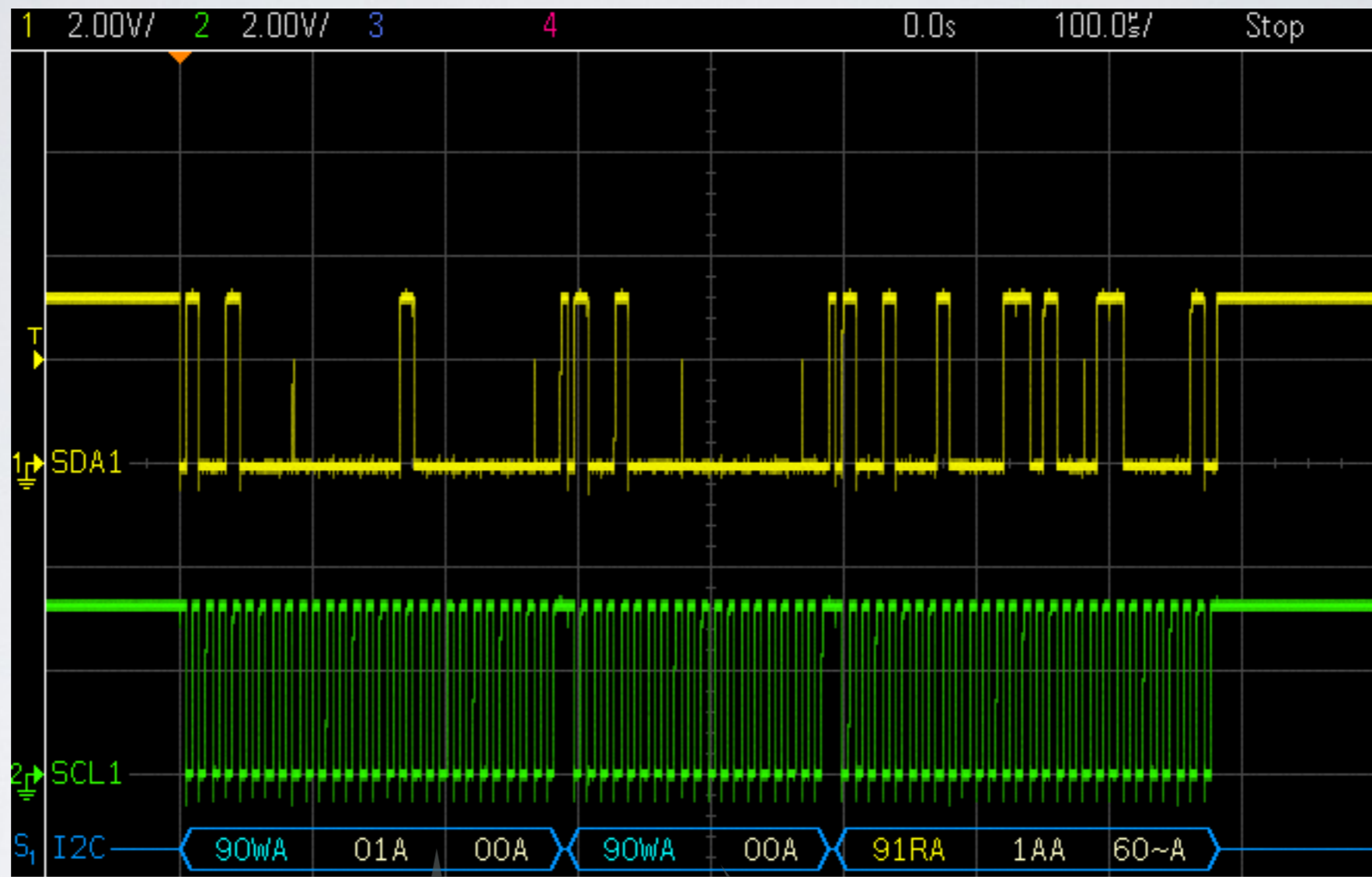
クラス変数の初期化は「初期化リスト」で

デストラクタ

メンバ関数：初期化

メンバ関数：デバイス読み出し

ちなみにこのような書き方をすると..

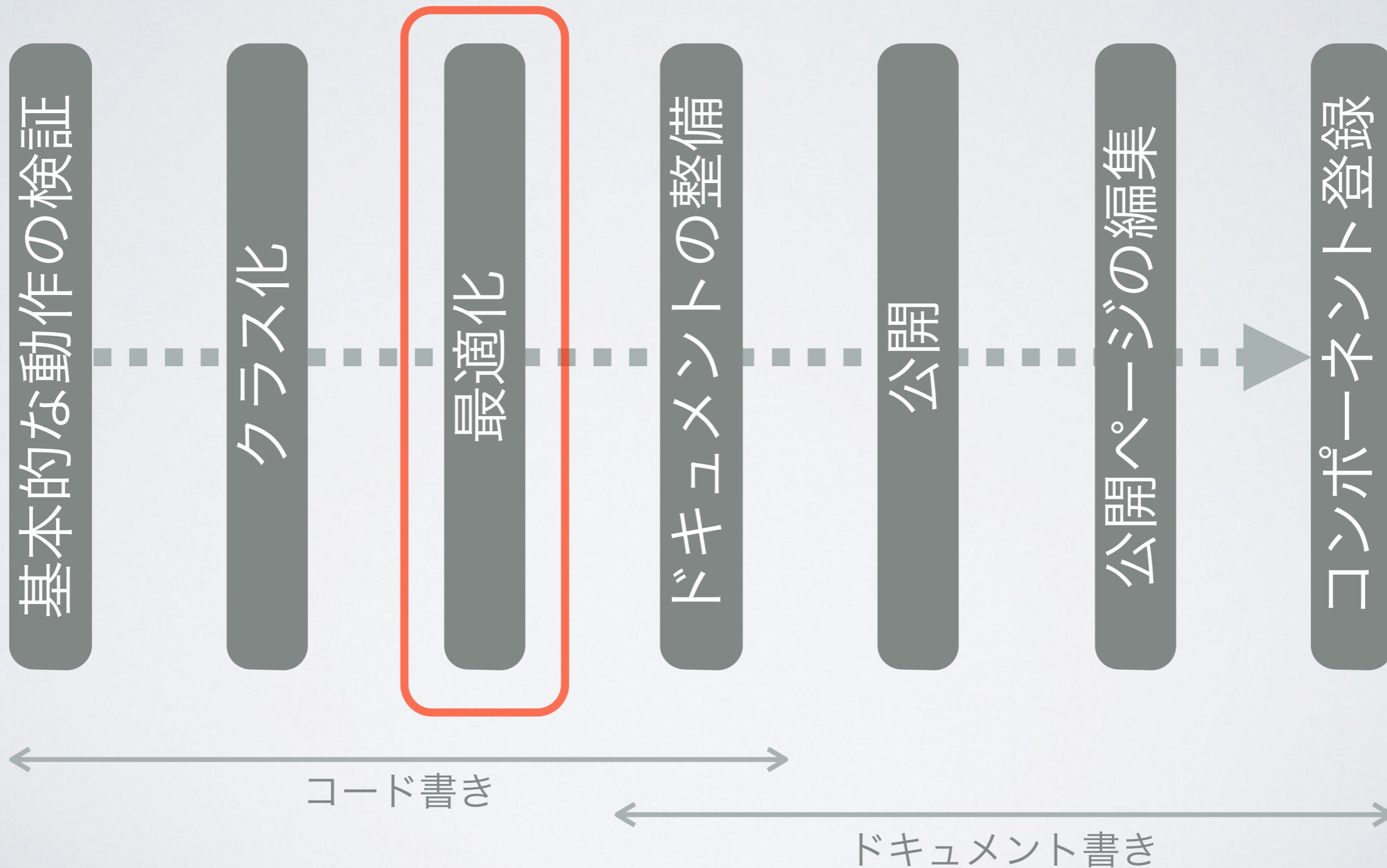


コンストラクタはmain()関数実行前に呼ばれます。
なので

main()関数内で実行される、
センサ読み出しのためのアクセス

メイン関数の実行はここから

ライブラリ作成の流れ



mbed Revision History

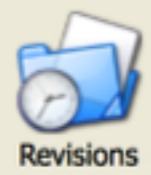
mbed LPC1768

New Import Save Save All Compile Commit Revisions Help

Program Workspace

- My Programs
 - test_LM75B_Hello
 - test_LM75B
 - main.cpp
 - mbed

Revision History



Revisions of program "test_LM75B_Hello"

Showing revisions of program "test_LM75B_Hello" and public repository at [okano/test_LM75B_Hello](#).

Commit Discard Changes Compare Switch Revert Merge

Graph	Revisor	When	Who	Comment
	<input checked="" type="radio"/>	8	21 minutes ago	okano default tip to include published version library
	<input type="radio"/>	7	39 minutes ago	okano comment added
	<input type="radio"/>	6	43 minutes ago	okano two constructors version
	<input type="radio"/>	5	46 minutes ago	okano (temporary)
	<input type="radio"/>	4	51 minutes ago	okano chabged to referencing I2C object
	<input type="radio"/>	3	53 minutes ago	okano modification on constructor : taking an I2C object
	<input checked="" type="radio"/>	2	58 minutes ago	okano added : slave address change capability & operator o
	<input type="radio"/>	1	1 hour, 1 minute ago	okano device access is made as a class
	<input type="radio"/>	0	1 hour, 1 minute ago	okano very basic code for hardware verification

Revision 2 (482581f76a1d)

Comment added : slave address cha
 When 58 minutes ago
 Date 2014-11-03 00:11:02
 Files changed 2
 Lines changed 7

Revision log

All Changes

main.cpp	<input checked="" type="checkbox"/>
test_LM75B.lib	<input checked="" type="checkbox"/>

Remote changes for [okano/test_LM75B_Hello](#) Incoming: 0 Outgoing: 0

Update Update From... Compare With ... Publish Changes

Graph Revisor When Who Comment

```
#include "mbed.h"

// LM75B I2C slave address
#define ADDRESS_LM75B 0x90

// LM75B registers
#define LM75B_Conf 0x01
#define LM75B_Temp 0x00
#define LM75B_Tos 0x03
#define LM75B_Thyst 0x02

class test_LM75B
{
public:
    test_LM75B( PinName sda, PinName scl, char address = ADDRESS_LM75B );
    ~test_LM75B();
    void init( void );
    float read( void );
    operator float( void );
private:
    I2C i2c;
    char adr;
};
```

3番目の引数が与えられなかった場合は
デフォルトの値として0x90を使う

```
.....
test_LM75B::test_LM75B( PinName sda, PinName scl, char address ) : i2c( sda, scl ), adr( address )
{
    init();
}
.....
```

```
#include "mbed.h"
#include "test_LM75B.h"

test_LM75B temp0 ( p28, p27, 0x90 );
test_LM75B temp1 ( p28, p27, 0x92 );

int main()
{
    while(1) {
        printf( "temp0 = %7.3f\r\n", temp0.read() );
        printf( "temp1 = %7.3f\r\n", temp1.read() );
        wait( 1 );
    }
}
```

スレーブアドレス=0x90のセンサのインスタンス

スレーブアドレス=0x92のセンサのインスタンス

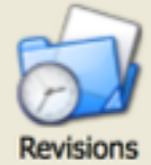
前ページのようなクラスにしておくと、
このような使い方が可能になります

New Import Save Save All Compile Commit Revisions Help

Program Workspace

- My Programs
 - test_LM75B_Hello
 - test_LM75B
 - main.cpp
 - mbed

Revision History



Revisions of program "test_LM75B_Hello"

Showing revisions of program "test_LM75B_Hello" and public repository at okano/test_LM75B_Hello.

Commit Discard Changes Compare Switch Revert Merge

Graph	Revisor	When	Who	Comment
	<input checked="" type="radio"/>	8	21 minutes ago	okano default tip to include published version library
	<input type="radio"/>	7	39 minutes ago	okano comment added
	<input type="radio"/>	6	43 minutes ago	okano two constructors version
	<input type="radio"/>	5	46 minutes ago	okano (temporary)
	<input type="radio"/>	4	51 minutes ago	okano chabged to referencing I2C object
	<input checked="" type="radio"/>	3	53 minutes ago	okano modification on constructor : taking an I2C object
	<input type="radio"/>	2	58 minutes ago	okano added : slave address change capability & operator o
	<input type="radio"/>	1	1 hour, 1 minute ago	okano device access is made as a class
	<input type="radio"/>	0	1 hour, 1 minute ago	okano very basic code for hardware verification

Revision 3 (863cf79582a4)

Comment modification on constructr
 When 53 minutes ago
 Date 2014-11-03 00:15:27
 Files changed 2
 Lines changed 7

Revision log

All Changes

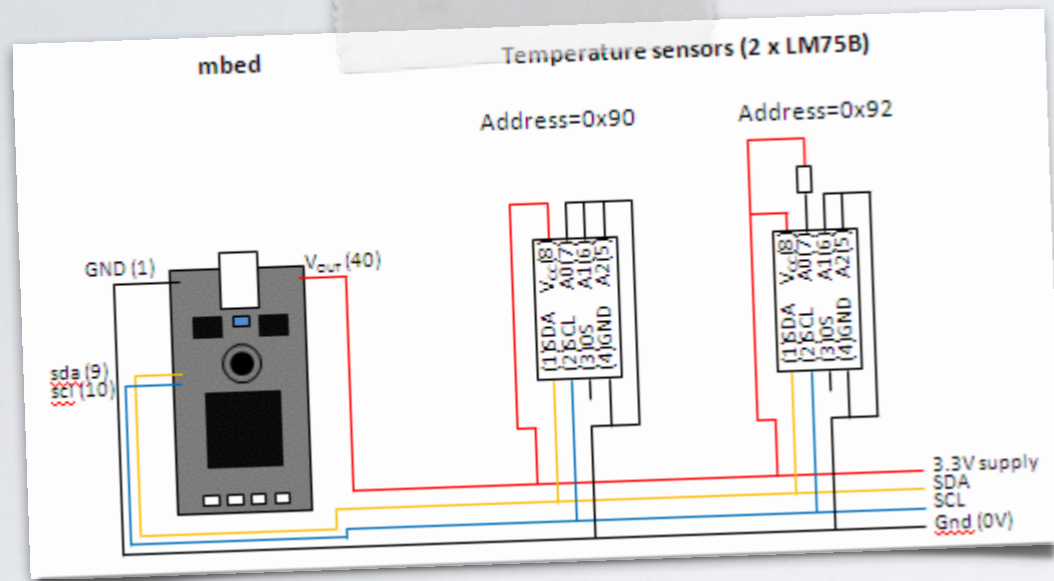
main.cpp	
test_LM75B.lib	

Remote changes for okano/test_LM75B_Hello Incoming: 0 Outgoing: 0

Update Update From... Compare With ... Publish Changes

Graph Revisor When Who Comment

素朴な疑問



Q 同時に2つのインスタンスを宣言すると各温度センサのインスタンス内にそれぞれ別のI2Cクラスのインスタンスが作られてしまう←これは問題ないか？

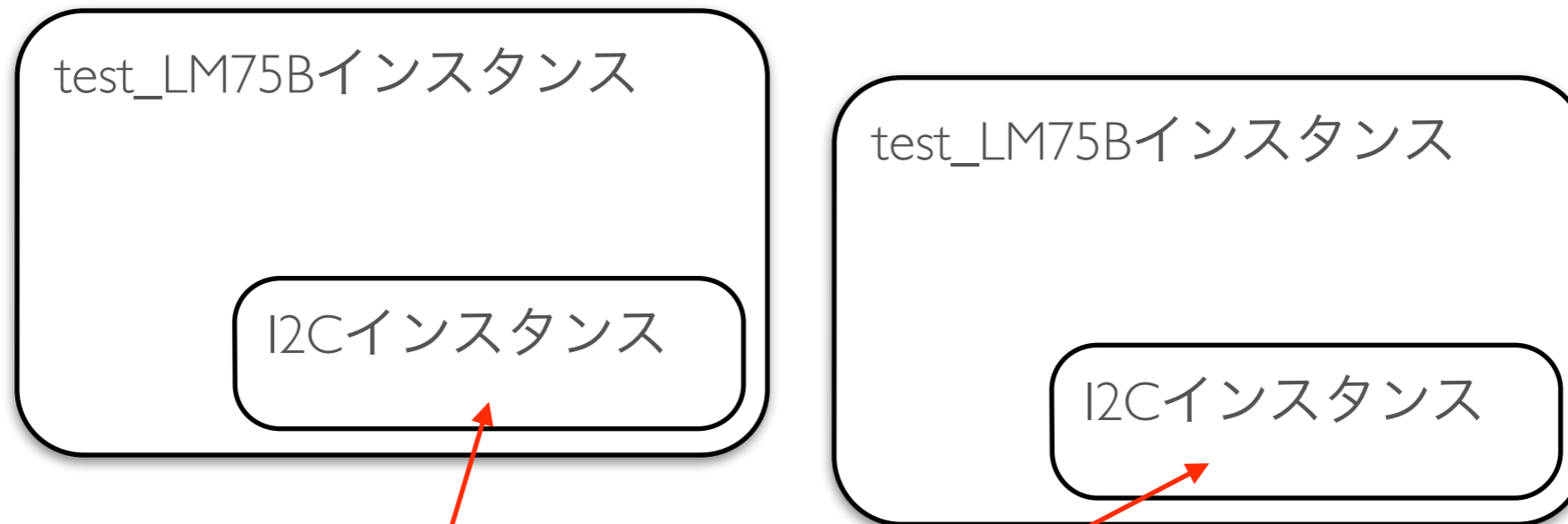
A 問題ありません。
mbed SDKは同一インターフェースに複数のインスタンスを持たせることができるように作られています。

```
#include "mbed.h"
#include "test_LM75B.h"

test_LM75B temp0( p28, p27, 0x90 );
test_LM75B temp1( p28, p27, 0x92 );

int main()
{
    while(1) {
        printf( "temp0 = %7.3f\r\n", temp0.read() );
        printf( "temp1 = %7.3f\r\n", temp1.read() );
        wait( 1 );
    }
}
```

プログラム

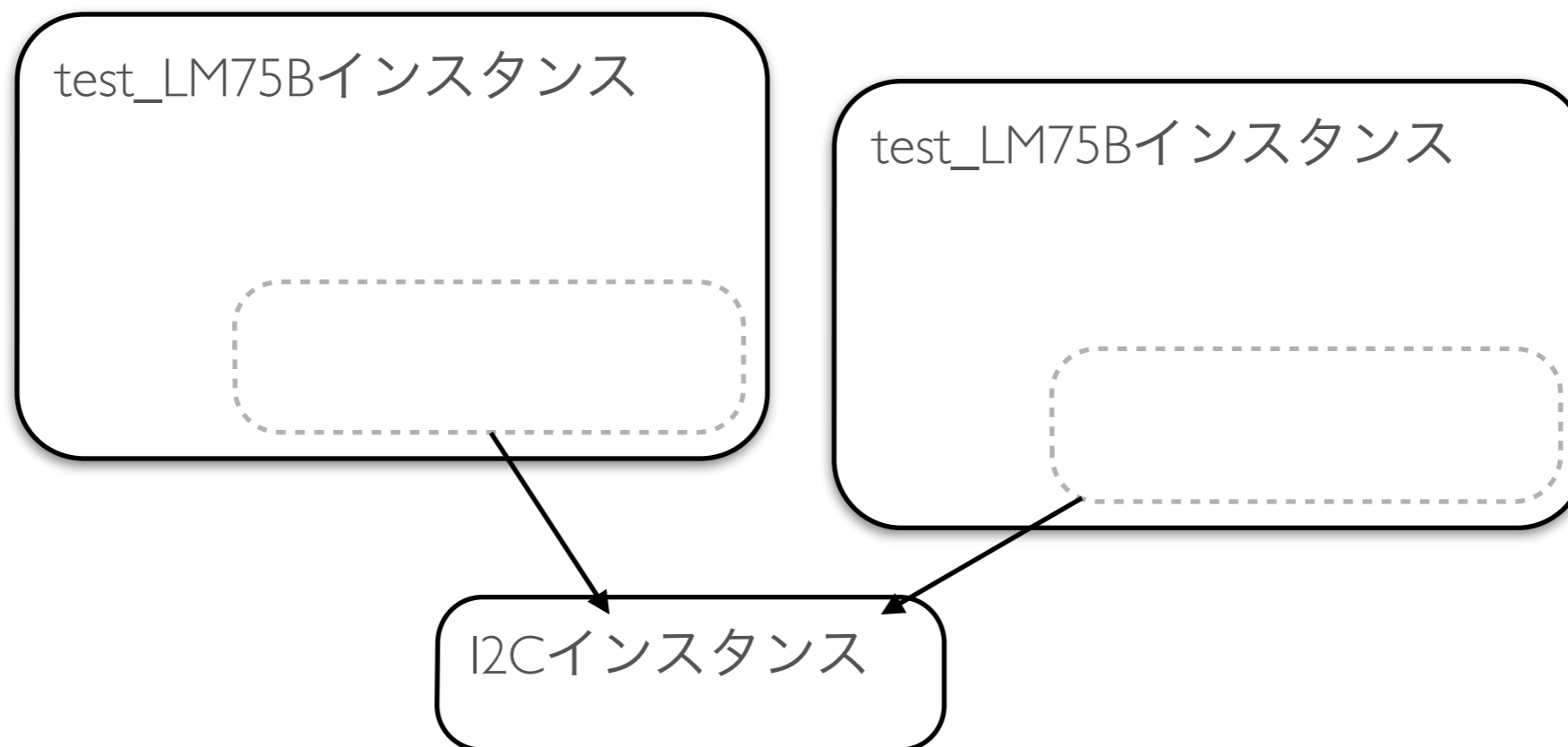


同じハードウェアを指す個別のインスタンスを持つことができる。
それぞれのインスタンスに別の設定をしておいて、自動的に切り替えながら使うなど

この機能の具体的な活用例は、後ほど

たとえば一方のデバイスに400kHz、もう一方のデバイスに100kHz
のクロック周波数を設定し、これを自動的に切り替えて使うなど
(実際にこのような使い方をすることは無いと思いますが..)

プログラム



こういうことも可能です。
test_LM75Bの中に自前のI2Cインスタンスを持つのではなく、外部の1個のインスタンスを使う

たくさんの

あとからクロック周波数を調整するような場合に便利

```

#include "mbed.h"
#include "test_LM75B.h"

test_LM75B tmp[] = {
    test_LM75B( p28, p27, 0x90 ),
    test_LM75B( p28, p27, 0x92 ),
    test_LM75B( p28, p27, 0x93 ),
    test_LM75B( p28, p27, 0x94 ),
    test_LM75B( p28, p27, 0x96 ),
    test_LM75B( p28, p27, 0x98 ),
    test_LM75B( p28, p27, 0x9A ),
    test_LM75B( p28, p27, 0x9C )
};

int main()
{
    for ( int i = 0; i < 4; i++ ) {
        printf( "temp = %7.3f\r\n",
            (float)(tmp[ i ]) );
    }
    wait( 1 );
}

```

```

#include "mbed.h"
#include "test_LM75B.h"

I2C          i2c( p28, p27 );

test_LM75B tmp[] = {
    test_LM75B( i2c, 0x90 ),
    test_LM75B( i2c, 0x92 ),
    test_LM75B( i2c, 0x93 ),
    test_LM75B( i2c, 0x94 ),
    test_LM75B( i2c, 0x96 ),
    test_LM75B( i2c, 0x98 ),
    test_LM75B( i2c, 0x9A ),
    test_LM75B( i2c, 0x9C )
};

int main()
{
    i2c.frequency( 10 * 1000 );

    while(1) {
        for ( int i = 0; i < 4; i++ ) {
            printf( "temp = %7.3f\r\n",
                (float)(tmp[ i ]) );
        }
        wait( 1 );
    }
}

```

たとえば..

I²Cを長く引き回すような場合、わざとクロック周波数を落として通信を行うことがある。これを行うのにクラス内部に触ること無く、外部から変更を加える事はできないか？

```
#include "mbed.h"
#include "test_LM75B.h"

I2C          i2c( p28, p27 );
test_LM75B    temp( i2c );

int main()
{
    float    t;

    i2c.frequency( 400 * 1000 );

    while(1) {
        t    = temp;
        printf( "temp = %7.3f\r\n", t );
        wait( 1 );
    }
}
```

main.cpp

```

I2C          i2c( p28, p27 );
test_LM75B    temp( i2c );

int main()
{
    float    t;

    i2c.frequency( 400 * 1000 );

    while(1) {
        t    = temp;
        printf( "temp = %7.3f\r\n", t );
        wait( 1 );
    }
}

```

main.cpp

```

class test_LM75B
{
public:
    test_LM75B( I2C i2c_obj, char address = ADDRESS_LM75B );
    ~test_LM75B();
    void    init( void );
    float   read( void );
    operator float( void );
private:
    I2C     i2c;
    char     adr;
};

```

```

#include "test_LM75B.h"

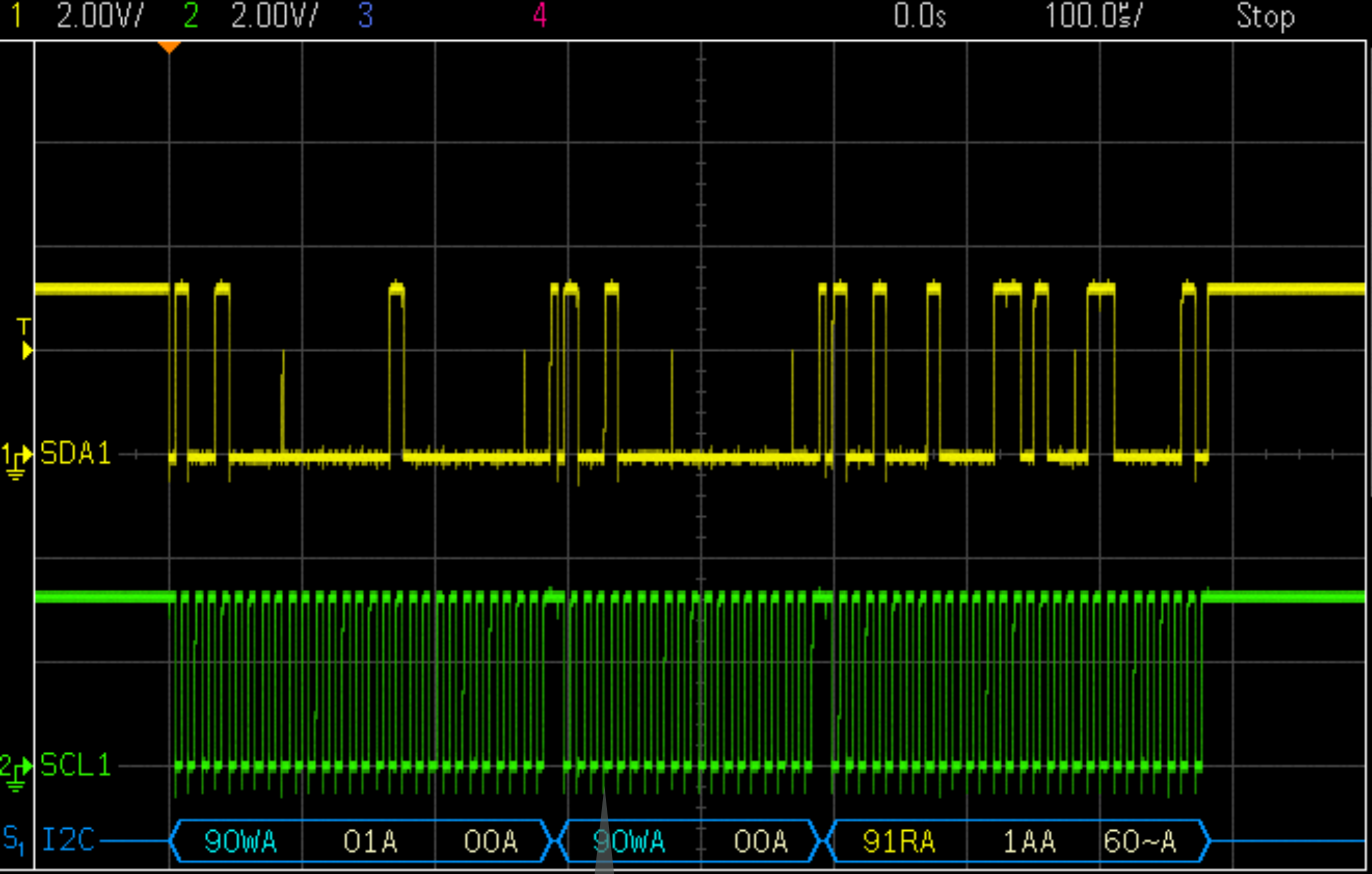
```

```

test_LM75B::test_LM75B( I2C i2c_obj, char address )
    : i2c( i2c_obj ), adr( address )
{
    init();
}

```

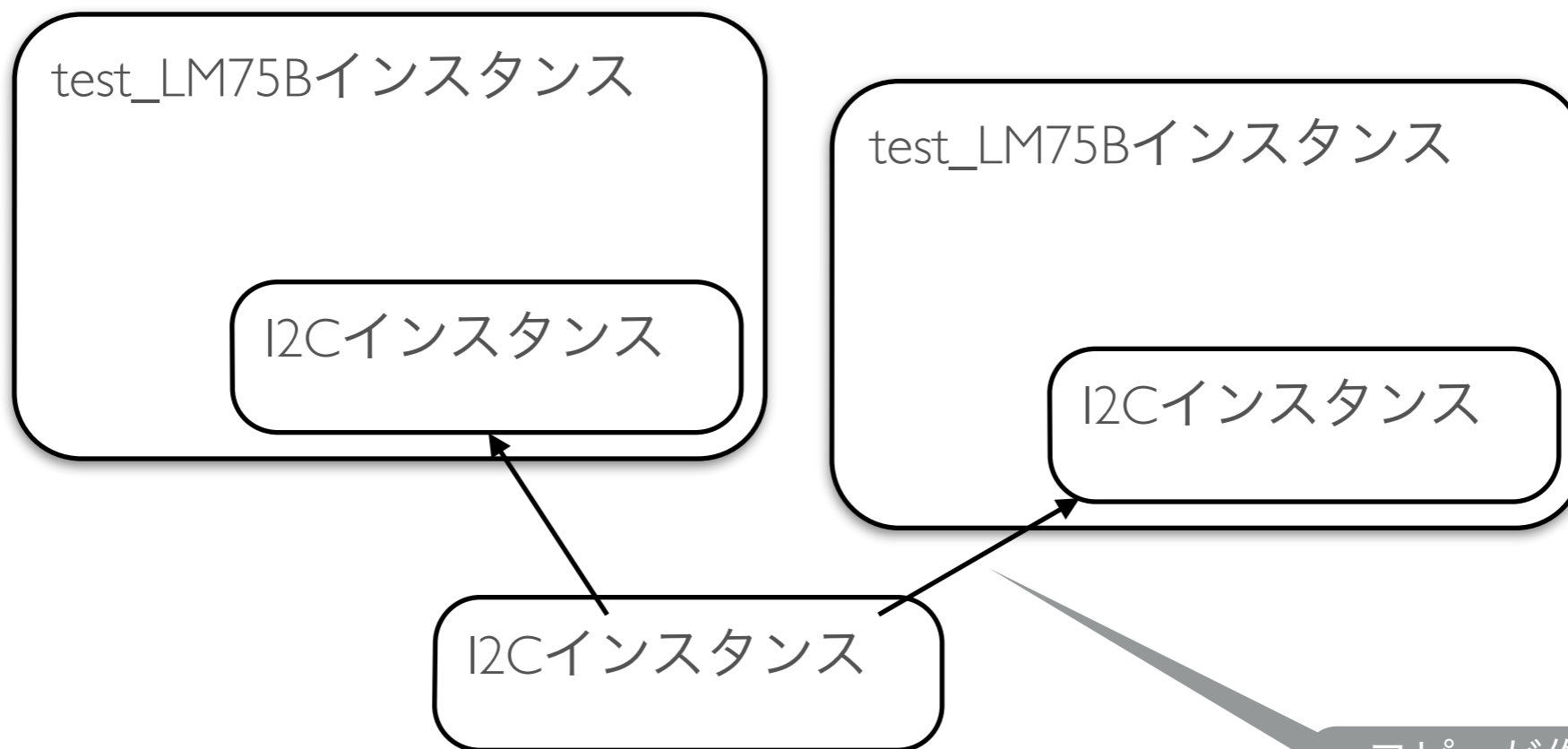
test_LM75B.cpp



ダメでした orz

センサ読み出し前に400kHzにしたはずなのに、100kHzのまま

プログラム



コピーが作られてしまった

こうなっていました orz

```

I2C          i2c( p28, p27 );
test_LM75B   temp( i2c );

int main()
{
    float    t;

    i2c.frequency( 400 * 1000 );

    while(1) {
        t    = temp;
        printf( "temp = %7.3f\r\n", t );
        wait( 1 );
    }
}

```

main.cpp

```

class test_LM75B
{
public:
    test_LM75B( I2C i2c_obj, char address = ADDRESS_LM75B );
    ~test_LM75B();
    void    init( void );
    float   read( void );
    operator float( void );
private:
    I2C     i2c;
    char    adr;
};

```

```

#include "test_LM75B.h"

```

```

test_LM75B::test_LM75B( I2C i2c_obj, char address )
    : i2c( i2c_obj ), adr( address )
{
    init();
}

```

test_LM75B.cpp

残念ながらこれはうまく動きませんでした。
test_LM75Bインスタンス内にはI2Cインスタンスが
宣言されているため、オブジェクトが
コピーされてしまいます

これを大中さんに教えていただきました m(_ _)m

New Import Save Save All Compile Commit Revisions Help

Program Workspace

- My Programs
- test_LM75B_Hello
 - test_LM75B
 - main.cpp
 - mbed

Revision History



Revisions of program "test_LM75B_Hello"

Showing revisions of program "test_LM75B_Hello" and public repository at okano/test_LM75B_Hello.

Commit Discard Changes Compare Switch Revert Merge

Graph	Revisor	When	Who	Comment	
	<input checked="" type="radio"/>	8	21 minutes ago	okano	default tip to include published version library
	<input type="radio"/>	7	39 minutes ago	okano	comment added
	<input type="radio"/>	6	43 minutes ago	okano	two constructors version
	<input type="radio"/>	5	46 minutes ago	okano	(temporary)
	<input checked="" type="radio"/>	4	51 minutes ago	okano	chabged to referencing I2C object
	<input type="radio"/>	3	53 minutes ago	okano	modification on constructor : taking an I2C object
	<input type="radio"/>	2	58 minutes ago	okano	added : slave address change capability & operator o
	<input type="radio"/>	1	1 hour, 1 minute ago	okano	device access is made as a class
	<input type="radio"/>	0	1 hour, 1 minute ago	okano	very basic code for hardware verification

Revision 4 (e79412c7b599)

Comment chabged to referencing I2

When 51 minutes ago

Date 2014-11-03 00:17:33

Files changed 1

Lines changed 2

Revision log

All Changes

test_LM75B.lib

Remote changes for okano/test_LM75B_Hello

Incoming: 0

Outgoing: 0

Update Update From... Compare With ... Publish Changes

Graph Revisor When Who Comment


```

I2C          i2c( p28, p27 );
test_LM75B   temp( i2c );

int main()
{
    float    t;

    i2c.frequency( 400 * 1000 );

    while(1) {
        t    = temp;
        printf( "temp = %7.3f\r\n", t );
        wait( 1 );
    }
}

```

main.cpp

```

class test_LM75B
{
public:
    test_LM75B( I2C &i2c_obj, char address = ADDRESS_LM75B );
    ~test_LM75B();
    void    init( void );
    float   read( void );
    operator float( void );
private:
    I2C     &i2c;
    char    adr;
};

```

```

#include "test_LM75B.h"

test_LM75B::test_LM75B( I2C &i2c_obj, char address )
    : i2c( i2c_obj ), adr( address )
{
    init();
}

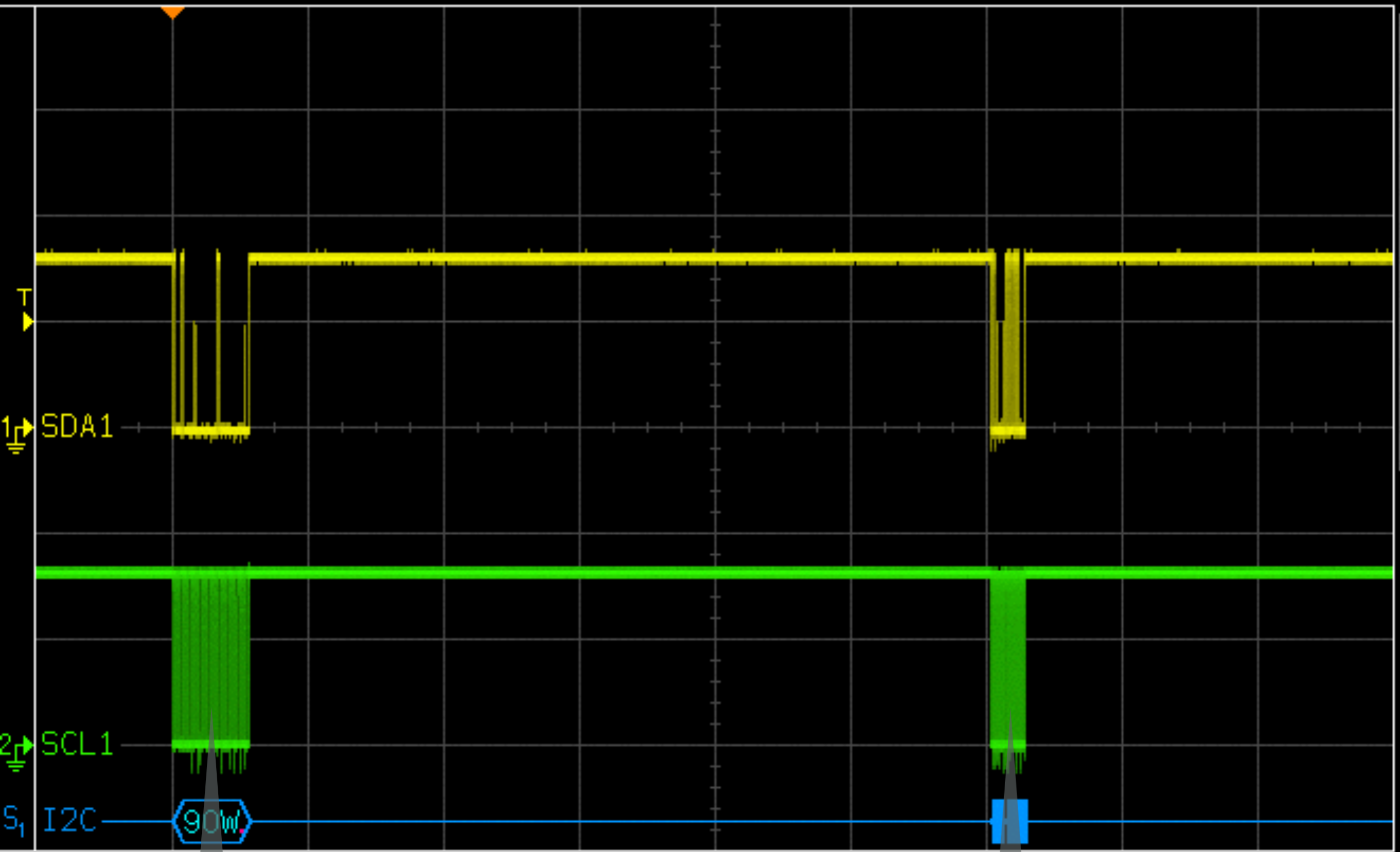
```

test_LM75B.cpp

元のI2Cインスタンスを参照できるようにしました。
ここではポインタではなく「参照型」を使っています。

オブジェクト自体をコピーするのではなく、
元のオブジェクトへの参照を渡しています。

参照型には代入はできません。
なので、初期化リストでの初期化が必要です。



初期化. 100kHz

できました. 400kHz

New Import Save Save All Compile Commit Revisions Help

Program Workspace

- My Programs
 - test_LM75B_Hello
 - test_LM75B
 - main.cpp
 - mbed

Revision History



Revisions of program "test_LM75B_Hello"

Showing revisions of program "test_LM75B_Hello" and public repository at okano/test_LM75B_Hello.

Commit Discard Changes Compare Switch Revert Merge

Graph	Revisor	When	Who	Comment
		8	okano	default tip to include published version library
		7	okano	comment added
		6	okano	two constructors version
		5	okano	(temporary)
		4	okano	chabged to referencing I2C object
		3	okano	modification on constructor : taking an I2C object
		2	okano	added : slave address change capability & operator o
		1	okano	device access is made as a class
		0	okano	very basic code for hardware verification

Revision 6 (b57b9dcfb515)

Comment two constructors version
 When 43 minutes ago
 Date 2014-11-03 00:26:12
 Files changed 2
 Lines changed 19

Revision log

All Changes

main.cpp	
test_LM75B.lib	

Remote changes for okano/test_LM75B_Hello

Incoming: 0 Outgoing: 0

Update Update From... Compare With ... Publish Changes

Graph Revisor When Who Comment

2つのバージョン

- インスタンスの作成方法
 - ピン名を渡すもの
 - I2Cインスタンスを渡すもの
- これらを分けて公開するのは効率が悪いので、統合してしまう
- C++は同じ関数名でも引数の種類が違うと、別物として扱ってくれる (^ ^)

こんな仕様にしてみた

- クラス内でI2Cにアクセスする際、インスタンス本体と参照を分けて処理するのは面倒.
 - なのでI2Cはコンストラクタ以外ではすべて参照を介してアクセス
- コンストラクタは
 - ピン名を渡されたら
 - そのピンを使うI2Cインスタンスを作る
 - そのインスタンスへの参照を保存して、以降はこれを使う
 - I2Cオブジェクトを渡されたら
 - そのインスタンスへの参照を作って保存. 以降はこれを使う
- デストラクタが呼ばれたら
 - 自身がI2Cインスタンスを保持しているのかどうかを確認して、保持しているなら開放する

```

.....
class test_LM75B
{
public:
    test_LM75B( PinName sda, PinName scl, char address = ADDRESS_LM75B );
    test_LM75B( I2C &i2c_obj, char address = ADDRESS_LM75B );
    ~test_LM75B();
    void      init( void );
    float     read( void );
    operator float( void );
private:
    I2C      *i2c_p;
    I2C      &i2c;
    char     adr;
};

```

2種類のコンストラクタ

I2Cインスタンスへのポインタと参照を保存できるようにしてある

```

.....
test_LM75B::test_LM75B( PinName sda, PinName scl, char address )
    : i2c_p( new I2C( sda, scl ) ), i2c( *i2c_p ), adr( address )
{
    init();
}

test_LM75B::test_LM75B( I2C &i2c_obj, char address )
    : i2c_p( NULL ), i2c( i2c_obj ), adr( address )
{
    init();
}

test_LM75B::~~test_LM75B()
{
    if ( NULL != i2c_p )
        delete i2c_p;
}
.....

```

ピン名を渡されたら：I2Cインスタンスを作成。さらにクラス内部ではI2Cインスタンスへの参照を使うのでその参照を初期化

オブジェクトを渡されたら、I2Cインスタンスは作らずに、そのオブジェクトへの参照を作る

コンストラクタでI2Cインスタンスが作られていたかどうかを確認し、自身がインスタンスを持っていたなら、それを開放する

```

#include "mbed.h"
#include "test_LM75B.h"

test_LM75B    temp0 ( p28, p27 );

I2C          i2c ( p28, p27 );
test_LM75B    temp1 ( i2c );

int main()
{
    float    t0;
    float    t1;

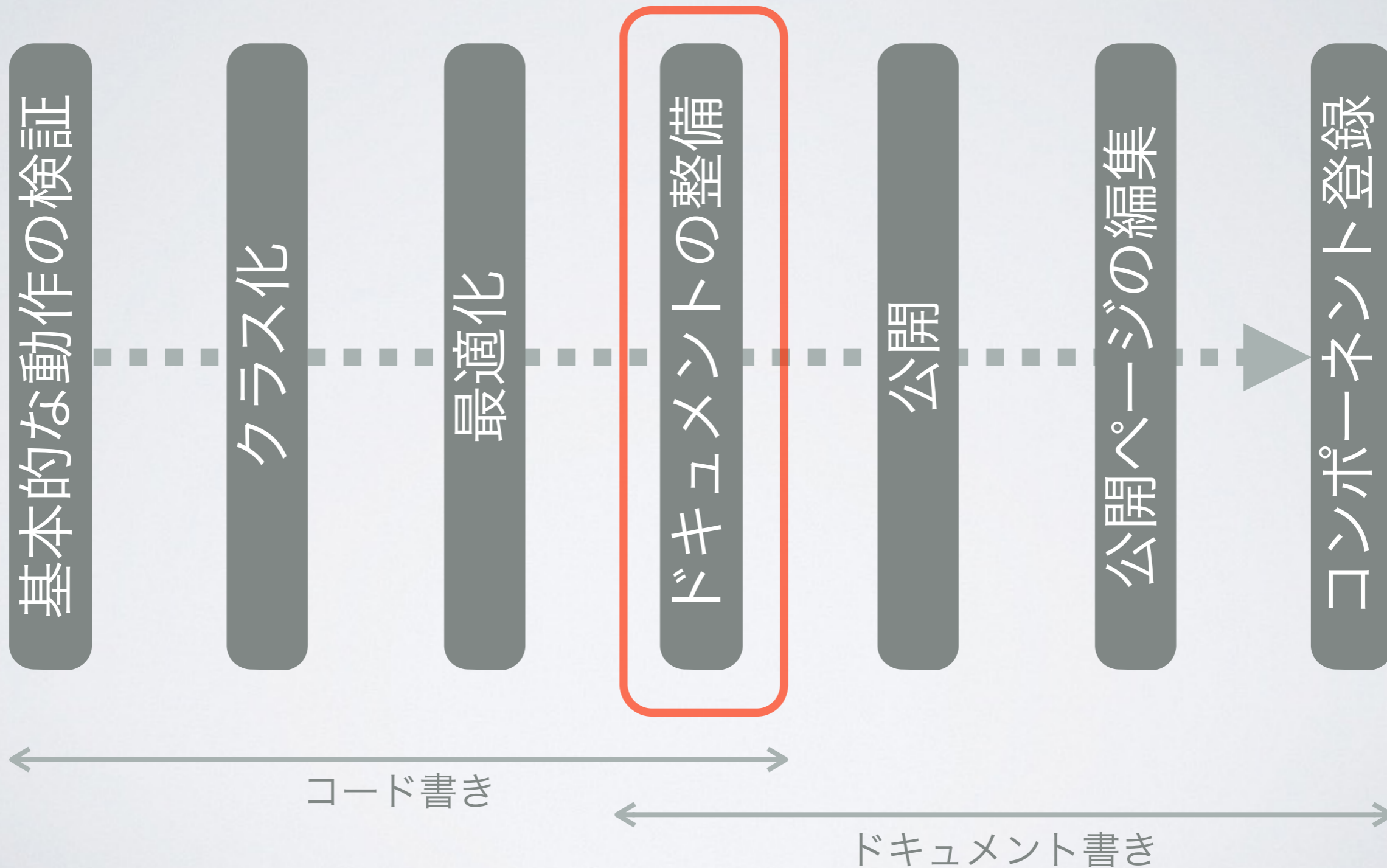
    i2c.frequency ( 400 * 1000 );

    while(1) {
        t0    = temp0;
        t1    = temp1;
        printf( "temp = %7.3f, %7.3f\r\n", t0, t1 );
        wait( 1 );
    }
}

```

main.cpp

ライブラリ作成の流れ



Program Workspace

- My Programs
 - test_LM75B_Hello
 - test_LM75B
 - main.cpp
 - mbed

Revision History

Revisions of program "test_LM75B_Hello"

Showing revisions of program "test_LM75B_Hello" and public repository at [okano/test_LM75B_Hello](#).

Commit Discard Changes Compare Switch Revert Merge

Graph	Revisor	When	Who	Comment	
		8	21 minutes ago	okano	default tip to include published version libran
		7	39 minutes ago	okano	comment added
		6	43 minutes ago	okano	two constructors version
		5	46 minutes ago	okano	(temporary)
		4	51 minutes ago	okano	chabged to referencing I2C object
		3	53 minutes ago	okano	modification on constructor : taking an I2C object
		2	58 minutes ago	okano	added : slave address change capability & operator o
		1	1 hour, 1 minute ago	okano	device access is made as a class
		0	1 hour, 1 minute ago	okano	very basic code for hardware verification

Revision 7 (553960b756ed)

Comment comment added
 When 39 minutes ago
 Date 2014-11-03 00:30:11
 Files changed 2
 Lines changed 5

Revision log

All Changes

- main.cpp
- test_LM75B.lib

Remote changes for [okano/test_LM75B_Hello](#) Incoming: 0 Outgoing: 0

Update Update From... Compare With ... Publish Changes

Graph	Revisor	When	Who	Comment
-------	---------	------	-----	---------

オンラインドキュメント

- APIの解説を書いておく
- オンライン・ドキュメント
- クラスライブラリの「.h」ファイルに
- Doxygenフォーマットで
 - コメントとして書いておけば、公開後に自動的にオンラインドキュメント形式に変換してくれる

developer.mbed.org

Platforms Components Handbook Cookbook Code Questions Forum

ARM mbed

Users - okano - Code - test_LM75B - Documentation

Tedd OKANO / test_LM75B

Code sample for class library development tutorial

Home History Graph **API Documentation** Wiki Pull Requests Admin settings

Back to documentation index Embed: <<library /users/okano/code/test_LM75B/do

test_LM75B Class Reference

test_LM75B class library More...

#include <test_LM75B.h>

Public Member Functions

<code>test_LM75B</code>	(PinName sda, PinName scl, char address=ADDRESS_LM75B)	Create a <code>test_LM75B</code> instance connected to specified I2C pins with specified address.
<code>test_LM75B</code>	(I2C &i2c_obj, char address=ADDRESS_LM75B)	Create a PCA9629A instance connected to specified I2C pins with specified address.
<code>~test_LM75B</code>	()	Destructor.
void	<code>init</code> (void)	Initialization.
float	<code>read</code> (void)	Read temperature.
	<code>operator float</code> (void)	Read temperature.

Detailed Description

test_LM75B class library

Class library to provide very simple interface for mbed

Example:

```

1 #include "mbed.h"
2 #include "test_LM75B.h"
3
4 test_LM75B temp0( p28, p27 );
5
6 I2C i2c( p28, p27 );
7 test_LM75B temp1( i2c );
8
9
10 int main()
11 {
12     float t0;
13     float t1;
14
15     i2c.frequency( 400 * 1000 );
16
17     while(1) {
18         t0 = temp0;
19         t1 = temp1;
20         printf( "temp = %7.3f, %7.3f\r\n", t0, t1 );
21         wait( 1 );
22     }
23 }

```

developer.mbed.org

mbed /how_to_cook_class_lib/test_LM75B/Classes/test_LM75B.doc

New Import Save Save All Compile Commit Revisions

Program Workspace

- My Programs
 - how_to_cook_class_lib
 - test_LM75B**
 - Classes
 - test_LM75B
 - test_LM75B.cpp
 - test_LM75B.h
 - main.cpp
 - mbed

test_LM75B Class Reference

```

#include <test_LM75B.h>

Public Member Functions
test_LM75B (PinName sda, PinName scl, char address=ADDRESS_LM75B)
Create a test_LM75B instance connected to specified I2C pins with specified address.
test_LM75B (I2C &i2c_obj, char address=ADDRESS_LM75B)
Create a PCA9629A instance connected to specified I2C pins with specified address.
~test_LM75B ()
Destructor.
void init (void)
Initialization.
float read (void)
Read temperature.
operator float (void)
Read temperature.

Detailed Description
test_LM75B class library
Class library to provide very simple interface for mbed
Example:
1 #include "mbed.h"
2 #include "test_LM75B.h"
3
4 test_LM75B temp0( p28, p27 );
5
6 I2C i2c( p28, p27 );
7 test_LM75B temp1( i2c );
8
9
10 int main()
11 {
12     float t0;
13     float t1;
14
15     i2c.frequency( 400 * 1000 );
16
17     while(1) {
18         t0 = temp0;
19         t1 = temp1;
20         printf( "temp = %7.3f, %7.3f\r\n", t0, t1 );
21         wait( 1 );
22     }
23 }

Definition at line 52 of file test_LM75B.h.

Constructor & Destructor Documentation
test_LM75B ( PinName sda,
            PinName scl,
            char address = ADDRESS_LM75B
            )

```

- 「/** ~ */」などの形のコメントとして書く
- 「@」を付けて、各引数や返り値の解説を書く
- その他の情報も@を付けて
 - サンプル
 - 作者
 - バージョン
 - などなど

```

*
*   i2c.frequency( 400 * 1000 );
*
*   while(1) {
*       t0  = temp0;
*       t1  = temp1;
*       printf( "temp = %7.3f, %7.3f\r\n", t0, t1 );
*       wait( 1 );
*   }
* }
* @endcode
*/
class test_LM75B
{
public:

    /** Create a test_LM75B instance connected to specified I2C pins with specified address
    *
    * @param sda I2C-bus SDA pin
    * @param scl I2C-bus SCL pin
    * @param address (option) I2C-bus slave address (default: 0x90)
    */
    test_LM75B( PinName sda, PinName scl, char address = ADDRESS_LM75B );

    /** Create a test_LM75B instance connected to specified I2C pins with specified address
    *
    * @param i2c_obj I2C object (instance)
    * @param address (option) I2C-bus slave address (default: 0x90)
    */
    test_LM75B( I2C &i2c_obj, char address = ADDRESS_LM75B );

    /** Destructor
    */
    ~test_LM75B();

    /** Initialization
    */
    void    init( void );

    /** Read temperature
    *
    * @return value of degree Celsius (in float)
    */
    float  read( void );

    /** Read temperature
    *
    * @return the object returns the read value
    */
    operator float( void );

private:
    I2C    *i2c_p;
    I2C    &i2c;
    char    adr;
};

```

DOXYGENの解説ページ

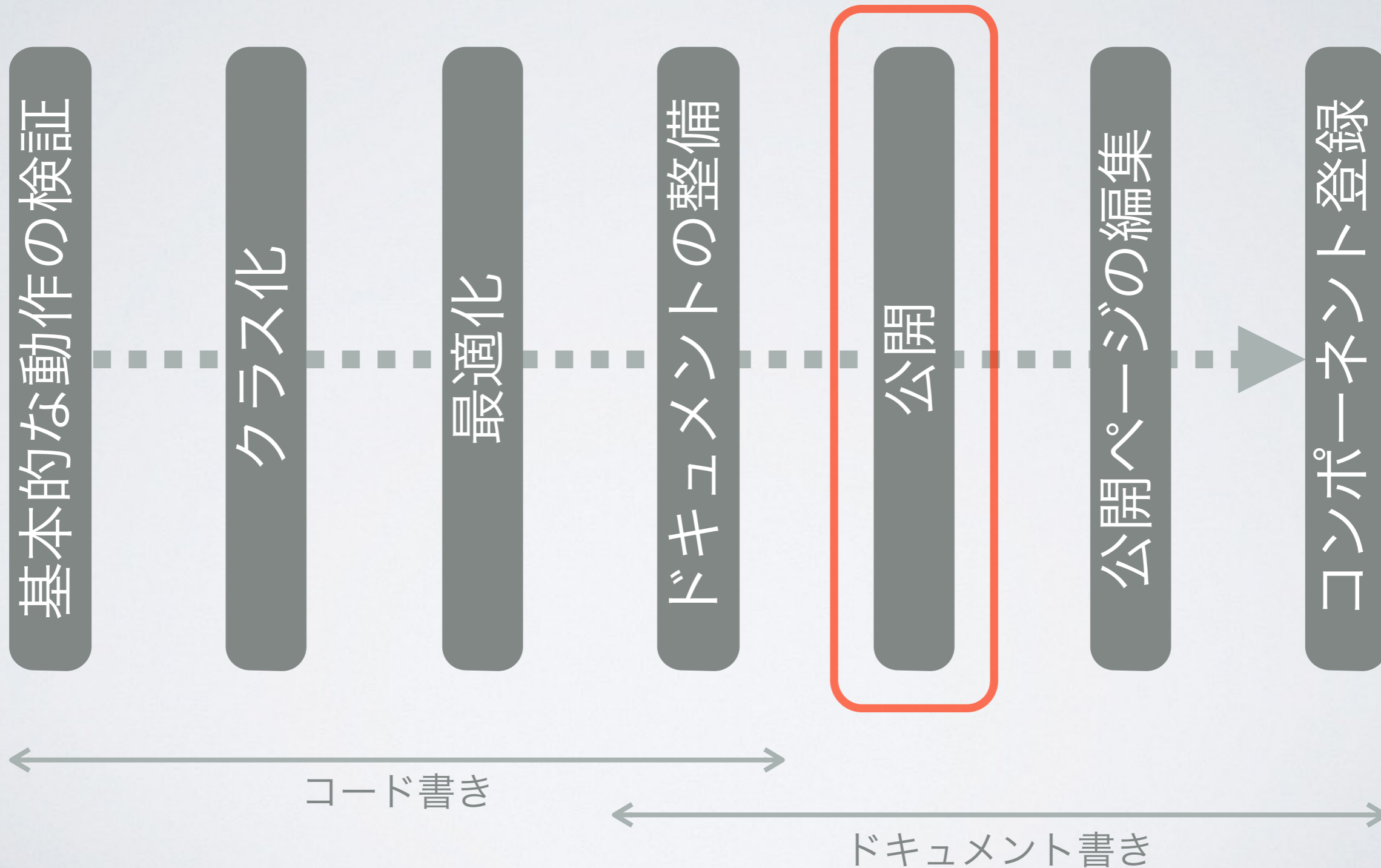
- <https://developer.mbed.org/handbook/API-Documentation>

The screenshot shows the 'API Documentation' page on the mbed developer website. The page is titled 'API Documentation' and has a sub-header 'What is API documentation?'. The main content explains that API documentation is a quick and concise reference containing what you need to know to use a library or work with a program. It details functions, classes, return types, and more. It also mentions that in mbed, API documentation for programs and libraries is fully supported both within the Compiler and in the code listings on the public site.

The page includes a table of contents for 'The mbed Tools' and 'Information' sections. The 'The mbed Tools' section includes: 1. Introduction, 2. The mbed Website, 3. The mbed Compiler, 4. Importing code, 5. Collaboration, 6. API Documentation, 7. Browsing documentation, 8. Adding documentation, 9. Extra features, 10. Final words, 11. Exporting code, 12. Getting help. The 'Information' section includes: Classes, methods, functions, etc which exist in the source code but aren't documented won't appear in the documentation.

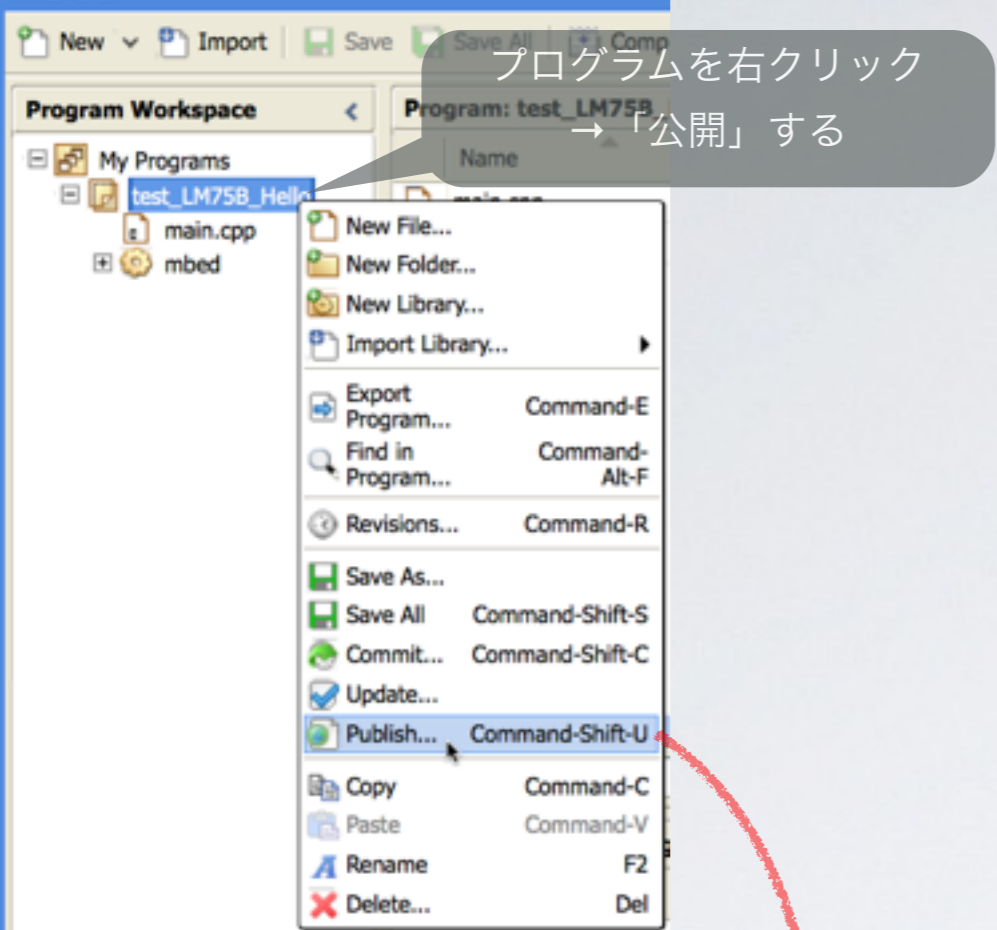
At the bottom of the page, there are two screenshots of the mbed Compiler's 'Program Workspace' showing the 'TextLCD' component and its associated files and classes.

ライブラリ作成の流れ



公開

- ライブラリとプログラムを公開
 - 未公開ライブラリを含むプログラムを公開しようとする
と、先にライブラリ公開を促される
 - リポジトリにコミットされていないプログラム/ライブラリ
を公開しようとする、先にコミットするよう促される
- 必要事項を入力してOKボタンを押すだけ！



公開名

公開プログラム/ライブラリの概略を書いておく

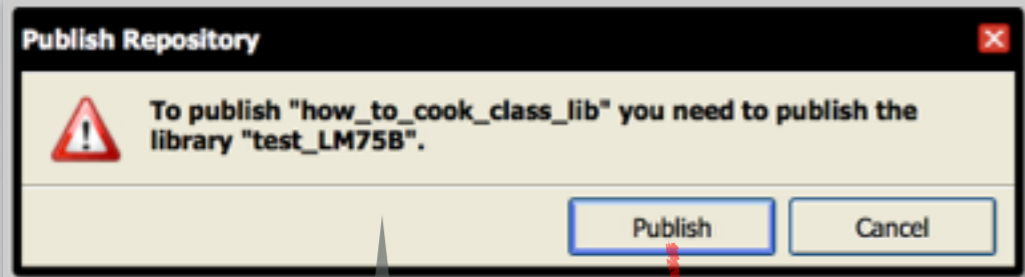
キーワード(型番や機能など)

公開はプログラムとして? ライブラリとして?

公開元, 個人として? チームとして?

他の人から見えるようにするかどうか

公開ライセンス



まだライブラリを公開していなければ, それが促される

公開時のダイアログの例

公開するとこんな感じのページができます

The screenshot shows the ARM mbed developer website interface. At the top, there's a navigation bar with links for Platforms, Components, Handbook, Cookbook, Code, Questions, and Forum. On the right, there are buttons for Dashboard and Compiler. Below the navigation, the ARM mbed logo is displayed on the left, and a search bar with a 'Go' button is in the center. On the right, a user is logged in as 'okano' with a 'Logout' button.

The main content area shows the user's profile 'Tedd OKANO' and a repository named 'test_LM75B_Hello'. The repository description is 'example project to explain how to write a class library'. Below this, there are 'Dependencies' listed as 'mbed' and 'test_LM75B'. A horizontal menu contains links for Home, History, Graph, API Documentation, Wiki, Pull Requests, and Admin settings. There are also links to 'Edit repository homepage' and 'Download repository' options for zip and gz.

The 'Files at revision 8:d01c24c1223c' section contains a table with the following data:

Name	Size	Actions
[up]		
main.cpp	412	Revisions Annotate
mbed.bld	65	Revisions Annotate
test_LM75B.lib	68	Revisions Annotate

At the bottom of the main content area, there are two buttons: 'Ask a question' and 'Start a discussion'.

On the right side, there is a 'Repository toolbox' with several buttons: 'Import this program', 'Export to desktop IDE', 'Build repository', 'Send Pull Request from here', 'Make featured', 'Following', and 'Embed url:'. Below these is a 'Clone repository to desktop' section with a terminal command: 'hg clone https://okano@devel'.

At the bottom right, there is a 'Repository details' section showing 'Type: Program'.

ちょっと殺風景

公開ページの例

ARM mbed

Teams - CQ Publishing - Code - MARMEX_VB_Hello

CQ Publishing / MARMEX_VB_Hello

A "Hello" program for MARMEX_VB library. This application may work 40pin type mbed platforms :) This application expects to have the MARMEX_VB module on a "MAPLE mini type-B (MARM03-BASE)" baseboard (slot2) with a MARMEX_OB module (on slot1)

Dependencies: MARMEX_VB NokiaLCD mbed

Home History Graph API Documentation Wiki Pull Requests

Sample code for MARMEX-VB (MARM-VB) camera module.

Download repository: zip gz



このページのこの部分は編集できます

This is a very simple program just copies the data from camera to OLED.



Repository toolbox

- Import this program
- Export to desktop IDE
- Build repository
- Follow

Embed url:
<<program /teams/CQ-Publ

Clone repository to desktop:

hg clone https://okano@dev

公開後はインポートされた回数

なども表示されます

Repository details

Type:	Program
Created:	09 Jun 2014
Imports:	31
Forks:	0
Commits:	3
Dependents:	0
Dependencies:	3
Followers:	10

Software licencing information



The code in this repository is Apache licensed.

Apache2ライセンスの項にチェックマークを入れると、オープンソースのマークが付きます

CQ出版のチームとして公開

http://developer.mbed.org/teams/CQ-Publishing/code/MARMEX_VB_Hello/

公開後の変更は可能か？

Admin settingsタブ内で
様々な項目が編集可能です。
まずはこちらの項目編集を
検討しましょう

それでもやっぱり消去するなら..
Admin settings表示のずーっと
下の方に「消去ボタン」があります

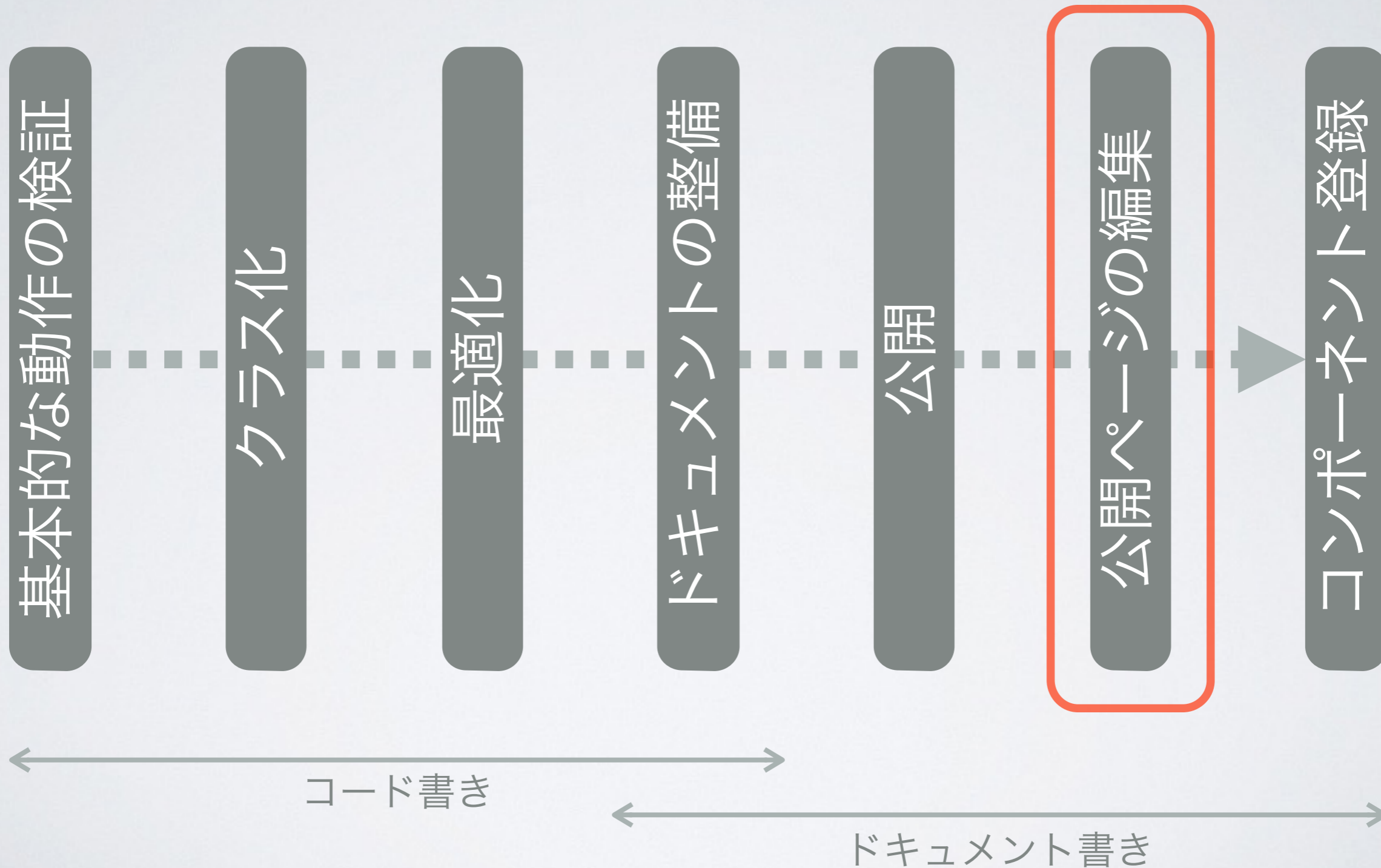
公開ページを一旦消去し、再度
同じ名前で公開することも
可能です。

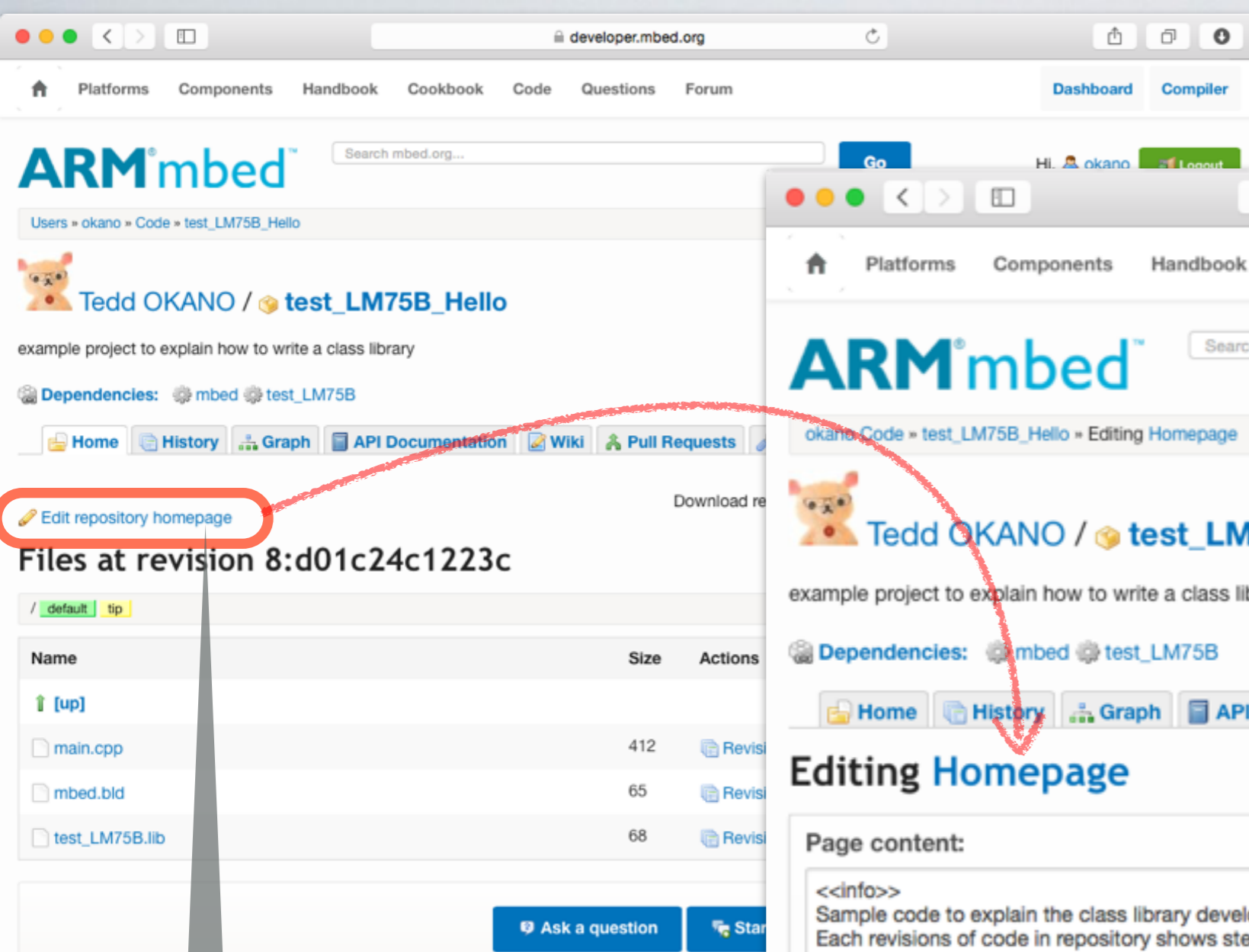
全く新規にページが作られるので
公開日やインポート数なども
新しいものとなります

The screenshot shows the ARM mbed developer website interface. At the top, there's a navigation bar with 'Dashboard' and 'Compiler' links. Below that, the user profile 'Tedd OKANO / test_LM75B_Hello' is visible. A red circle highlights the 'Admin settings' link in the top navigation bar. A red arrow points from this link to the 'Delete' section of the repository page. The 'Delete' section contains a warning: 'Deleting a repository is permanent and irreversible.' and a confirmation question: 'Are you sure you want to delete this repository? Are you really sure?'. A red circle highlights the 'Yes, delete the repository 'test_LM75B_Hello'' button. A red arrow also points from the 'Admin settings' link to this button. The 'Copy repository' section is partially visible above the 'Delete' section.

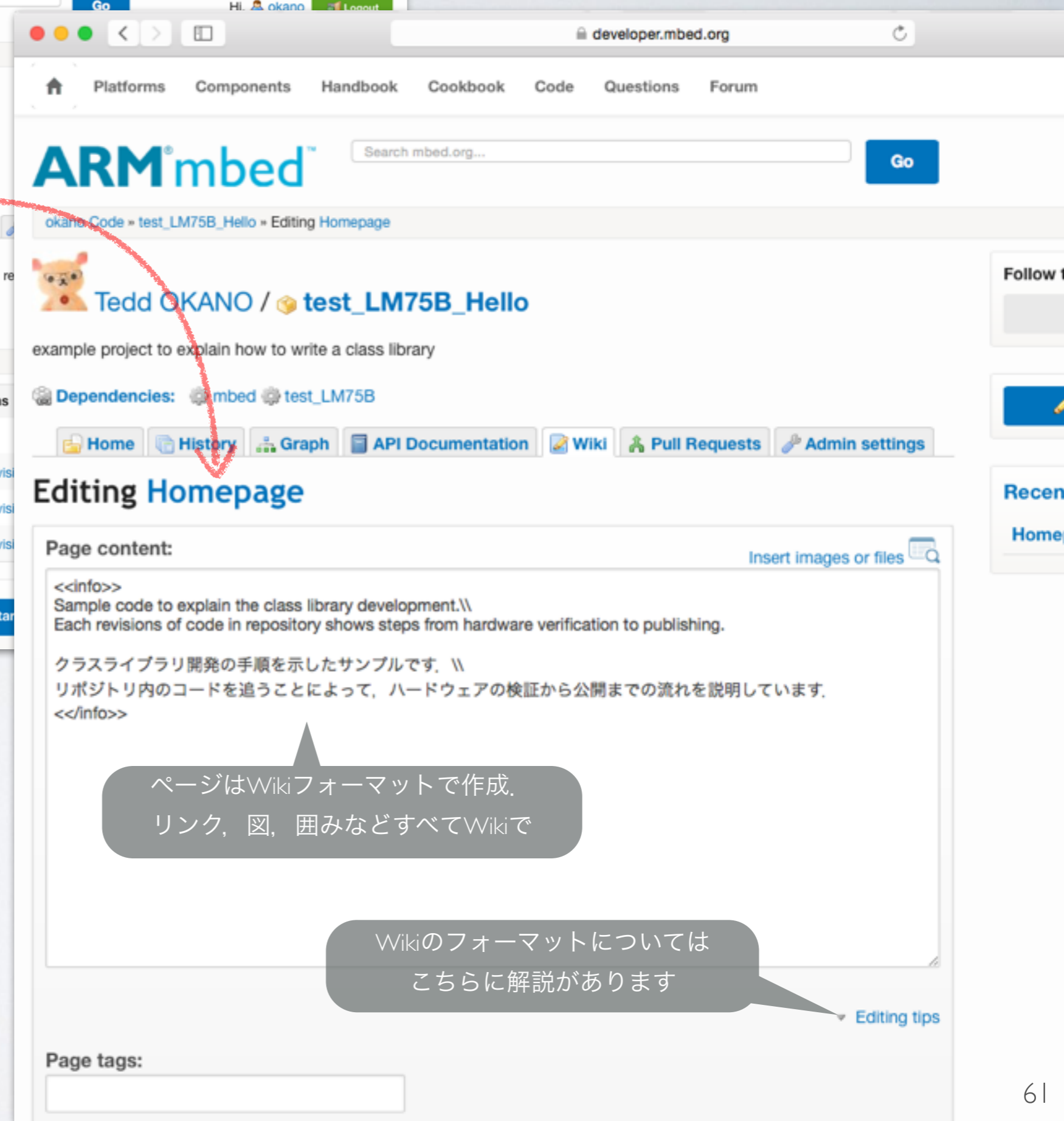
消去ボタンを押すと「ホントに消してもいいの？」と念を
押されます。そこでもう一度考えてから、消去しましょう

ライブラリ作成の流れ





「編集」リンクをクリックすると、公開ページを編集できる



ページはWikiフォーマットで作成、リンク、図、囲みなどすべてWikiで

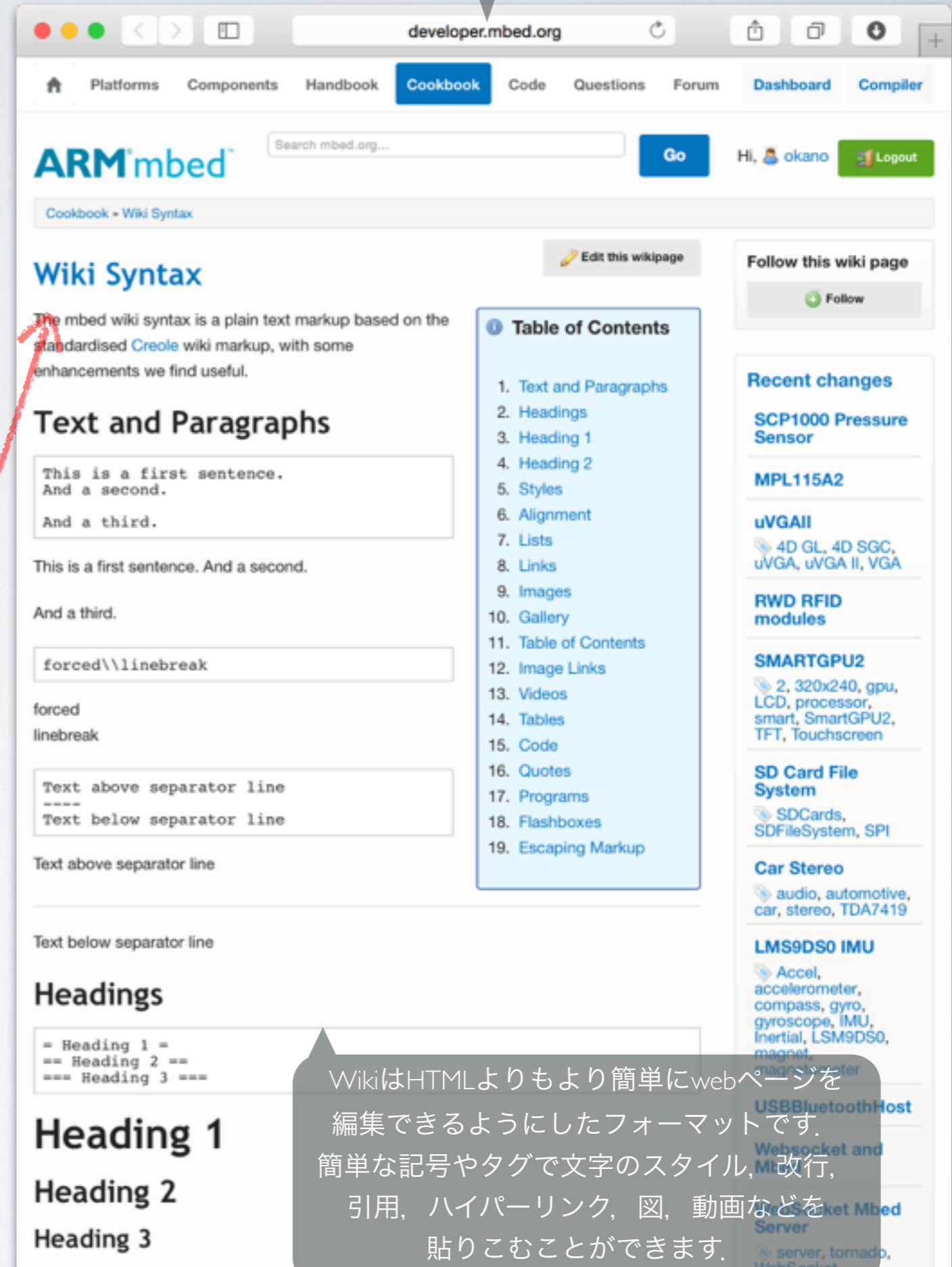
Wikiのフォーマットについてはこちらに解説があります



Wikiの書式については「Editing tips」をクリックするとページ内に解説が表示されます

「Full Wiki syntax」をクリックすると別ウィンドウにより詳しい解説が開きます

「Preview」を押すことで保存・公開前に表示を確認できます



WikiはHTMLよりもより簡単にwebページを編集できるようにしたフォーマットです。簡単な記号やタグで文字のスタイル、改行、引用、ハイパーリンク、図、動画などを貼りこむことができます。

ARM mbed

Forum - mbed LPC1768 - Char array to int issue

mbed LPC1768
Rapid Prototyping for general microcontroller applications

Char array to int issue

Topic last updated 12 Aug 2014, by Tedd OKANO. 1 reply

Fernanda Patton
07 Aug 2014

Hi all,

I've been trying to retrieve two specific characters from a file using a lot of different methods. Mostly I've been trying to use ATOI() but of my code it just prints out a 1. //

```
char rec[2];

FILE *fp = fopen("/sd/mydir/trailer4.txt", "r");

fseek(fp,45,SEEK_SET); rec[0]=fgetc(fp); get a character/byte

fseek(fp,46,SEEK_SET); rec[1]=fgetc(fp);

int rec2= atoi(rec);

fclose(fp);
```

Tedd OKANO
12 Aug 2014

The string should be terminated by "null character". So you need more byte in rec[] array and put zero in last element. I think

```
sample
1 char rec[ 3 ];
2 FILE *fp = fopen( "/sd/mydir/trailer4.txt", "r" );
3
4 fseek( fp, 45, SEEK_SET );
5 rec[ 0 ] = fgetc( fp );
6 rec[ 1 ] = fgetc( fp );
7 rec[ 2 ] = '\0';
8
9 int rec2= atoi( rec );
10
11 fclose( fp );
```

Post reply

ARM mbed

Questions - LPC1768での実行

Hide Tanaka

1 month, 1 week ago.

LPC1768での実行

LPC1768で実行させる場合、プログラムのNJL5501Rを使用、増幅後出力波形が1が全く点灯しません。

mainルーチン先頭にLED2点灯させようとしています。

```
int main() { uint32_t num; int32_t wave;
    . . . }
```

よろしくをお願いします。

Question relating to:

tg_201410s6_Plethysm
トランジスタ技術2014年10月号

Comment on this question

2 Answers

Sort by: Votes Active Newest

Tetsuro Tanaka
1 month, 1 week ago.

Tanakaさん
質問ありがとうございます。

OKANOさん
ご回答ありがとうございます。

mainの先頭が動いていないとなると、LPC1768のpinoutを確認し、pinoutが正しいか確認してください。

問題無い場合、今のところ他に思いつきません。

「Update all libraries to the latest revision」にチェックを入れて、Libraryを最新にしてください。

Import Program
Import a program from mbed.org into your work

ARM mbed

Users - okano - Code - SB1602E_test

Tedd OKANO / SB1602E_test

Test program for SB1602E class library

Dependencies: SB1602E mbed

Fork of TextLCD_SB1602E by Tedd OKANO

Home History Graph API Documentation

Test program for the text LCD "SB1602E" class library

This is the version 2.0 of the TextLCD_SB1602E.



Update all libraries to the latest revision

Import Program

Import Program
Import a program from mbed.org into your work

mbed

TextLCD

GND (1) Vcc (40)

ARM mbed

Users - okano - Notebook - イカ醤油ポッコ焼きはイカにしてか(´I`);

Tedd OKANO. post a reply

2013年、夏のある日。お昼ごはんを食べたあと、オフィスへ帰る途中、Twitterへの投稿、これがそもそものきっかけだった。

イカ醤油ポッコ焼き 以前

その前のしばらくの間、LPC1114FN28とLPC810にmbedオンボードで行ったりすることがよくあった。

当時はまだmbed LPC1114FN28のような便利な環境は誕生しなかった。フラッシュメモリに書き込みを行わなければならなかった。これを手っ取り早く行うにはLPCXpressoのような統合開発環境はいいのだけれど、せっかくmbedでビルドしたコードをわざわざ環境に慣れせしめ、フッターの環境で作業しなければならなかった。ここでこれらのDIPパッケージ・チップにはもうひとつの書き込み方法、シリアル(UART)インターフェースを介して書き込みが行える。ISPを行うにはPCとケーブルとISPの手順を実行するソフトが必要。今となってはシリアルポートを持ったPCなど無い。USBとUARTの便利なものもあるが、PCにドライバを入れたり、それも面倒。ISPの手順を実行するソフトはFlashMagicやlpc21ispなどWindowsはbinフォーマットをそのまま扱うことが出来ず、Intelhexフォーマットもやたらと面倒。

lpc21ispはオープンソースで自由度も高いが、コマンドラインで実行するのは面倒。

- DIP_ARM LPC1114FN28のプログラムをシリアルで書き込み
- LPC810 (= DIP8_ARMマイコン)のフラッシュへの書き込み

そんなわけで「面倒だなあ」と思いつつも、ずーっとそういう環境でやってきた。

一方、とても便利なmbedはUSBシリアルの変換などお手のもの。実際にISPの機能を使ってLPC1768に書き込む方法として、調べていた。USB-シリアルのケーブルをわざわざ買ってこなくて済む。USB-シリアル変換はmbedにさせてたりしてた。

口は災いの元

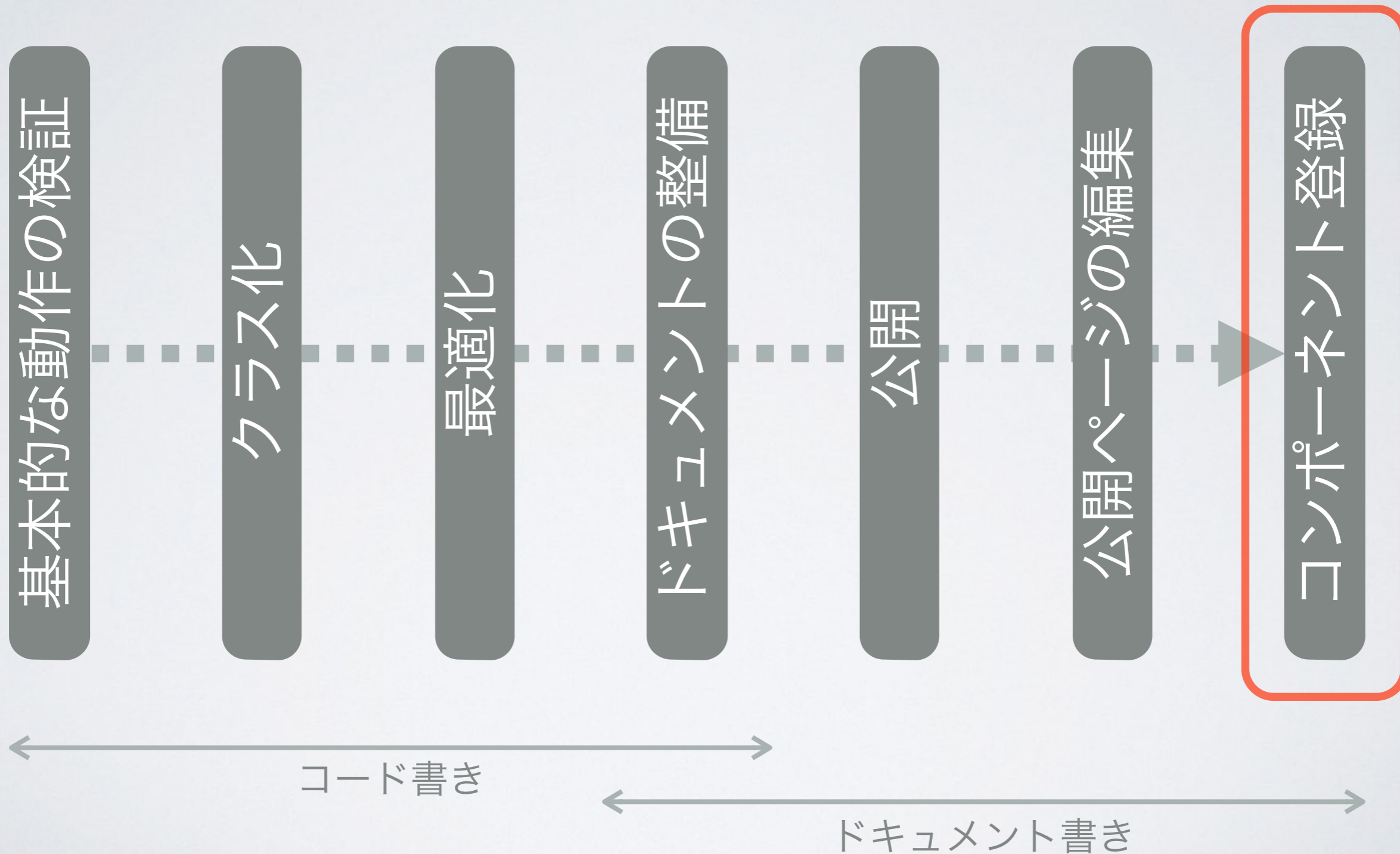
で、思いついた。

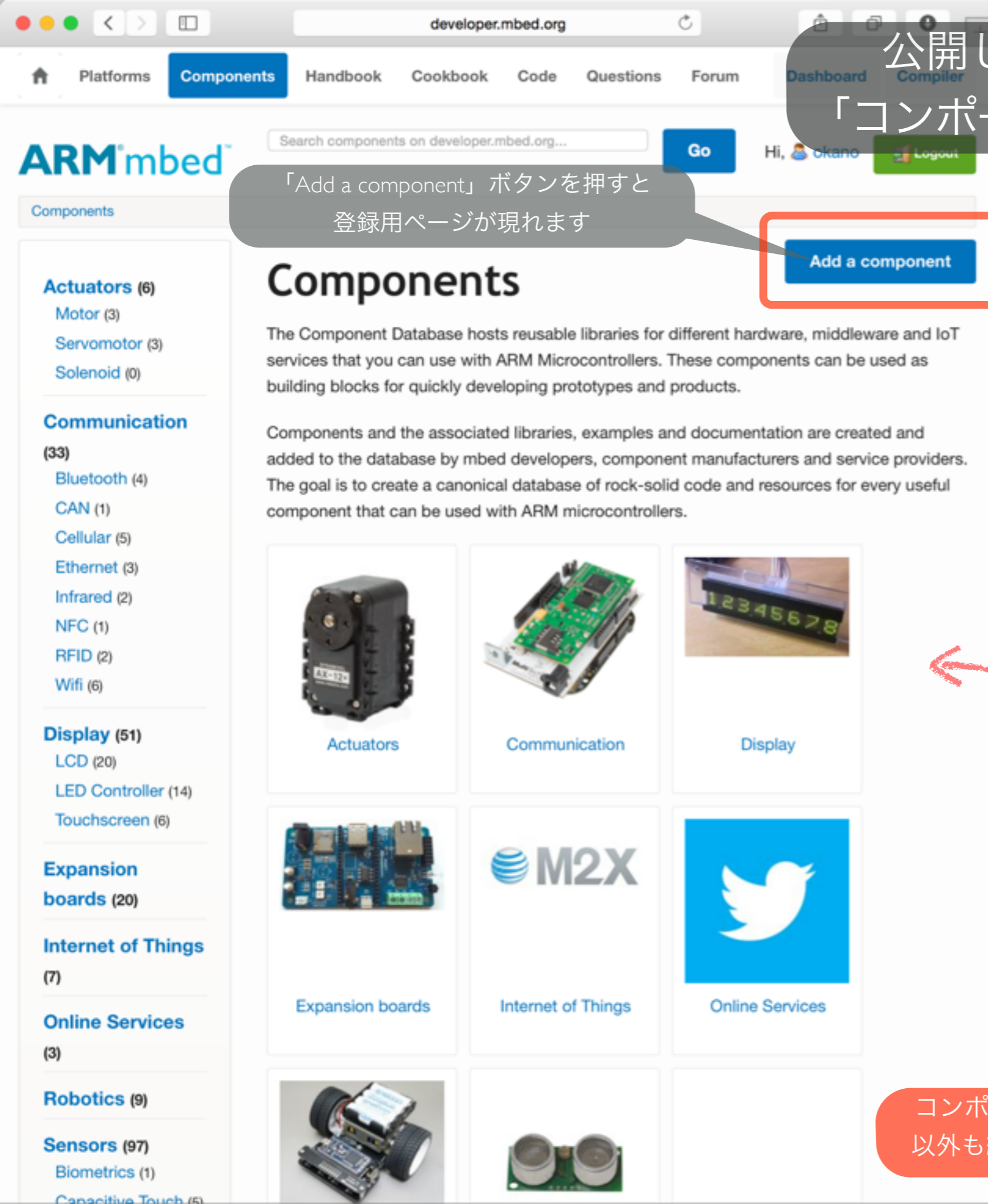
(´I`) .oO(せっかくmbedを使っているのだから、PCで実行すればいい。それがお昼ごはんを食べた後のエスカレーターの上から投稿した。lpc21ispをmbedにポートしたら済む話と思ったんですね。

しかしこの投稿をしてから思い出した。私のTwitter

Wikiフォーマットはコードの公開ページだけでなく、フォーラム、Q&A、ノートページなど様々なページの編集に使われます。

ライブラリ作成の流れ





公開しただけではもったいない
「コンポーネンツ・ページ」に登録を！

「Add a component」 ボタンを押すと
登録用ページが現れます

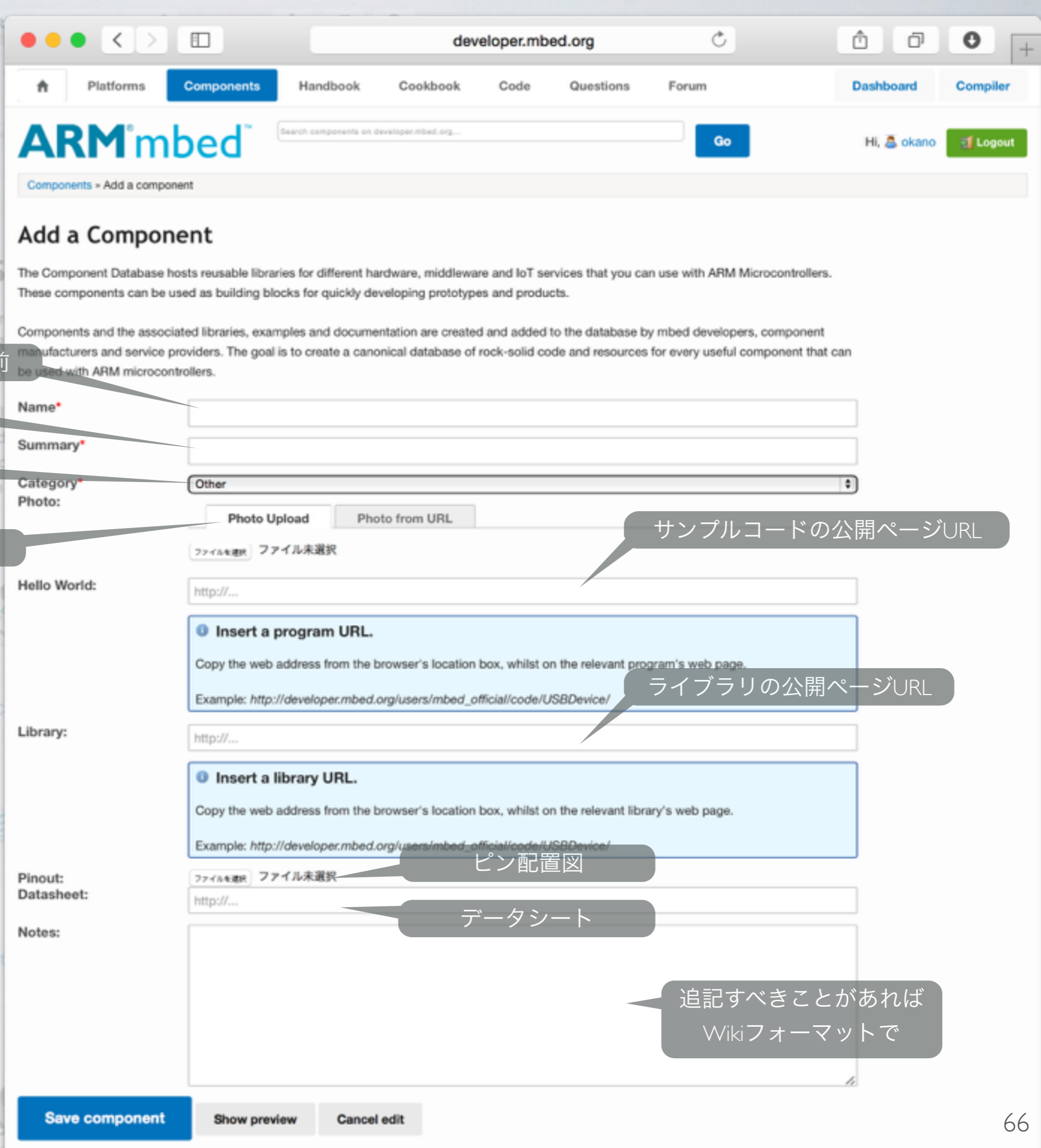
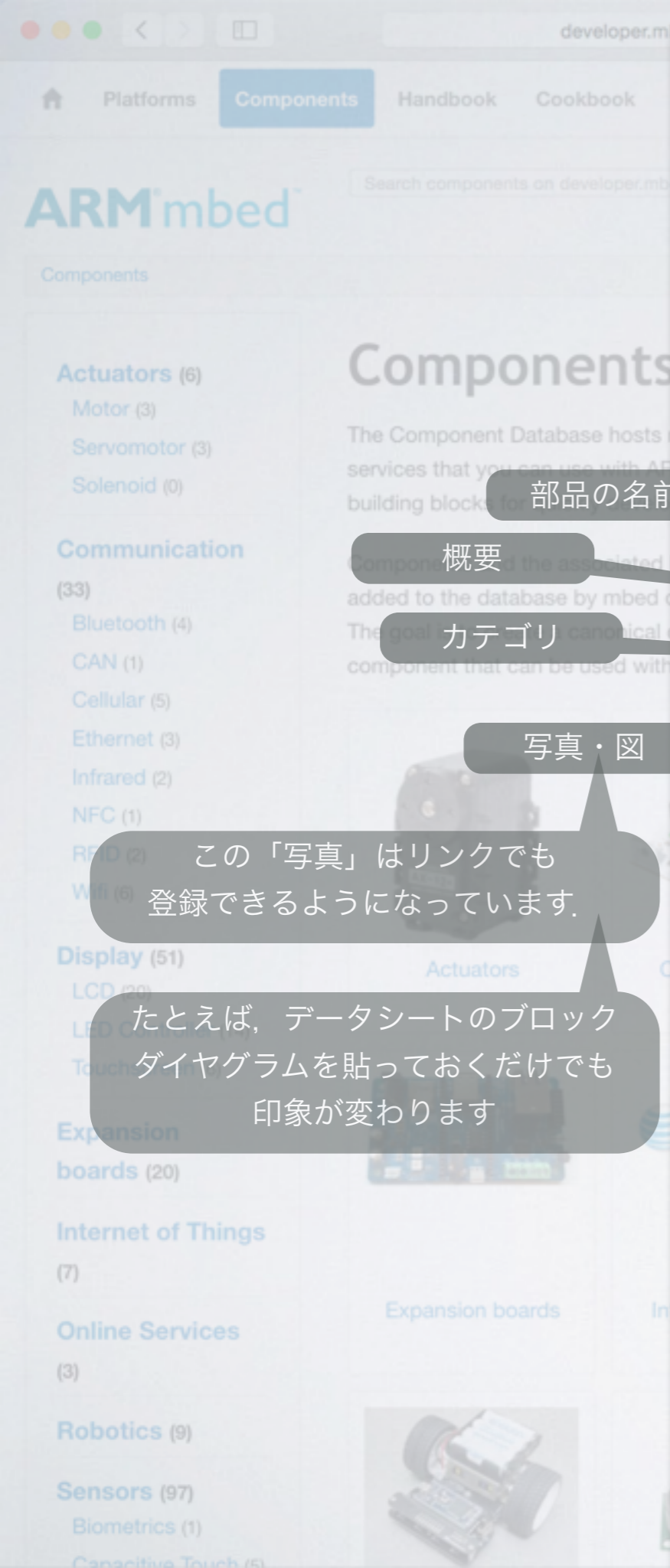
「コンポーネンツ・ページ」は誰でもページを追加できる「部品のデータベース」
ここに登録しなくても、部品型番をサイト内検索するなどで見つけてもらえますが、ここがあれば見つかるチャンスが増えます

これらのリンクを登録し、
簡単な説明を書いてやるだけ！

すでに用意出来ているものを
登録するだけなのですぐできます

- ライブラリ
- HelloWorldサンプル
- 結線図・データシート

コンポーネント・ページは、そのページの作成者以外も編集できます。誰でも変更・追記が可能です



部品の名前

概要

カテゴリ

写真・図

サンプルコードの公開ページURL

この「写真」はリンクでも登録できるようになっています。

ライブラリの公開ページURL

たとえば、データシートのブロックダイアグラムを貼っておくだけでも印象が変わります

ピン配置図

データシート

追記すべきことがあれば Wikiフォーマットで

Name* MARMEX-VB (MARY-VB) Camera module

Summary* Camera module operation library. mbed controls a

Category* - Imaging

Image

Upload image Image from URL

Currently: components/components/DSC_0497.jpg
Change: ファイルを選択 ファイル未選択

Hello World:

http://mbed.org/teams/CQ-Publishing/code/MAR

Library:

http://mbed.org/teams/CQ-Publishing/code/MAR

Pinout:

Currently: components/pinouts/MARMEX-VB_sche
Change: ファイルを選択 ファイル未選択

Datasheet:

http://marutsu.co.jp/data/0000000200314097.pdf

Notes:

<<info>>
Place an order from here. : [[http://www.marutsu.co.jp/shohin_238825/|MARMEX-VB(MARY-VB)]]
<</info>>

<<info>>
日本語による解説が、こちらのページ[[/teams/CQ-Publishing/wiki/mbed-components/MARMEX-VB-library]]
Japanese page is available in next URL : [[/teams/CQ-Publishing/wiki/mbed-components/MARMEX-VB-library]]
<</info>>

A camera module operation library for [[http://www.marutsu.co.jp/shohin_238825/|MARMEX-VB(MARY-VB)]].

developer.mbed.org

Platforms Components Handbook Cookbook Code Questions Forum

ARM mbed

Components > Sensors > Imaging > MARMEX-VB (MARY-VB) Camera module

MARMEX-VB (MARY-VB) Camera module

Camera module operation library. mbed controls and transfer data via I2C and SPI interfaces

Hello World

MARMEX_VB_Hello Import program

A "Hello" program for MARMEX_VB library. This application may work 40pin type mbed platforms :) This application expects to have the MARMEX_VB module on a "MAPLE mini type-B (MARM03-BASE)" baseboard (slot2) with a MARMEX_OB module (on slot1)

Last commit 20 Jun 2014 by CQ Publishing

Library

MARMEX_VB Import library

MARMEX-VB : "Mary Camera module" library

Last commit 20 Jun 2014 by CQ Publishing

Pinout

Follow this component

Follow

Tested platforms

- NP mbed LPC1768
- NP mbed LPC11U24
- NP TG-LPC11U35-501

Tested platforms:

mbed LPC1768 × mbed LPC11U24 ×

TG-LPC11U35-501 ×

Save component Show preview Cancel edit

コンポーネントの登録後、再度編集画面を開くと「Tested platforms」が入力できます。

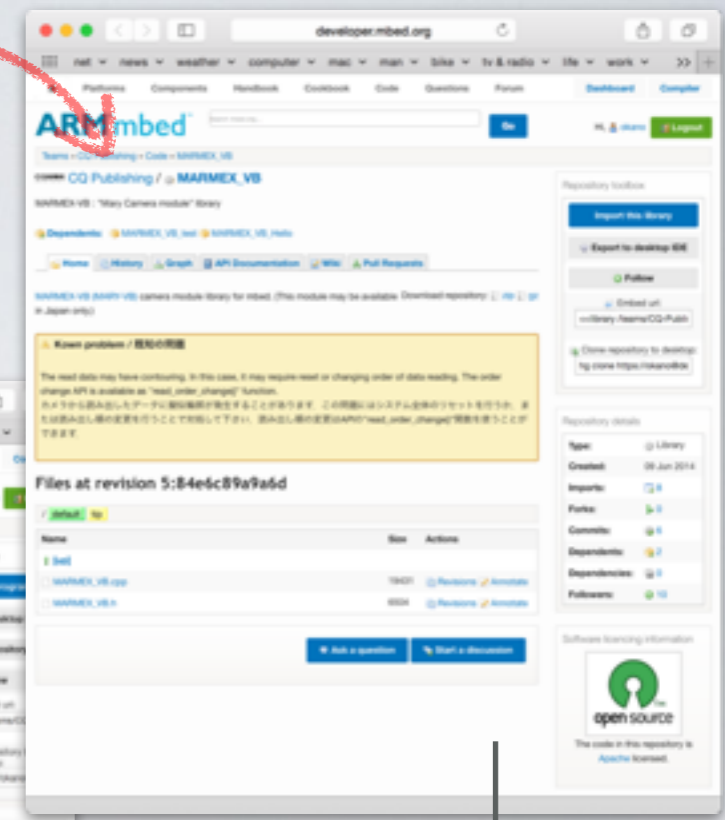
コンポーネント・ページは関連情報へのハブ

ライブラリ公開・ページ

コンポーネント・ページ



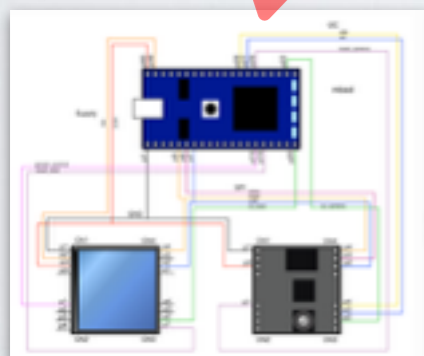
Helloプログラム公開・ページ



外部リンク(データシート)



接続図など



写真・ビデオ



その他プログラム公開・ページ

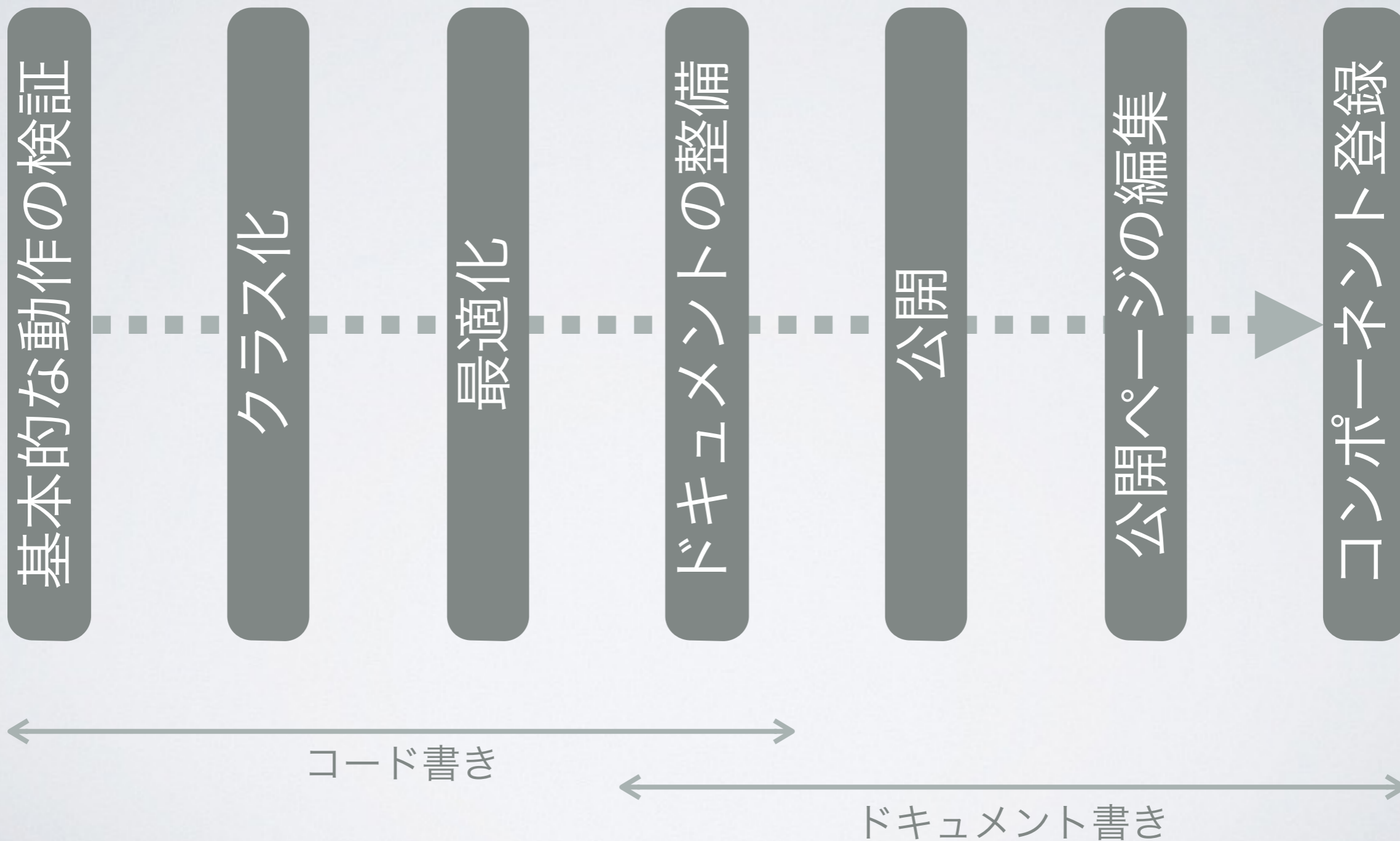


ライブラリ

Helloプログラム

その他プログラム

ライブラリ作成の流れ



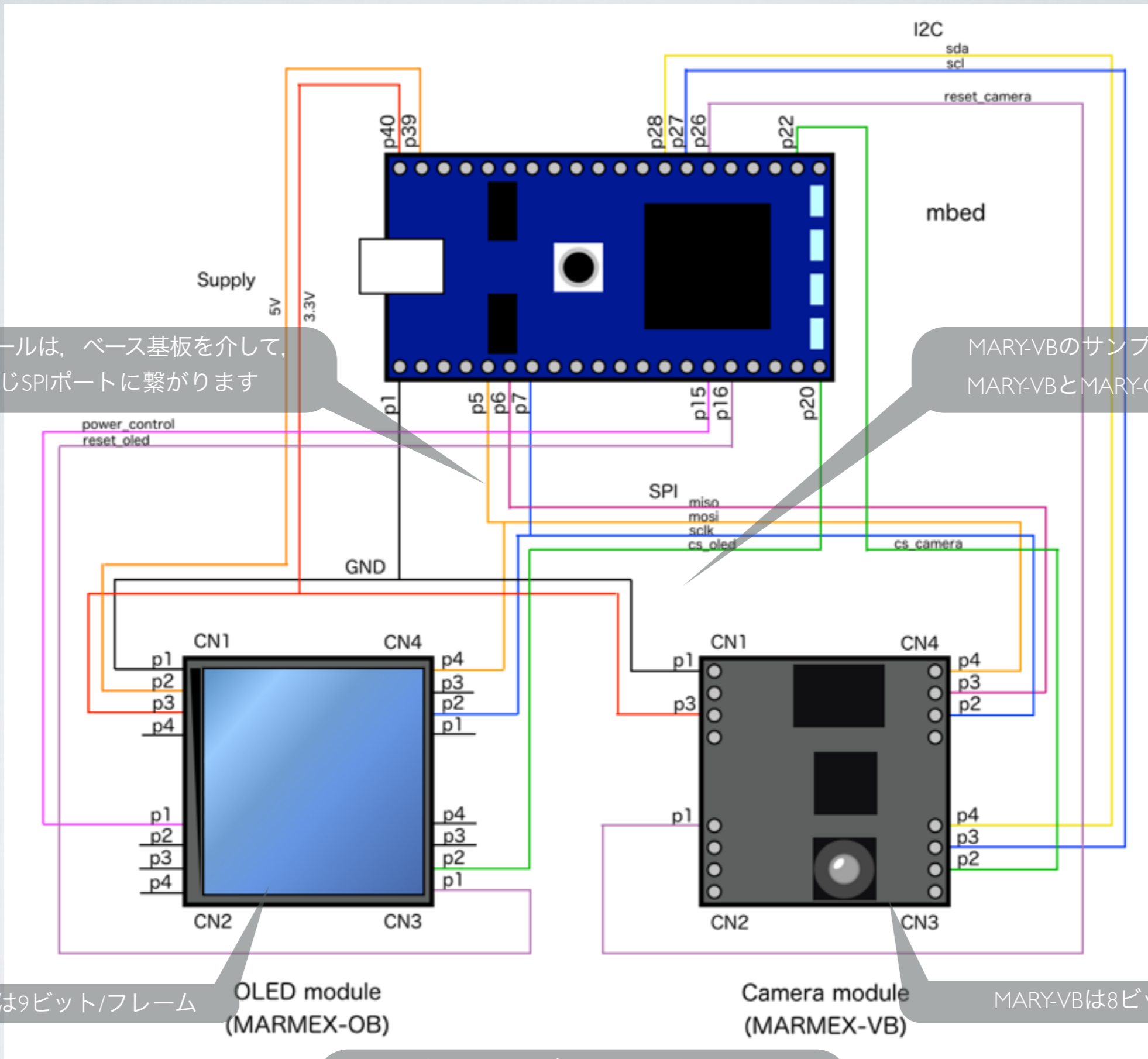
公開後

- 特にすることはありませんが..
 - 必要があればアップデート
 - バグフィクス
 - 機能追加
 - 対応プラットフォーム追加
 - ユーザからのプルリクエスト
 - 同じ部品用のライブラリが、フォークして公開されている場合にはPullリクエストを出してもらいましょう
- (できれば)インポート数が上がるように、公開ページの改良・宣伝活動など
 - 使ってもらえればそれだけフィードバックも多くなります
 - 多くの改善のアイデアで、よりよいライブラリに

参考 (I)

- クラス内でのインターフェース・インスタンスの保持について
- MARY-VBを例に





これらのモジュールは、ベース基板を介して、どちらも同じSPIポートに繋がります

MARY-VBのサンプルコードでは MARY-VBとMARY-OBを使います

MARY-OBは9ビット/フレーム

OLED module (MARMEX-OB)

Camera module (MARMEX-VB)

MARY-VBは8ビット/フレーム

しかしこのモジュールはそれぞれに SPIの設定が違います

プログラム

MARMEX-OBインスタンス

SPIインスタンス

9ビット/フレームに設定

```
MARMEX_OB_oled(  
  PinName mosi,  
  PinName sclk,  
  PinName cs,  
  PinName rst,  
  PinName power_pin  
) :  
  NokiaLCD( mosi, sclk, cs, rst, NokiaLCD::LCD6100 ),  
  _power_pin( power_pin )  
{  
  ...  
  _spi.format( 9 );  
  _spi.frequency( SPI_FREQUENCY );  
  ...  
}
```

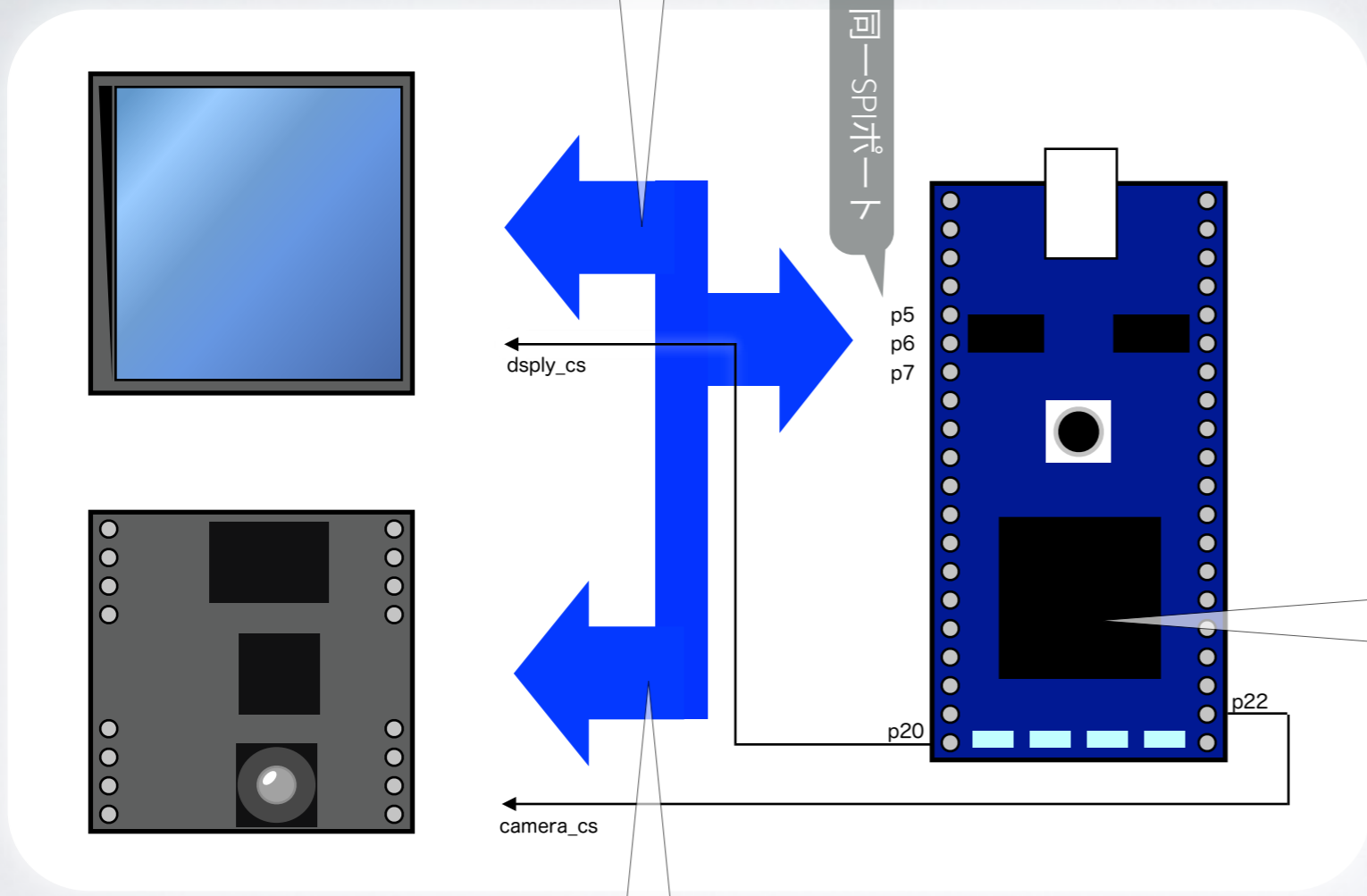
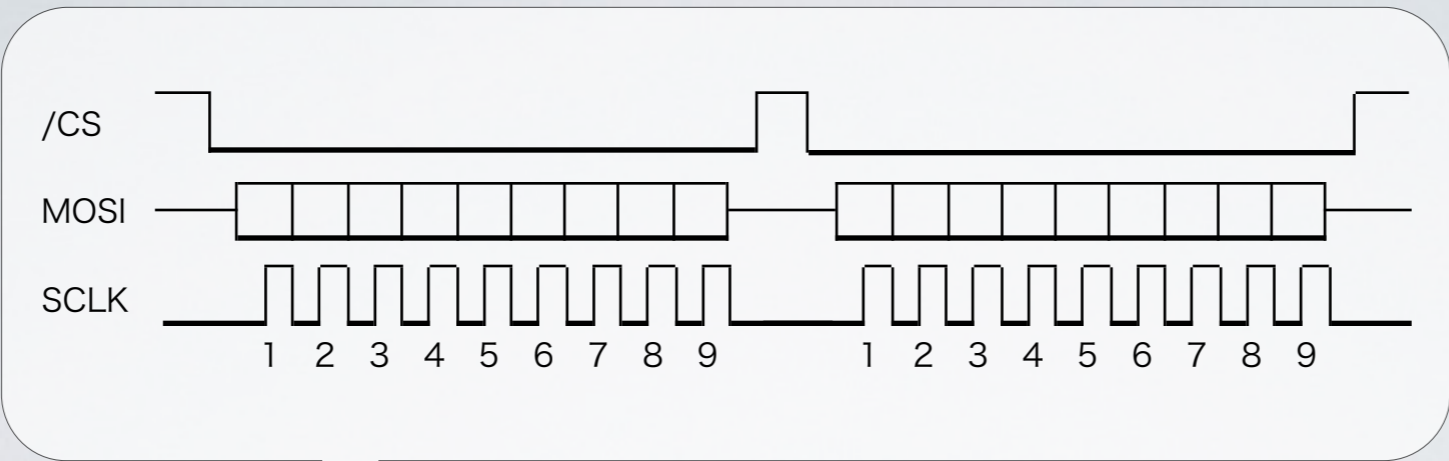
MARMEX-VBインスタンス

SPIインスタンス

8ビット/フレームに設定

```
MARMEX_VB::MARMEX_VB(  
  PinName SPI_mosi,  
  PinName SPI_miso,  
  PinName SPI_sck,  
  PinName SPI_cs,  
  PinName cam_reset,  
  PinName I2C_sda,  
  PinName I2C_scl  
) :  
  _spi( SPI_mosi, SPI_miso, SPI_sck ),  
  _cs( SPI_cs ),  
  _reset( cam_reset ),  
  _i2c( I2C_sda, I2C_scl )  
{  
  ...  
  _spi.format( 8 );  
  _spi.frequency( SPI_FREQUENCY );  
  ...  
}
```

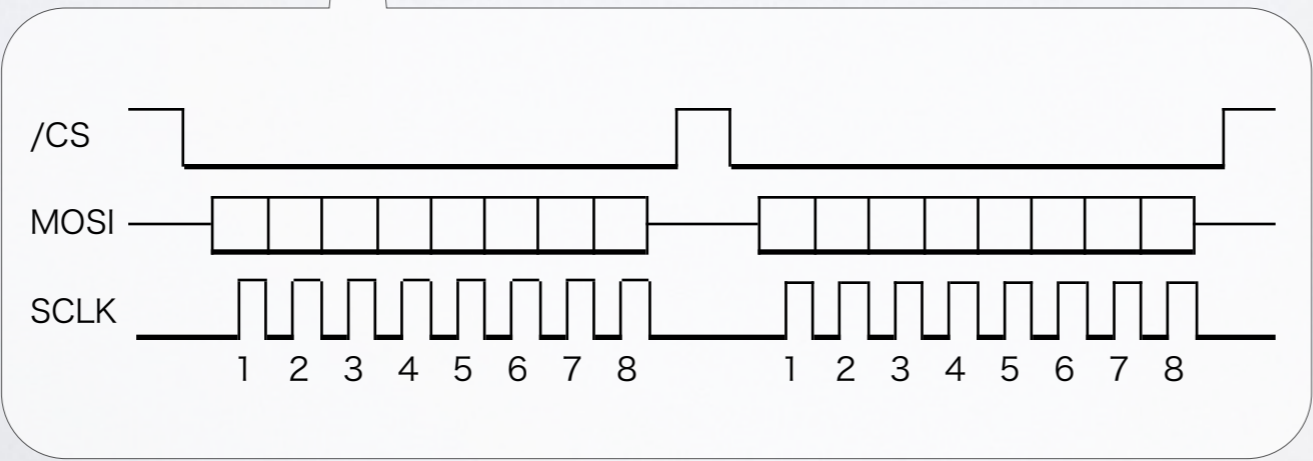
このような同一ピンに対するインターフェースでもインスタンスを別に持つことによって、それぞれのクラス内で「_spi.write()」で書き出しを行うと、自動的にハードが切り替わって9ビット/フレーム、8ビット/フレームの転送が行われる



```

SPI          disply_spi( p5, p6, p7 );
SPI          camera_spi( p5, p6, p7);
DigitalOut  disply_cs( p20 );
DigitalOut  camera_cs( p22 );
...
disply_spi.format( 9 );
camera_spi.format( 8 );
...
disply_cs = 0;
disply_spi.write( data_for_disply );
disply_cs = 1;
...
camera_cs = 0;
camera_spi.write( data_for_camera );
camera_cs = 1;
...

```

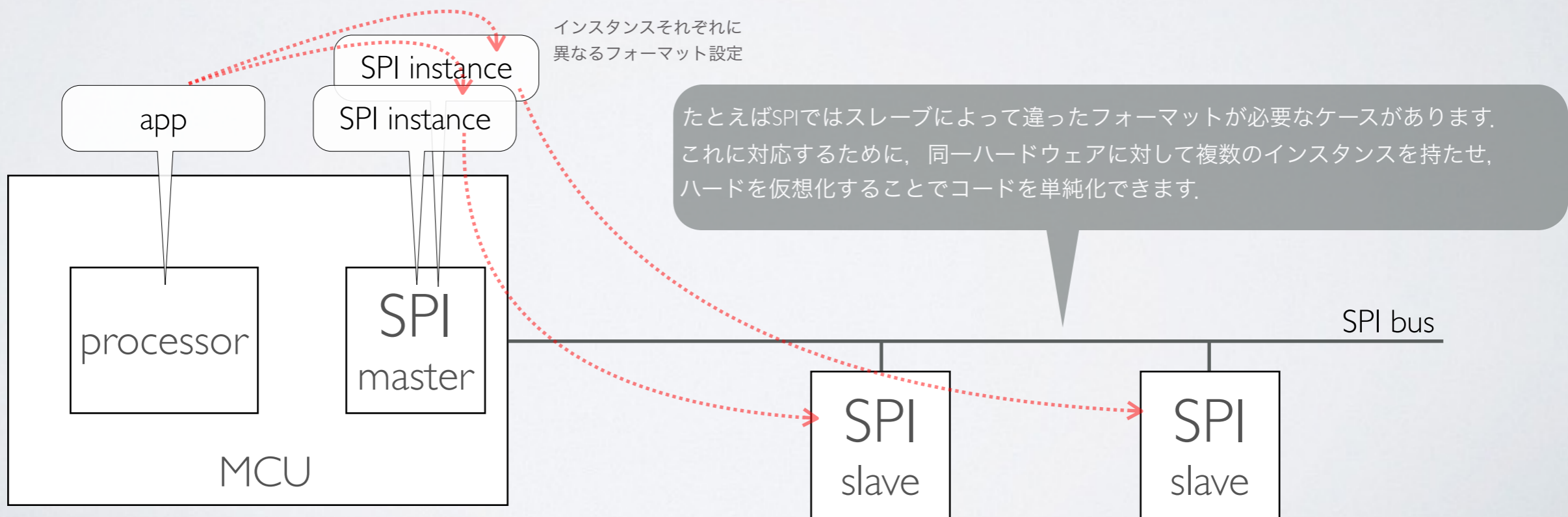
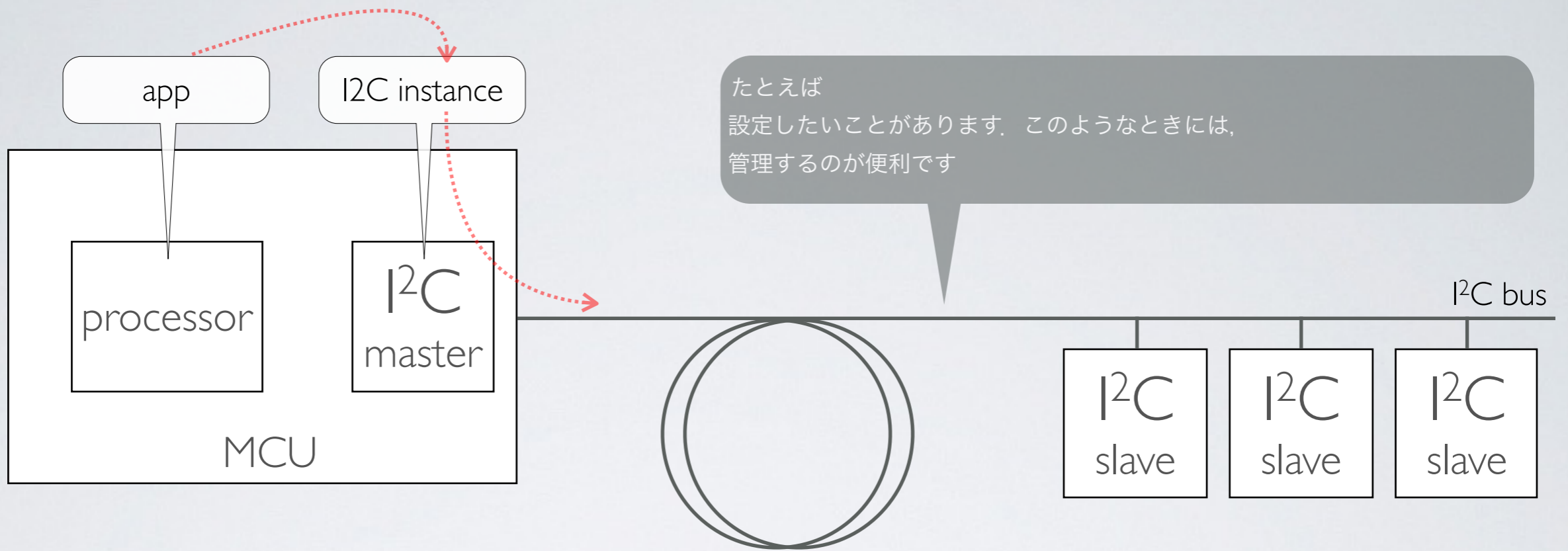


MARMEX-VBのデモプログラムを単純化した例。
別のインスタンスを作っておけば、同一ポートでもSDK内部でその設定に合わせたアクセスが行われます



前ページのコードを実行してみた様子

http://developer.mbed.org/users/okano/code/multiple_instance_for_SPI/



参考 (2)

- 公開用リポジトリ
 - 開発用と分けたほうがいいかもしれません
- 公開すると、全ての履歴(コミット)が公開されます
 - 後から公開バージョンを追っていく
 - 公開バージョンはこのうちのいくつかだけなのに「バグ付きのまま公開して更新ばかりしている」なんて揶揄されます
- 逆に開発ステップを含めて公開したい場合には分ける必要は無いでしょう

The screenshot shows the ARM mbed developer website interface. At the top, the URL is 'developer.mbed.org'. The page title is 'ARM mbed'. Below the title, there is a search bar and a breadcrumb trail: 'Users » okano » Code » MARY_CAMERA_Hello'. The user profile 'Tedd OKANO / MARY_CAMERA_Hello' is visible. The project name is 'mary camera library test harness'. Below that, the dependencies are listed: 'MARY_CAMERA', 'NokiaLCD', and 'mbed'. There are navigation buttons for 'Home', 'History', 'Graph', 'API Documentation', 'Wiki', 'Pull Requests', and 'Admin settings'. The main content is a 'Revision graph' showing a list of commits. Each commit entry includes a hash, a description, a date, and the author. The commits are listed in descending order of time, starting from the most recent at the top. The graph shows a linear sequence of commits, with some branches indicated by colored lines.

Revision Hash	Description	Date	Author
32:bf038420f809	test code	24 Mar 2014	Tedd OKANO
31:5ab6567aba69	camera reset implemented	22 Mar 2014	Tedd OKANO
30:eaab2c4bbe22	test code	22 Mar 2014	Tedd OKANO
29:a026f42d0498	test code	22 Mar 2014	Tedd OKANO
28:ef40d40b5d39	test code for contouring problem	13 Mar 2014	Tedd OKANO
27:625e40d50c7b	resolution change (by 12 registers overwriting) demo	12 Mar 2014	Tedd OKANO
26:4e4f954a2e2a	size change test added	12 Mar 2014	Tedd OKANO
25:185159c780d6	test code for BMP picture and colorbar	11 Mar 2014	Tedd OKANO
24:c74b706c25d6	colorbar demo	11 Mar 2014	Tedd OKANO
23:8471197d3096	size changed enabled	11 Mar 2014	Tedd OKANO
22:d5e24ab4afb7	test code	11 Mar 2014	Tedd OKANO
21:a2ac746dd516	1st frame saving into mbed storage if the mbed is blue or yellow	10 Mar 2014	Tedd OKANO
20:3a8a4fc846dd	camera register initialization array row-col swapped	10 Mar 2014	Tedd OKANO
19:b8c58b2298f2	code cleaning	08 Mar 2014	Tedd OKANO
18:2fbad209b31a	code for noise test	06 Mar 2014	Tedd OKANO
17:899406e350f0	line data transfer data order fix for Cortex-M0(+)	06 Mar 2014	Tedd OKANO
16:fa1bd83e34b0	prototype for yellow-mbed and tragi-ARM-boards	06 Mar 2014	Tedd OKANO
15:c81a197f485b	camera status detection by whole data byte	05 Mar 2014	Tedd OKANO
14:cc83bb625f92	ready signal triggered	05 Mar 2014	Tedd OKANO
13:52f7b692e00d	unnecessary waits are removed. comment added on class lib	18 Feb 2014	Tedd OKANO
12:6ddd07d59c55	test code	18 Feb 2014	Tedd OKANO
11:149993faf2be	bmp 32bit format file save function added	18 Feb 2014	Tedd OKANO
10:3c8c9569377	bmp file saving function added	18 Feb 2014	Tedd OKANO
9:68408189efdc	cleaner code	17 Feb 2014	Tedd OKANO
8:6d792029ff10	mary camera library test harness	17 Feb 2014	Tedd OKANO
7:380026dd09fd	the line noise occurs	17 Feb 2014	Tedd OKANO
6:f5b4e088087b	working but still having line noise	17 Feb 2014	Tedd OKANO
5:187400676e34	test 0 "doesn't work"	15 Feb 2014	Tedd OKANO
4:5e1828a8e238	c++ lib version "doesn't work"	14 Feb 2014	Tedd OKANO
3:152362acd181	simple prototype #1	14 Feb 2014	Tedd OKANO
2:2e03fc4f485b	it's working!	14 Feb 2014	Tedd OKANO
1:ce27bc7b44d4	not working correctly. it seems frame rate is OK but color is bad	14 Feb 2014	Tedd OKANO
0:1062142e5718	not working correctly	13 Feb 2014	Tedd OKANO

Revision graph

Commit Hash	Description	Date	Author
32:bf038420f809	test code	24 Mar 2014	Tedd OKANO
31:5ab6567aba69	camera reset implemented	22 Mar 2014	Tedd OKANO
30:aaab2c4bbe22	test code	22 Mar 2014	Tedd OKANO
29:a026f42d0498	test code	22 Mar 2014	Tedd OKANO
28:ef40d40b5d39	test code for contouring problem	13 Mar 2014	Tedd OKANO
27:625e40d60c7b	resolution change (by 12 registers overwriting) demo	12 Mar 2014	Tedd OKANO
26:4e4f954a2e2a	size change test added	12 Mar 2014	Tedd OKANO
25:185159c780d6	test code for BMP picture and colorbar	11 Mar 2014	Tedd OKANO
24:c74b706c25d6	colorbar demo	11 Mar 2014	Tedd OKANO
23:8471197d3096	size changed enabled	11 Mar 2014	Tedd OKANO
22:d5e24ab4afb7	test code	11 Mar 2014	Tedd OKANO
21:a2ac746dd516	1st frame saving into mbed storage if the mbed is blue or yellow	10 Mar 2014	Tedd OKANO
20:3a8a4fc846dd	camera register initialization array row-col swapped	10 Mar 2014	Tedd OKANO
19:b8c58b2298f2	code cleaning	08 Mar 2014	Tedd OKANO
18:2fbad209b31a	code for noise test	06 Mar 2014	Tedd OKANO
17:899406e350f0	line data transfer data order fix for Cortex-M0(+)	06 Mar 2014	Tedd OKANO
16:fa1bd83e34b0	prototype for yellow-mbed and tragi-ARM-boards	06 Mar 2014	Tedd OKANO
15:c81a197f4f5b	camera status detection by whole data byte	05 Mar 2014	Tedd OKANO
14:cc83bb625f92	ready signal triggered	05 Mar 2014	Tedd OKANO
13:52f7b692e00d	unnecessary waits are removed. comment added on class lib	18 Feb 2014	Tedd OKANO
12:6dd07d59c55	test code	18 Feb 2014	Tedd OKANO
11:149993faf2be	bmp 32bit format file save function added	18 Feb 2014	Tedd OKANO
10:3c8fc9569377	bmp file saving function added	18 Feb 2014	Tedd OKANO
9:68408189efde	cleaner code	17 Feb 2014	Tedd OKANO
8:6d792029f110	mary camera library test harness	17 Feb 2014	Tedd OKANO
7:389326d129d1	the line noise occurs	17 Feb 2014	Tedd OKANO
6:5b4e080087b	working but still having line noise	17 Feb 2014	Tedd OKANO
5:187400676e34	test 0 "doesn't work"	15 Feb 2014	Tedd OKANO
4:5e1828a8e238	c++ lib version "doesn't work"	14 Feb 2014	Tedd OKANO
3:152362acd181	simple prototype #1	14 Feb 2014	Tedd OKANO
2:2e03fc4f485b	it's working!	14 Feb 2014	Tedd OKANO
1:ce27bc7b44d4	not working correctly. it seems frame rate is OK but color is bad		

コードを別プログラムに移して公開

https://developer.mbed.org/users/okano/code/MARY_CAMERA_Hello/graph

Revision graph

Commit Hash	Description	Date	Author
8:86aa677a68b	SPI-FIFO operation option added	20 Jun 2014	NXP fan (in Japan)
7:125538c50c22	optimized SPI : FIFO	19 Jun 2014	NXP fan (in Japan)
6:b61b876d50a8	baseboard type selector added	18 Jun 2014	NXP fan (in Japan)
5:ac4f0c5d1c6f	optimization done for SPI operation: ChipSelect signal kept asserted during line data transfer.	16 Jun 2014	NXP fan (in Japan)
4:1d15e17d14f9	unnecessary library has been removed	15 Jun 2014	NXP fan (in Japan)
3:34e54e8a3051	update to use latest MARMEX_VB library	09 Jun 2014	NXP fan (in Japan)
2:a35c919aa9c2	MARMEX_VB test application program. This application works on "mbed NXP LPC1768" only. ; This application expects to have the MARMEX_VB module on a "M	09 Jun 2014	NXP fan (in Japan)
1:dbe2dc31542d	MARMEX-VB test version 0.2 (mbed_NXP_LPC1768 optimization added)	09 Jun 2014	NXP fan (in Japan)
0:343c01965543	version 0.1	06 Jun 2014	NXP fan (in Japan)

MARMEX-VBは当初、
個人のアカウント内で作成

その後、
会社の公式アカウントから
CQ出版チームとして公開

http://developer.mbed.org/teams/CQ-Publishing/code/MARMEX_VB_test/graph

developer.mbed.org

Platforms Components Handbook Cookbook Code Questions Forum Dashboard(5) Compiler

ARM mbed

Users - okano - Code - ika_shouyu_poppoyaki

Tedd OKANO / ika_shouyu_poppoyaki

this transfers data (which is stored in "bin" file in mbed storage) into LPC1114, LPC81x and LPC82x internal flash memory through ISP.

Dependencies: mbed MODSERIAL

Home History Graph API Documentation Wiki Pull Requests Admin settings

Revision graph

43:c7d6d62abc14	version info updated :) default tip	16 days ago, by Tedd OKANO
42:2b40666d8177	LPC82x series supported (only 824 is tested)	16 days ago, by Kazuki Yamamoto
41:74b0ff21098f	enabled to handle <0x300 (768) bytes data file	19 Nov 2013, by Tedd OKANO
40:615dc8275648	ver 0.98 : cleaned-up	29 Sep 2013, by Tedd OKANO
39:f68f9fa1e88e	ver 0.98 : suppressed debug message in default setting. it improves speed of writing and verifying	29 Sep 2013, by Tedd OKANO
38:cb95bfe0546a	ver 0.97 : can verify non 4*N size binary.; can write full 32768 bytes for LPC1114FN28, can write 4096 bytes for LPC810.	27 Sep 2013, by Tedd OKANO
37:4cd12c9c1cc2	correcting version number in message	26 Sep 2013, by Tedd OKANO
36:44a2eac67549	to use latest mbed-lib	26 Sep 2013, by Tedd OKANO
35:0b434e44a49	added: warning/error by compile option settings	26 Sep 2013, by Tedd OKANO
34:eaca33d3e632	[1]: fix: verification function was having a bug. 3/4 of the code was having chance of verification fail.; [2]: "ENABLE_WRITING" option added (see "	26 Sep 2013, by Tedd OKANO
33:ce9ff4cbf09	options of "ENABLE_VERIFYING" and "CHECK_CRP_CODE" are added in _user_settings.h.	25 Sep 2013, by Tedd OKANO
32:3700d5d4e18	CRP checking code added (not tested yet)	24 Sep 2013, by Tedd OKANO
31:1a4d59d7bd72	cleaner code :)	20 Sep 2013, by Tedd OKANO
30:e0d7524661ca	** version 0.95. redundant code and files are removed	20 Sep 2013, by Tedd OKANO
29:96e28bc1bd99	parameter added to writing and verifying functions to report transferred size	20 Sep 2013, by Tedd OKANO
28:689c3880e0e4	made function returns error. LED1 and LED2 assigned to toggle by TX and RX	20 Sep 2013, by Tedd OKANO
27:2b5c1eb39bb5	(cannot be build) an a precess of clean-up	19 Sep 2013, by Tedd OKANO
26:a63e73885b21	code is still dirty but it works. I hope I will have chance to clean up some day...	13 Sep 2013, by Tedd OKANO
25:33cb5ad8ae24	dividing code into modules	13 Sep 2013, by Tedd OKANO
24:9830b4f1207b	deviding code into modules	13 Sep 2013, by Tedd OKANO
23:017f306cf3ca	dividing code into modules	13 Sep 2013, by Tedd OKANO
22:bd98a782fba6	dividing code into modules	13 Sep 2013, by Tedd OKANO
21:e149d0bdb44a	"command_utilities" module made	13 Sep 2013, by Tedd OKANO
20:98d7b5878e3e	verification for uencode type device (LPC1114) supported	13 Sep 2013, by Tedd OKANO
19:7a7381e78025	verification function is available for "BINARY transfer" device (LPC810)	12 Sep 2013, by Tedd OKANO
18:b401da200216	some comments edited	09 Sep 2013, by Tedd OKANO
17:339f40a1467	version number updated to "0.7"	09 Sep 2013, by Tedd OKANO
16:cac2348cfcfb	mbed goes into "serial through mode" after ISP writing finished	09 Sep 2013, by Tedd OKANO
15:051ca36c64b	wait removed before "reset_target[NO_ISP_MODE]"	28 Aug 2013, by Tedd OKANO
14:a7b9f74fb856	"AUTO_PROGRAM_START" option added.; Header comment modified in main.cpp	28 Aug 2013, by Tedd OKANO
13:60995bf8b2c7	corrected a bug of reset_target() function	28 Aug 2013, by Tedd OKANO


Repository toolbox

- Import this program
- Export to desktop IDE
- Build repository
- Send Pull Request from here
- Make featured
- Following
- Clone repository to desktop: hg clone https://okano@de

Repository details

- Type: Program
- Created: 24 Aug 2013
- Imports: 273
- Forks: 1
- Commits: 44
- Dependents: 0
- Dependencies: 2
- Followers: 20

Software licencing information



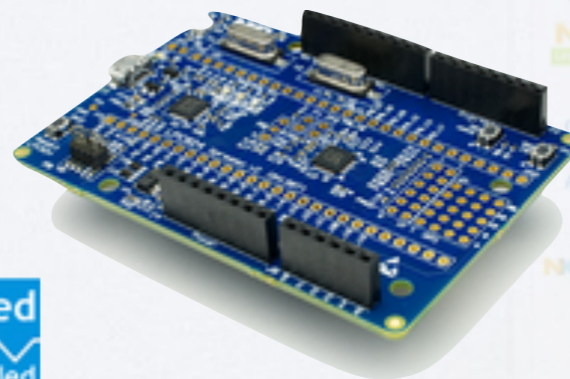
The code in this repository is Apache licensed.

- こちらは開発用リポジトリをそのまま公開してしまった例
- 全てのコード変更→コミットのステップが見えてしまっており各公開バージョンがどれだったかが判らなくなっています
- 細かいコミットも表示されてしまっており、後からこれを見た人に「まともに検証すら行っていないコードを公開・訂正を繰り返すとはナニゴトか！」と怒られたことも(^^;
- その一方で、ISPプログラムを独自に開発されている方から「シリアル上のプロトコルを確認するのに、履歴の初めから追うことによってよく理解出来ました」との声も

http://developer.mbed.org/users/okano/code/ika_shouyu_poppoyaki/graph

参考 (3)

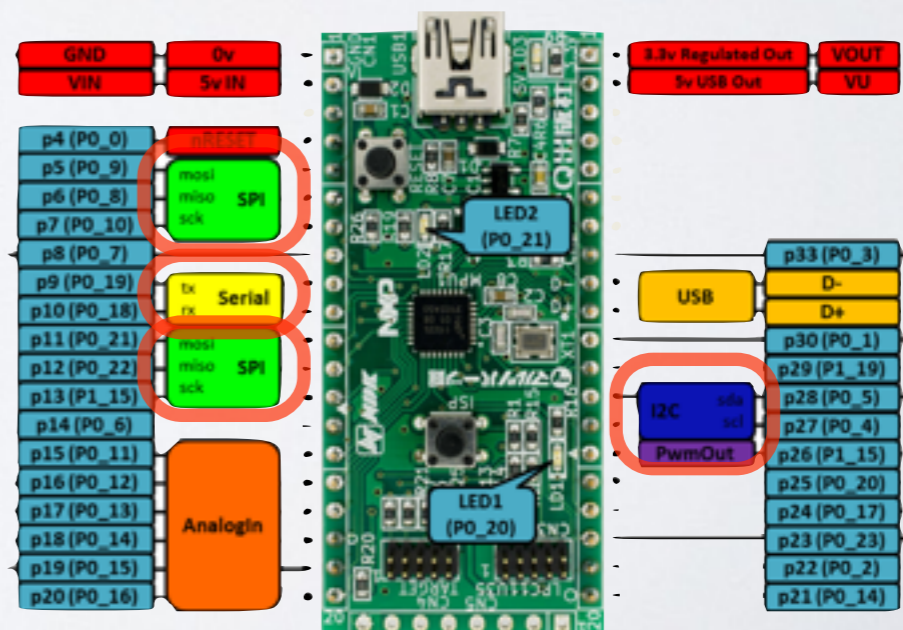
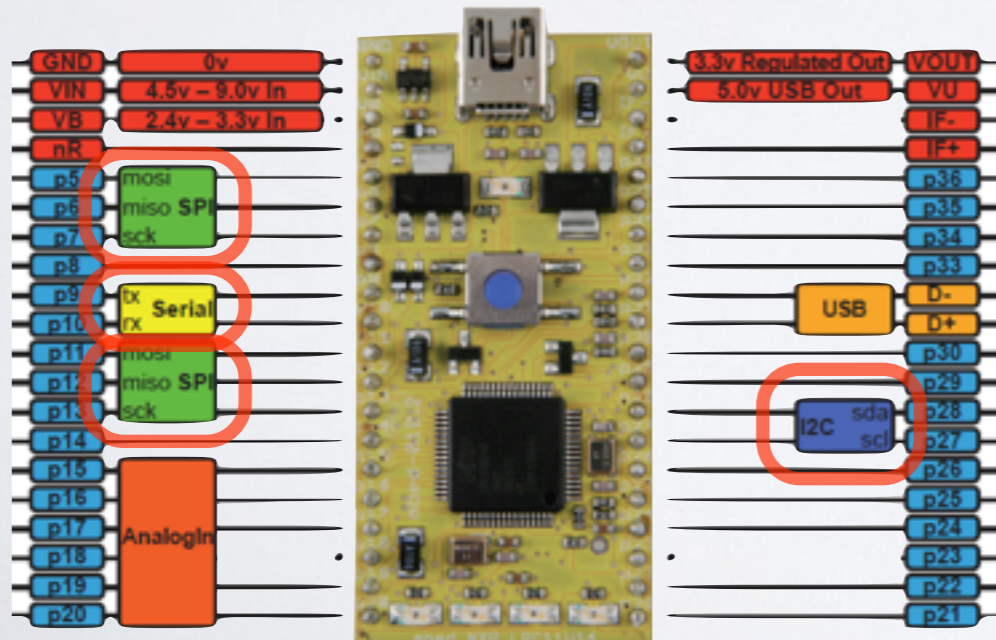
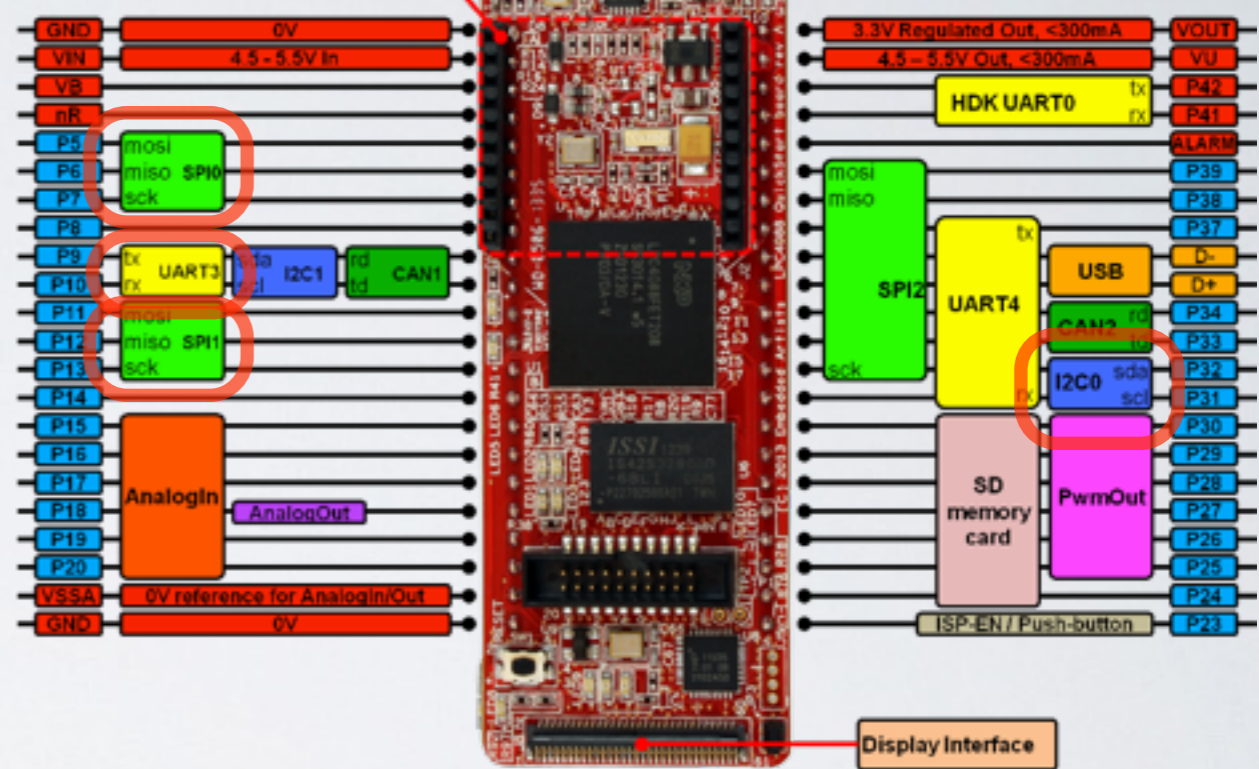
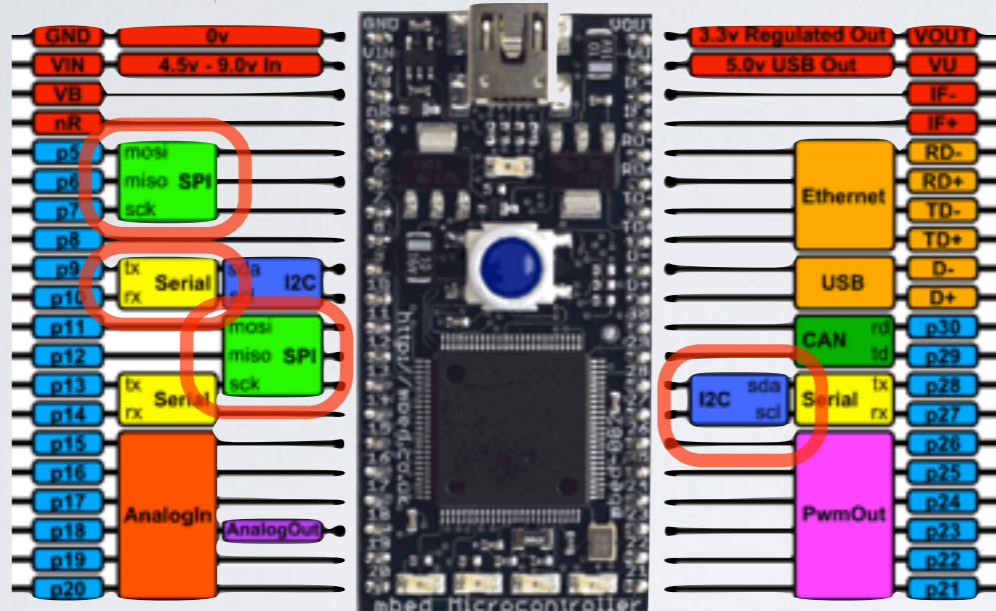
- どのプラットフォームでテストしておくべきか
- mbed LPC1768 (mbedのリファレンス)
- あとは好みで (^_^;
- トラ技ARMライタも！



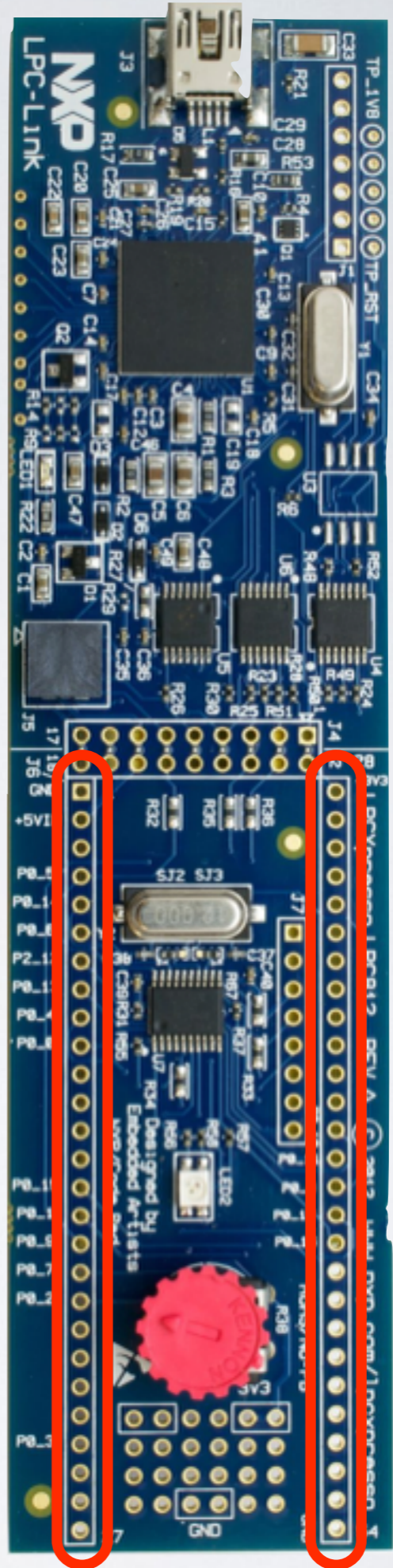
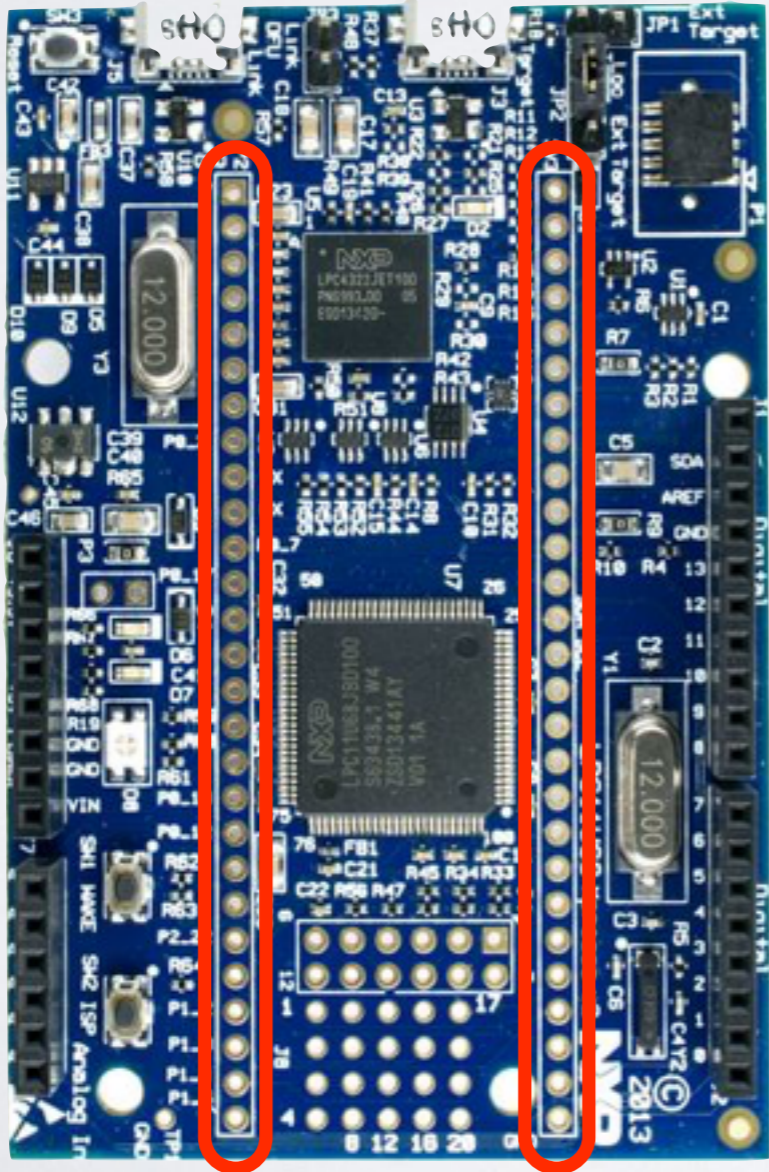
参考 (4)

- サンプルコードではどのピンを使うか？

40pin, LPCXpresso型のピンを持つ基板は、ピン機能の基本配置は決まっている。
 公約数的なピン選択をしておけば、すぐに使えるプログラムとなる
 Arduino配列のものはそれに則して..



ちなみに..
LPCXpressoシリーズのピンヘッダ用端子も最大限、
青mbedに合わせて配置されています



参考 (5)

- (MCUの)レジスタレベルの最適化
- サンプルコードに使うピンを，ターゲットによって切り替える

```

#if ( LINE_READ_OPT == USING_SSP_FIFO )

#define FIFO_DEPTH 4

#if defined( SSP_AUTO_SELECTION )
    #if defined( TARGET_MBED_LPC1768 )
        #define SPI_PORT_SELECTOR LPC_SSP1
    #elif defined( TARGET_LPC11U35_501 ) || defined( TARGET_LPC11U24_401 )
        #define SPI_PORT_SELECTOR LPC_SSP0
    #endif
#elif defined( SSP_USE_SSP0 )
    #define SPI_PORT_SELECTOR LPC_SSP0
#elif defined( SSP_USE_SSP1 )
    #define SPI_PORT_SELECTOR LPC_SSP1
#else
    #error when using FIFO option for the optimization, choose one of definition SSP_USE_SSP0 ..
#endif // #if defined( SSP_AUTO_SELECTION )

char reg = COMMAND_READ | CAMERA_DATA_REGISTER | COMMAND_ADDR_INCREMENT;
int n;

if ( _read_order_change ) {

    _cs = 0;

    for(n = FIFO_DEPTH; n > 0; n--) {
        SPI_PORT_SELECTOR->DR = reg;
    }

    do {
        while (!(SPI_PORT_SELECTOR->SR & 0x4));
        *p = (SPI_PORT_SELECTOR->DR & 0xFF);

        if (n++ < (n_of_pixels << 1) - FIFO_DEPTH)
            SPI_PORT_SELECTOR->DR = reg;

        while (!(SPI_PORT_SELECTOR->SR & 0x4));
        *p++ |= (SPI_PORT_SELECTOR->DR << 8);

        if (n++ < (n_of_pixels << 1) - FIFO_DEPTH)
            SPI_PORT_SELECTOR->DR = reg;

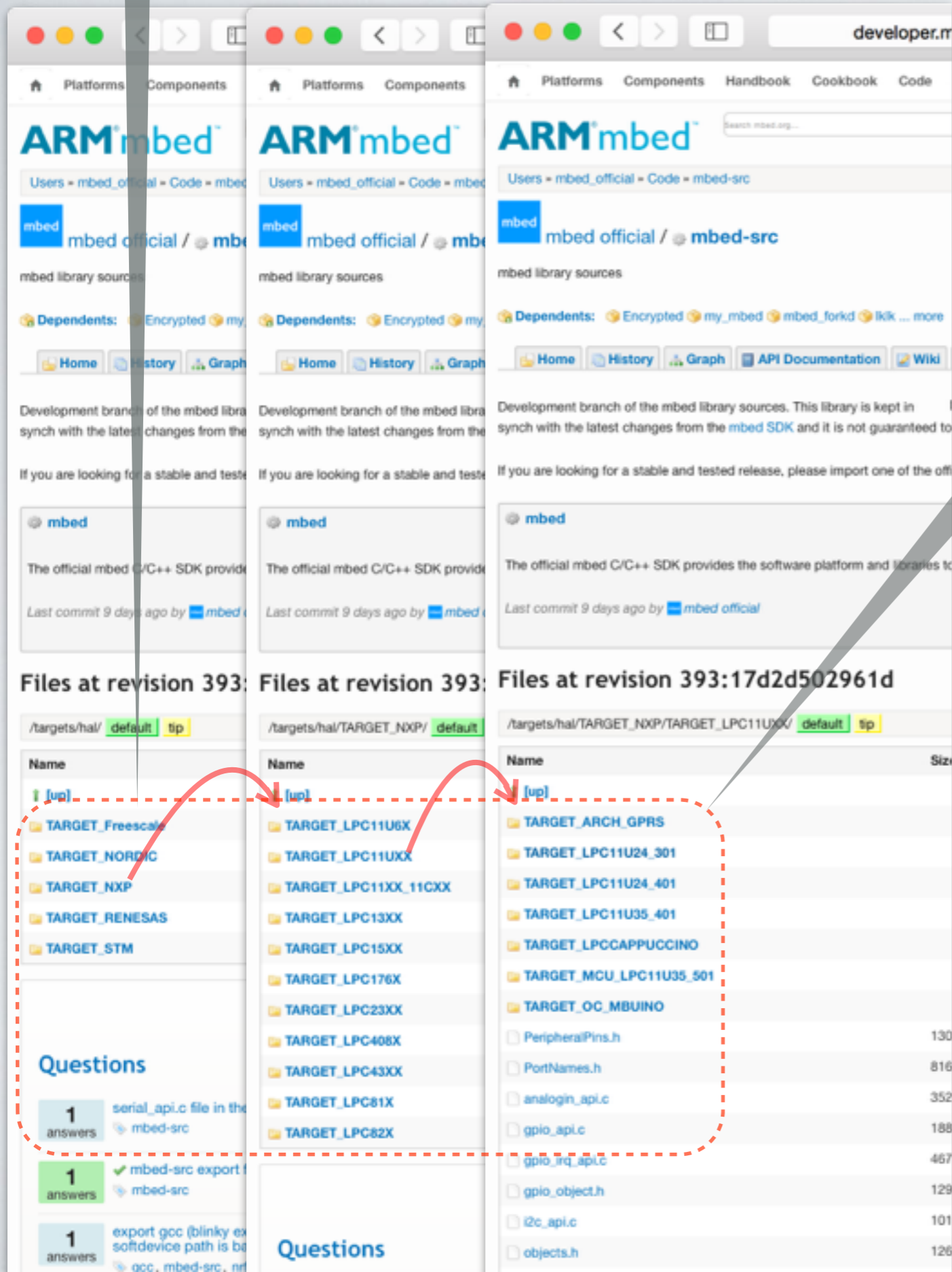
    } while(n < (n_of_pixels << 1));

    _cs = 1;

```

ターゲットによってコードを切り替える

CMSIS定義のレジスタにアクセス
(アドレス直叩きも可能ですが、お行儀としては
こちらのほうがマシでしょう)



ターゲットを識別する定義済みのシンボル名は
mbed-srcのコード中のフォルダ名「TARGET_〇〇」
が使えます

たとえば
NXPのMCUであるかどうかを識別したいなら
「TARGET_NXP」
LPC11Uxxシリーズであることを識別するなら
「TARGET_LPC11UXX」
LPC11U24/401であることを認識するなら
「TARGET_LPC11U24_401」
のようなシンボル名を使えます

```

1 #include "mbed.h"
2
3 #if defined( TARGET_NXP )
4 #warning TARGET_NXP detected
5 #endif
6
7 #if defined( TARGET_LPC176X )
8 #warning TARGET_LPC176X detected
9 #endif
10
11 #if defined( TARGET_MBED_LPC1768 )
12 #warning TARGET_MBED_LPC1768 detected
13 #endif
14
15 #if defined( TARGET_LPC11UXX )
16 #warning TARGET_LPC11UXX detected
17 #endif
18
19 #if defined( TARGET_LPC11U24_401 )
20 #warning TARGET_LPC11U24_401 detected
21 #endif
22
23
24 DigitalOut myled(LED1);
25
26 int main() {
27     while(1) {
28         myled = 1;
29         wait(0.2);
30         myled = 0;
31         wait(0.2);
32     }
33 }
34

```

試してみる方法

ここでは試しに「mbed LPC11U24」をターゲットに選択

#if define (○○○)を使って検出

この「#if」に引っかければコンパイラがwarningとしてメッセージを表示してくれます

コンパイルしてみると...
ちゃんと定義されていたことがわかりました (^ ^)

Compile output for program: target_checker Errors: 0 Warnings: 3 Infos: 1

Description	Error Number
#warning directive: TARGET_NXP detected "#warning TARGET_NXP detected	
#warning directive: TARGET_LPC11UXX detected "#warning TARGET_LPC11L	
#warning directive: TARGET_LPC11U24_401 detected "#warning TARGET_LP	
Success!	

http://developer.mbed.org/users/okano/code/target_checker/

参考 (6)

- 協業しましょう
 - プルリクエストを出しましょう
 - 他人様のコードを改良したら→プルリクエスト
 - オリジナルのリポジトリにマージしてもらおう



Tedd OKANO / 📦 [ika_shouyu_poppoyaki](#)

this transfers data (which is stored in "bin" file in mbed storage) into LPC1114, LPC81x and LPC82x internal flash memory through ISP.

Dependencies: mbed MODSERIAL

[Home](#)[History](#)[Graph](#)[API Documentation](#)[Wiki](#)[Pull Requests](#)[Admin settings](#)

Revision graph

- 43:c7d6d62abc14 version info updated ;) default tip 16 days ago , by 🐱 Tedd OKANO
- 42:2b40666d8177 LPC82x series supported (only 824 is tested) 16 days ago , by 👤 [Kazuki Yamamoto](#)
- 41:74b9ff21098f enabled to handle <0x300 (768) bytes data file 19 Nov 2013 , by 🐱 Tedd OKANO
- 40:615dc8275648 ver 0.98 : cleaned-up 29 Sep 2013 , by 🐱 Tedd OKANO
- 39:f68f9fa1e88e ver 0.98 : suppressed debug message in default setting. it improves speed of writing and verifying 29 Sep 2013 , by 🐱 Tedd OKANO
- 38:cb95bfe0546a ver 0.97: ; can verify non 4*N size binary.; can write full 32768 bytes for LPC1114FN28, can write 4096 bytes for LPC810 27 Sep 2013 , by 🐱 Tedd OKANO

プルリクエストを頂いて
マージした例

Revisions of program "test_LM75B_Hello"
Showing revisions of program "test_LM75B_Hello" and public repository at [okano/test_LM75B_Hello](#).

Commit Discard Changes Compare Switch Revert Me

Graph	Revisor	When	Who	Comment
	9	moments ago	nxpj0	default tip comment addi
	8	1 day, 1 hour ago	okano	to include published
	7	1 day, 1 hour ago	okano	comment added

Publish Repository

Publish repository
This will publish changes in "test_LM75B_Hello" to public repository at [okano/test_LM75B_Hello](#)
You can also Fork the repository and re-publish to a different URL.

Fork... **OK** Cancel

インポートしたプログラムに変更を加え、コミットすると自分のアカウント内に置かれたプログラムの履歴に自分の変更が足される

変更を加えたプログラムを公開。その公開ページを開くと「プルリクエストを送る」ボタンがあるのでクリック

Users = nxpj0 = Code = test_LM75B_Hello

NXP-Japan Demo-0 / test_LM75B_Hello

pull request test

Dependencies: [mbed](#) [test_LM75B](#)

Fork of test_LM75B_Hello by [Tedd OKANO](#)

Home History Graph API Documentation Wiki Pull Requests Admin settings

You can edit this area!

Download repository: [zip](#) [gz](#)

Edit repository homepage

Repository toolbox

[Import this program](#)

[Export to desktop IDE](#)

[Build repository](#)

[Send Pull Request from here](#)

[Make featured](#)

[Following](#)

Users = nxpj0 = Code = test_LM75B_Hello = Create Pull Request

NXP-Japan Demo-0 / test_LM75B_Hello

pull request test

Dependencies: [mbed](#) [test_LM75B](#)

Fork of test_LM75B_Hello by [Tedd OKANO](#)

Create a Pull Request

This pull request will be sent to Tedd OKANO for the program test_LM75B_Hello

http://developer.mbed.org/users/nxpj0/code/test_LM75B_Hell → http://developer.mbed.org/users/okano/code/test_LM75B_He

pull request test

pull request test

Send Pull Request Cancel

プルリクエストのタイトルとメッセージを書いて送信

Tedd OKANO / test_LM75B_Hello

example project to explain how to write a class library

Dependencies: [mbed](#) [test_LM75B](#)

Home History Graph API Documentation Wiki Pull Requests Admin settings

pull request test

pull request test

[Review](#)

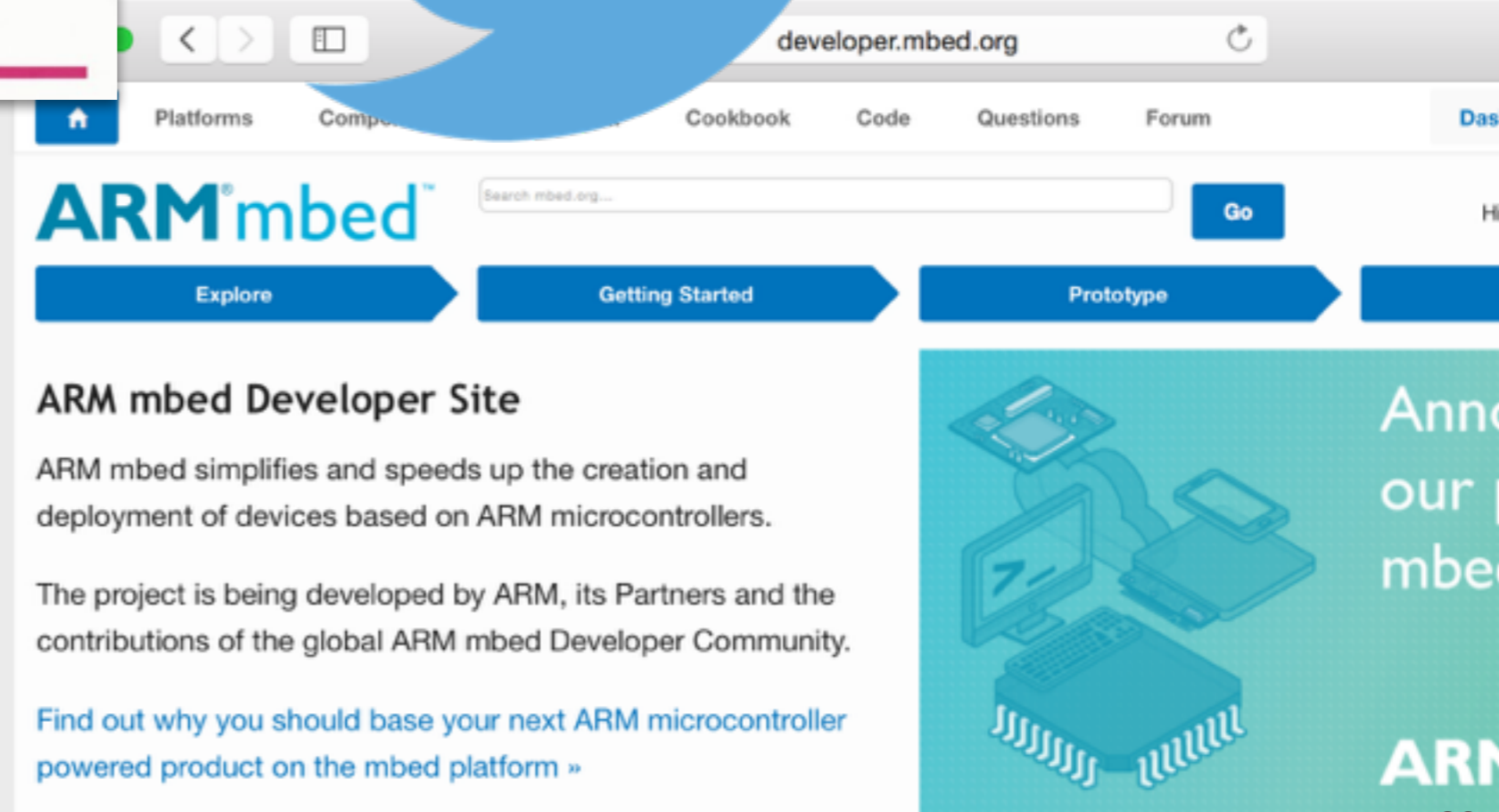
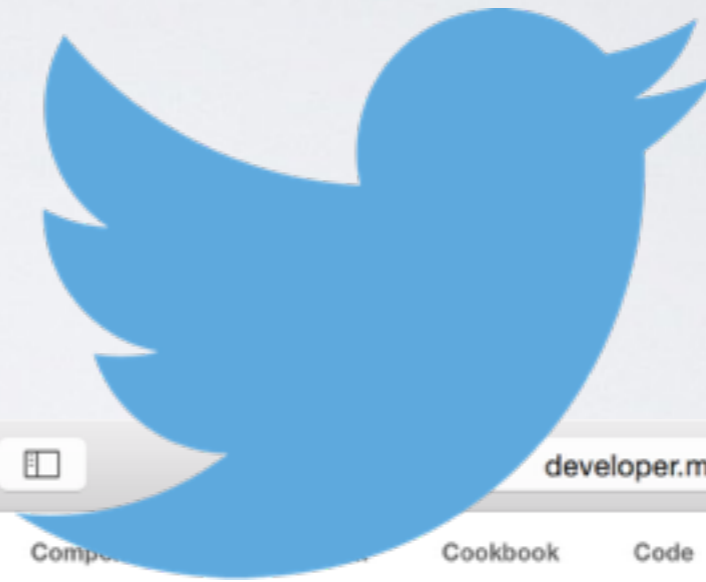
プルリクエストが元の公開者に届く

プルリクエストを受け取ったら
Reviewボタンを押してコードを確認します。
問題なければマージしましょう

The screenshot shows a GitHub pull request interface. At the top, the repository name is 'Tedd OKANO / test_LM75B_Hello' with a description 'example project to explain how to write a class library'. Below this are navigation tabs: Home, History, Graph, API Documentation, Wiki, Pull Requests, and Admin settings. The main content area shows a pull request titled 'pull request test' by user 'NXP-Japan Demo-0'. It includes a 'Review' button (highlighted by a callout) and a 'Close' button (highlighted by another callout). The pull request details show it was last updated 11 minutes ago and has 10 comments and 0 reviews. On the right, there are two sidebars: 'Repository toolbox' with buttons like 'Import this program', 'Export to desktop IDE', 'Build repository', 'Send Pull Request from here', 'Make featured', and 'Following'; and 'Repository details' showing statistics such as Type: Program, Created: a day ago, Imports: 2, Forks: 0, Commits: 9, Dependents: 0, Dependencies: 2, and Followers: 1. At the bottom, there is a 'Post a new comment' section with a text area, an 'Insert images or files' button, and 'Post comment' and 'Show preview' buttons.

Closeを押してしまうと「リクエスト棄却」に
なってしまいますので注意

参考 (7)



- 情報源

勉強会参加者に教えて頂いたんですが、この本だったのに、ひょっとすると絶版かもしれません。残念 (>_<)

参考 (8)

- どのようにしていいか、わからないとき
- →優れた例を探してみる

<https://developer.mbed.org/users/shintamainjp/>

The screenshot shows the profile page for Shinichiro Nakamura on the mbed.org developer platform. The page is titled "Shinichiro Nakamura" and includes a "Notebook" section with 62 pages. The notebook entries are:

- A simple wave recorder & player** (Last updated: 14 Apr 2012)
- 实用ライブラリシリーズ : Natural Tiny Shell (NT-Shell)** (Last updated: 22 May 2011)
- Practical library series : Natural Tiny Shell (NT-Shell)** (Last updated: 22 May 2011)
- StarBoard Orange - Expansion Board One: (Example No.2: Record thermistor values using Pachube)** (Last updated: 30 Apr 2011)
- ☆ボードオレンジ拡張基板 : 「活用事例2 : Pachubeを使って温度センサの値を記録しよう！」** (Last updated: 30 Apr 2011)
- Japan / Earthquake Donation** (Last updated: 15 Mar 2011)
- 实用ライブラリシリーズ : Anchor (小型GUIフレームワーク)** (Last updated: 04 Jan 2011)
- Practical library series : Anchor (A tiny GUI framework)** (Last updated: 04 Jan 2011)
- Backup your published programs and libraries using a** (Last updated: 31 Dec 2010)

The right sidebar shows user details for Shinichiro Nakamura: 7502 points, 41 published programs, 23 published libraries, 62 notebook pages, and joined on 27 May 2011.

参考 (9)

- mbed-SDKの中身を知りたいときは？

The screenshot shows the ARM mbed developer website. The main navigation includes Platforms, Components, Handbook, Cookbook, Code, Questions, and Forum. The user is logged in as 'okano'. The page displays the 'mbed official / mbed-src' repository. It includes a 'Repository toolbox' with options like 'Import this library', 'Export to desktop IDE', and 'Follow'. A 'Repository details' section shows statistics: Type: Library, Created: 20 Nov 2012, Imports: 6916, Forks: 21, Commits: 392, Dependents: 61, Dependencies: 0, Followers: 234, Issues: 6. A 'Questions' section lists several user queries and their answers. At the bottom, there is a footer with links for 'mbed', 'blog', 'we're hiring!', 'support', 'service status', 'privacy policy', 'terms and conditions', and 'Language: en ja es de'.

http://developer.mbed.org/users/mbed_official/code/mbed-src/

The screenshot shows the GitHub repository page for 'mbedmicro / mbed'. The repository has 2,652 commits, 1 branch, 27 releases, and 78 contributors. The 'Code' tab is active, showing a commit history table with columns for commit hash, author, message, and time. The commit history includes entries like '[RZ_A1H] Remove target-specific check from build system' by bogdann. Below the commit history, there is a 'README.md' section with the title 'mbed SDK'. The README text describes the mbed Software Development Kit (SDK) as a C/C++ microcontroller software platform. It mentions that the SDK is licensed under the permissive Apache 2.0 license and is designed to provide hardware abstraction. A 'Documentation' section lists links for 'Tools: how to setup and use the build system', 'mbed library internals', and 'Adding a new target microcontroller'. The 'Supported Microcontrollers and Boards' section is also visible at the bottom.

<https://github.com/mbedmicro/mbed>

変更履歴

- **Version 1.0 (07-Nov-2014)**
 - 正式公開版
- **Version 1.1 (08-Nov-2014)**
 - 59ページを追加：「公開後の変更は可能か？」ページ
 - 65ページを変更：コンポーネント・ページは作成者以外も編集可能であることを追記
 - 86, 87ページを追加：「ターゲット設定を識別するための定義済みの語」
 - 92ページを変更：書籍の入手性情報を追記
 - 96ページを追加：変更履歴ページ