



不可能への挑戦

株式会社日昇テクノロジー

低価格、高品質が不可能？

日昇テクノロジーなら可能にする

## ARM Cortex-M3 多機能通信 STM32F103ZET6

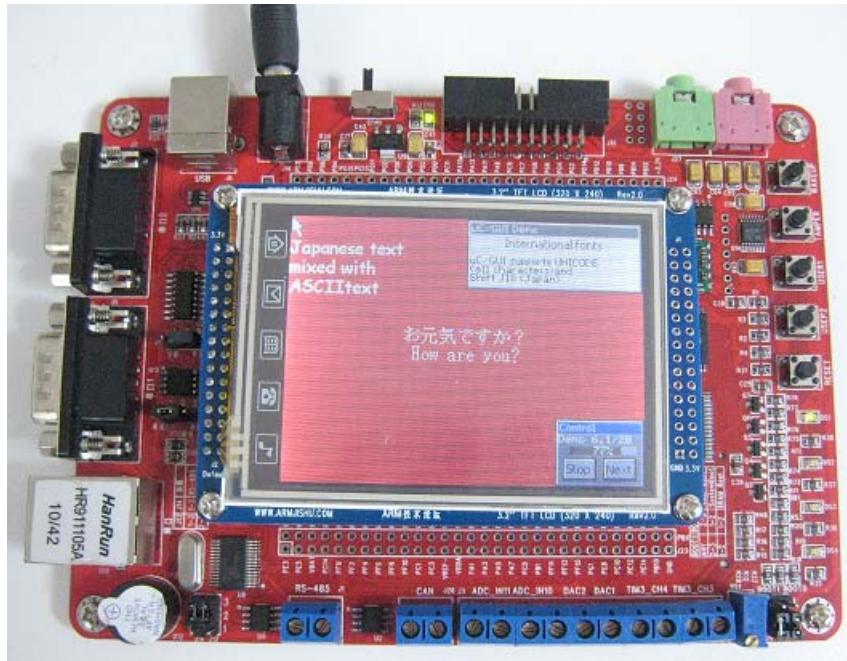
マニュアル

株式会社日昇テクノロジー

<http://www.csun.co.jp>

info@csun.co.jp

2011/12/01



copyright@2011

## • 修正履歴

NO	バージョン	修正内容	修正日
1	Ver1.0	新規作成	2011/07/26
2	Ver1.1	2. 4G 無線通信インタフェースのピン配列変更 -修正箇所 4. 2. 16 節 (2. 4G 無線モジュールと 直結できる様に変更しました)	2011/12/01

※ この文書の情報は、文書を改善するため、事前の通知なく変更されることがあります。最新版は弊社ホームページからご参照ください。

「<http://www.csun.co.jp>」

※ (株)日昇テクノロジーの書面による許可のない複製は、いかなる形態においても厳重に禁じられています。



## 目次

第一章 通信機能搭載-STM32F103ZEキット概要 .....	5
1.1 主な特徴 .....	5
第二章 回路の説明 .....	6
2.1 STM32F103ZET6 .....	6
2.2 Nor Flash.....	6
2.3 Nand Flash.....	6
2.4 SRAM.....	7
2.5 SPI Flash.....	7
2.6 SDカードインタフェース .....	8
2.7 I2C EEPROM.....	8
2.8 Audio回路.....	8
2.9 Tuner.....	8
2.10 Ethernet.....	9
2.11 Boot option (起動オプション) .....	9
2.12 RTCリアルタイムクロック .....	9
2.13 CANバスインタフェース .....	9
2.14 ブザー .....	10
2.15 LED.....	10
2.16 KEY.....	10
2.17 JTAG/SWDインタフェース.....	10
2.18 電源変換と電源指示LED .....	11
2.19 5V DC電源入力インタフェース.....	11
2.20 USBインタフェース.....	11
2.21 COM1.....	11
2.22 COM2.....	11
2.23 RS-485 インタフェース .....	12
2.24 RS-232 とRS-485 の選択ジャンパ.....	12
2.25 殊機能インタフェース .....	12
2.26 拡張インタフェース.....	12
2.27 2.4G無線モジュールインタフェース.....	13
2.28 315MHz無線モジュールインタフェース .....	13
第三章 タッチパネル付き 3.2 インチ/2.8 インチTFT 液晶.....	14



第四章 サンプルプログラムについて.....	18
4.1 サンプルプログラムの構造.....	18
4.2 サンプルプログラム紹介 .....	20
4.2.1 LED.....	20
4.2.2 BEEP.....	21
4.2.3 KEY検出と 315MHz無線モジュール.....	22
4.2.4 COM1 のprintfテスト .....	24
4.2.5 COMの送受信テスト.....	26
4.2.6 RS-485 の送受信テスト .....	27
4.2.7 CAN通信テスト.....	29
4.2.8 ADCテスト.....	31
4.2.9 I2C EEPROMテスト.....	32
4.2.10 SPI Flashテスト.....	33
4.2.11 SysTickテスト.....	35
4.2.12 SRAMテスト.....	35
4.2.13 Nor Flashテスト.....	37
4.2.14 Nand Flashテスト.....	39
4.2.15 FM Tunerテスト.....	42
4.2.16 2.4G無線通信テスト.....	43
4.2.17 LAN通信テスト.....	46
4.2.18 SDカードテスト.....	49
4.2.19 Audio Playテスト.....	52
第五章 実行ファイルの書き込み.....	55
5.1 シリアルポートで書き込む.....	55
5.2 OpenLinkで書き込む .....	60
5.2.1 ドライバのインストール.....	60
5.2.2 J-FLASH ARMで実行ファイルを書き込む.....	63
5.3 H-JTAGで実行ファイルを書き込む.....	68
第六章 OpenLinkでデバッグ .....	74
6.1 J-Link commandでデバッグ.....	74
第七章 開発ツールKEILの応用 .....	75
7.1 KEILのインストール.....	75
7.2 既存のプロジェクトから .....	77
7.3 新しいプロジェクトの作成.....	81



## 第一章 通信機能搭載-STM32F103ZEキット概要

ARM コア新型プロセッサCortex-M3 を採用した ST マイクロエレクトロニクス社の STM32F103ZET6 (LQFP144)。

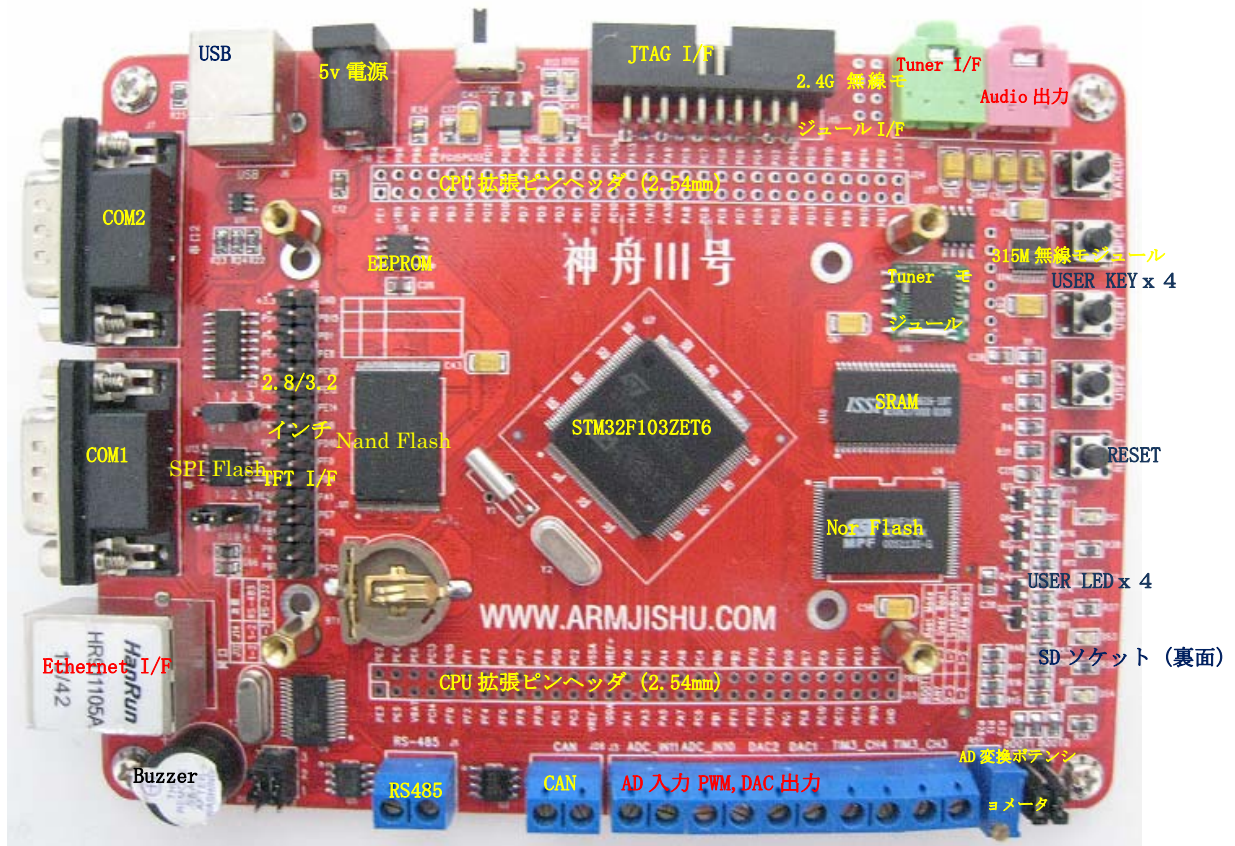
標準外付け：4M bit SRAM、16M bit Nor Flash、1G bit Nand Flash、SPI、IIC、USB、LAN、無線LAN、SDIO、FSMC、ADC、DAC、PWM、CANなど。

豊富なハードウェアの上、色々なサンプルソースを提供しているので、初心者最適です。

### 1.1 主な特徴

- STM32F103ZE、LQFP144、512KB FLASH/64KB RAM内蔵
- 外付け 4M bit SRAM、16M bit NOR FLASH(128M bit まで拡張可)、大容量のデータ採集、処理と分析ができる
- 1G bit Nand Flash、画像などのデータを出来る
- SPI インタフェース、W25X16(16M bit DATA FLASH)
- IIC インタフェース、24LC02(2M bit EEPROM)
- Tuner モジュール搭載
- I2S オーディオ出力 DA モジュール搭載
- USB slave インタフェース搭載
- 10M Ethernet 搭載
- SD インタフェース搭載
- GUI、3.2或いは2.8インチ、320\*240、26 万色TFT-LCD、8/16BitのBUSをサポートする、16Mbit SPI Flash(AT45DBxxx)未実装、SDソケット付き、タッチパネル (ADS7843) 付き
- ユーザーボタン x 4、RESET x 1
- ユーザーLED x 4
- 外部電源インタフェース、極性：センタープラス
- 電源SW x 1
- 標準JTAG/ICE デバッグ用インタフェース(20pin)、JLinkに給電
- RS485 x 1、RS232 x 2 (DB9)
- CAN BUS x 1、SN65VHD230
- ブザー x 1
- ポテンショメータ入力アナログ信号 x 1
- AD、DC、PWMインタフェース
- 2.4G無線通信モジュールインタフェース搭載
- 315M無線通信モジュール搭載
- RTCインタフェース搭載、CR1220をサポート
- CPU のすべての IO を 2.54mm 拡張ピンヘッダで引き出される
- 外形寸法：110×150(mm) ※突起物は除く

## 第二章 回路の説明

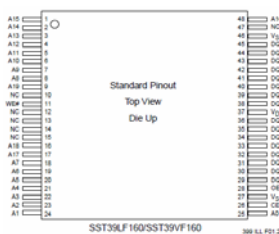


### 2.1 STM32F103ZET6

本ボードは ST 社の STM32F103ZET6 を採用しております。

主な仕様：LQFP144、ARM Cortex M3 コア、32 bit データ バス幅、512 KB Flash プログラム メモリ、64 KB データ RAM、72 MHz 最高クロック周波数、112 個プログラム可能 I/O 数、8 個タイマー数、3 (12 bit, 16 Channel) オンチップ ADC、2 (12 bit, 2 Channel) オンチップ DAC、CAN, I2C, SPI, USART、SDIO、USB インタフェース。

### 2.2 Nor Flash

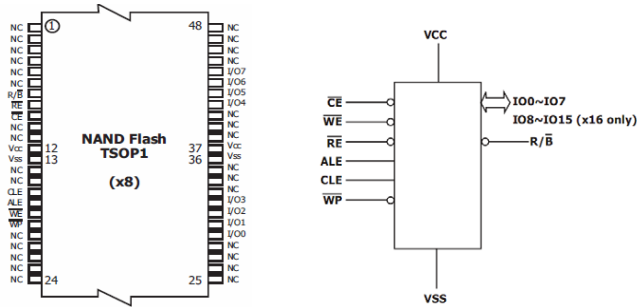


FSMC バスより 16M bit の Nor Flash を搭載しております。最大 128M bit まで拡張可。OS 或いは重要なデータを保存する場合利用します。

### 2.3 Nand Flash

FSMC バスより 1G bit の Nand Flash を搭載しております。画像或いは他のデータを保存す

る場合利用します。

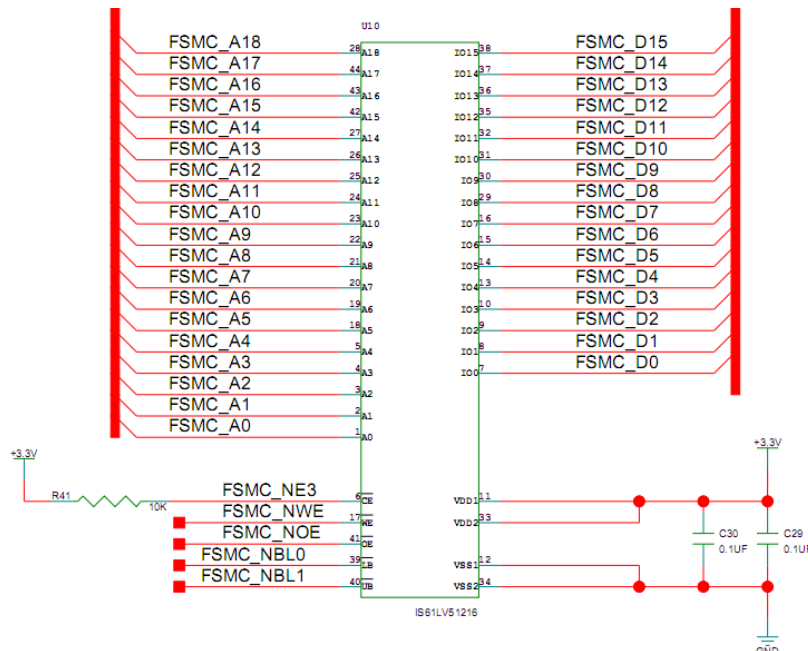
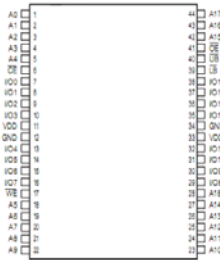


割り込みモードでNand Flash をアクセスする場合 J13 の 1-2 をショートします。

ポーリングモードでNand Flash をアクセスする場合 J13 の 2-3 をショートします。

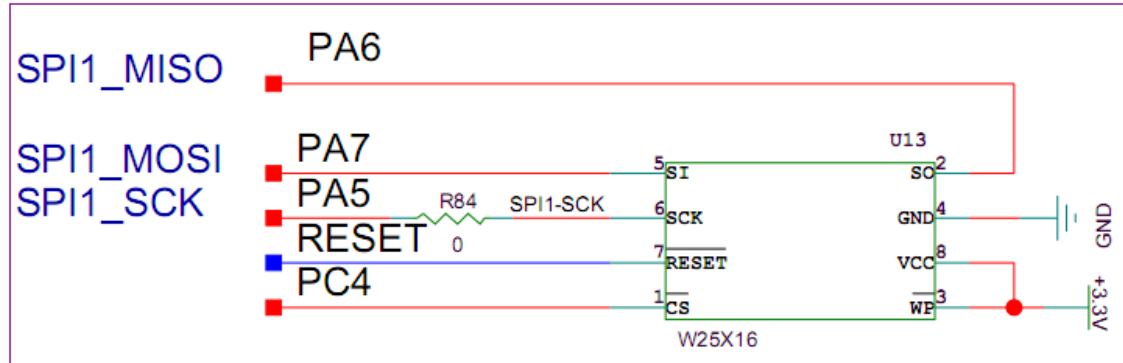
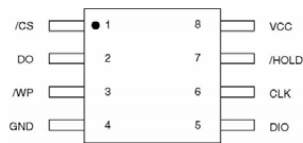
## 2.4 SRAM

FSMC バスより 8M bit の SRAM を搭載しております。システム或いはプログラムの実行中の臨時的なデータを書き込んだり、読み出したりする場合利用します。



## 2.5 SPI Flash

本ボードは 16M bit の SPI Flash チップ W25X16 を搭載しております。あんまり変更しないデータを保存する場合利用します。



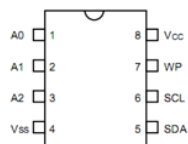
注意：本ボードでは W25X16 とイーサネット両方も SPI1 を利用していて、同時に両方の CS を有効に設定しないでください。

## 2.6 SDカードインタフェース

本ボードは SDIO モードの SD カードインタフェースを搭載しております。外部メモリとして利用します。

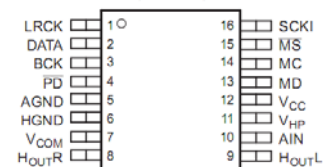
## 2.7 I2C EEPROM

本ボードは 24C02 を搭載しております。コンフィグデータ保存或いはあんまり変更しないデータを保存する場合利用します。



## 2.8 Audio回路

本ボードは Audio 回路 PCM1770 を搭載しております。Flash 上に保存している音声ファイルを再生して歌を聴いたり、プログラム上提示音を再生したりできます。

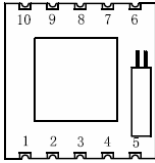


MCU から I2S3 を通じて音声信号を PCM1770 に転送します。PCM1770 でディコードして J2 に出力します。また PCM1770 のコンフィグインタフェースは MCU の SPI2 と繋いでいて、MCU は SPI で PCM1770 をアクセスします。

## 2.9 Tuner

本ボードは TEA5767 を搭載して、Tuner 機能を実現しております。





STM32F103ZET6 の MCU は I2C インタフェースで TEA5767 をアクセスと配置します。

### 2.10 Ethernet

本ボードは ENC28J60 で実現した 10M のイーサネットインタフェースを搭載しております。インターネット通信ができます。

MCU は SPI1 で ENC28J60 をアクセスします。

### 2.11 Boot option (起動オプション)

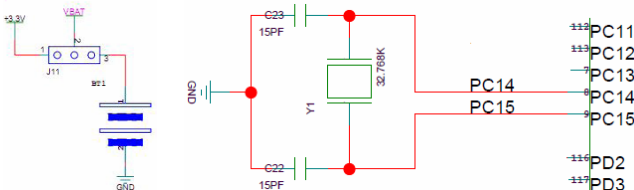
ジャンパで三つの Boot 方法を設定する。

BOOT1 (J9) と BOOT0 (J10) で制御する。

BOOT1 (J9)	BOOT0 (J10)	
ANY (1-2、2-3 or open)	2-3	Embedded user Flash (デフォルト) モード ※Flash は起動アドレス 0x0000 0000 にマップされる、でも本来のアドレス 0x0800 0000 からアクセスできる。
2-3	1-2	System memory モード ※システムメモリは起動アドレス 0x0000 0000 にマップされる、でも本来のアドレス 0x1FFF F000 からアクセスできる。
1-2	1-2	Embedded SRAM モード ※アドレス 0x2000 0000 から SRAM アクセスできる。

### 2.12 RTCリアルタイムクロック

外部バッテリー (CR1220 ボタン型バッテリーをサポートする) と 32768 水晶振動子で本当の RTC 機能を実現できます。外部バッテリー設置してない場合、システムに影響しない様に、ジャンパで RTC クロック機能をマスクできます。

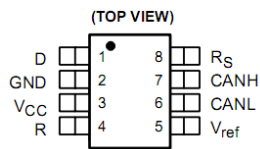


外部バッテリー設置した後ジャンパ J11 の 2-3 をショートすれば、VBAT ピンは外部バッテリーから給電されます。ジャンパ J11 の 1-2 をショートすれば、VBAT ピンは+3.3V のシステム電源から給電されます。

### 2.13 CANバスインタフェース

本ボードは VP230 の 3.3VCAN トランシーバを利用して STM32 の CAN バスインタフェースを

引き出しています。

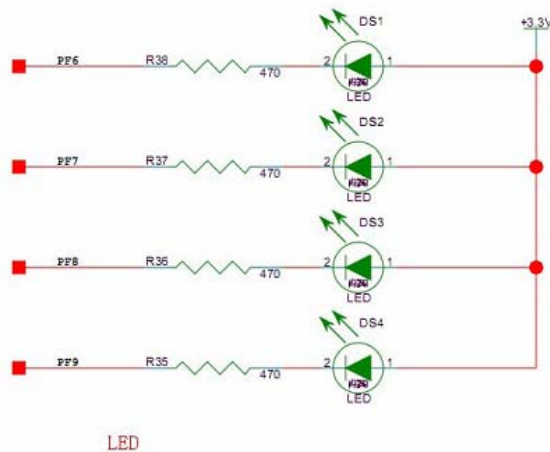


## 2.14 ブザー

本ボードはブザーを搭載しております。両側の電源を入れると、固定周波数の音声を出します。プログラム上提示音或いは警告音を出せます。

## 2.15 LED

本ボードは電源指示 LED 以外、4つの LED を搭載しております。それぞれ GPIO の PF6-9 の4つのピンと繋いで、Low レベルの時点灯します。



## 2.16 KEY

本ボードは reset key 以外、4つのユーザーKEY を搭載しております。

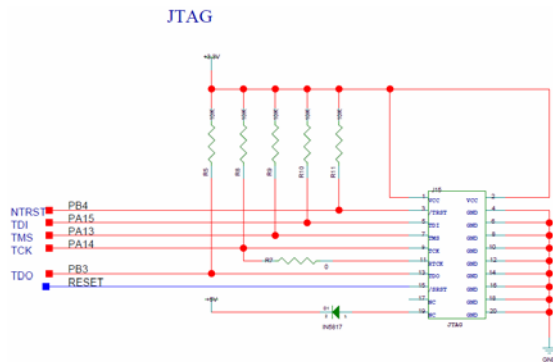
Reset Key 押下すると、全ボードハードウェア復帰で、MCU、液晶、イーサネット、Audio などの回路が復帰になります。

4つのユーザーKEY の中、User1 と User2 キー以外、Wakeup キー、Tamper キーもプログラムによって定義できます。

Wakeup キー、Tamper キー、User1 と User2 キーはそれぞれ GPIO の PA0、PC13、PA8、PD3 で制御しております。キー押下すると、その GPIO ピンは Low レベルで押下されてない場合は High レベルになります。

## 2.17 JTAG/SWDインタフェース

本ボードは標準の 20 ピンの JTAG インタフェースを搭載しております。ULINK、JLINK と直接繋がります。また SWD もサポートします。



JTAG/SWD インタフェースの信号定義：

SWJ-DP pin name	JTAG debug port		SW debug port		Pin assignment
	Type	Description	Type	Debug assignment	
TMS/SWDIO	I	JTAG Test Mode Selection	I/O	Serial Wire Data Input/Output	PA13
TCK/SWCLK	I	JTAG Test Clock	I	Serial Wire Clock	PA14
TDI	I	JTAG Test Data Input	-	-	PA15
TDO/TRACESWO	O	JTAG Test Data Output	-	TRACESWO if async trace is enabled	PB3
NTRST	I	JTAG Test nReset	-	-	PB4

## 2.18 電源変換と電源指示LED

STM32F103ZET6 は 3.3V のワーク電圧なので、外部入力電源あるいは USB から提供した 5V の電源を 3.3V に変換する必要があります。本ボードは ASM1117-3.3V で電源変換機能を実現しております。またユーザーから電源が正しく提供しているか確認できる様に、一つの緑色の電源指示 LED を搭載しております。ボードの 3.3V 電源が正常の状態であれば点灯します。

## 2.19 5V DC電源入力インタフェース

5V/1A DC センタープラグ、内径 2.1mm、外径 5.5mm の電源インタフェースを搭載しております。

## 2.20 USBインタフェース

標準的な USB SLAVE インタフェースを搭載しております。また PC 或いは他の USB マスター設備からボードへの給電も出来ます。最大 500mA。ESD (IEC61000-4-2 (ESD 15kV air, 8kV Contact) 保護回路も搭載しております。

## 2.21 COM1

STM32F103ZET6 の USART12 を MAX3232 通じて 232 レベルのシリアルポートに変換しております。インタフェースは DB9 オス型、3 線シリアルを実現しております。ピン定義：

ピン No.	信号
2	RXD
3	TXD
5	GND

## 2.22 COM2

COM2 と COM1 は同じ方法で実現しております。違う所は、ジャンパで USART2 を RS232 にす

るか RS485 にするか選択できる点です。

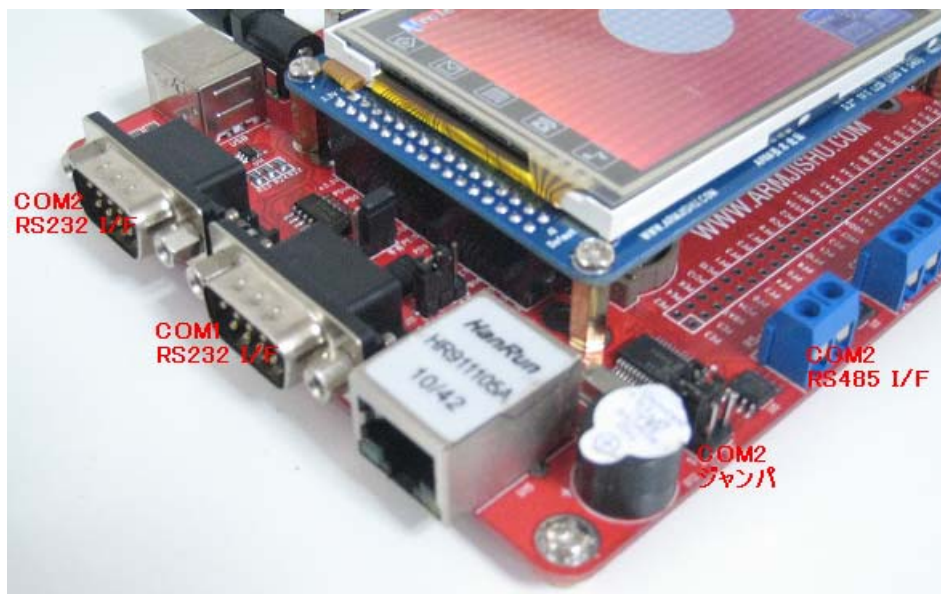
J14	J12	COM2 機能
1-2	1-2	RS-485
2-3	2-3	RS-232

### 2.23 RS-485 インタフェース

SP3485VP230 で実現しております。

### 2.24 RS-232 と RS-485 の選択ジャンパ

本ボードに搭載している COM2 は RS-232 或いは RS-485 のインタフェースを実現しております。



J12	J14	機能
1-2	1-2	RS-485 インタフェース
2-3	2-3	RS-232 インタフェース

### 2.25 特殊機能インタフェース

本ボードには STM32 の MCU の特殊機能のピンを予備インタフェースとして引出しております。ピン定義は下記です：

ピン No.	機能	ピン No.	機能
1	GND	6	DAC1
2	ADC_IN11	7	GND
3	ACD_IN10	8	TIM3_CH4
4	GND	9	TIM3_CH3
5	DAC2	10	GND

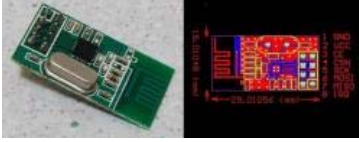
### 2.26 拡張インタフェース

本ボードには全ての GPIO を 2.54mm 拡張ヘッダで引き出しております。プログラムの開発、

デバッグがより便利にできます。

## 2.27 2.4G無線モジュールインタフェース

本ボードは2.4G 無線微弱モジュール nRF24L01 (オプション) のインタフェースを提供しております。

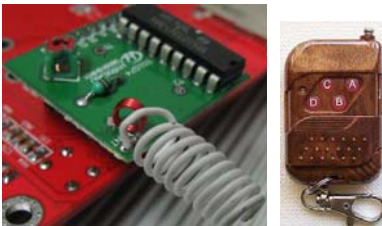


商品紹介URL : <http://www.csun.co.jp/SHOP/2009061924.html>

注意：nRF24L01 の IRQ ピンは 3.2 インチの LCD で搭載している SD カードの CS 信号と共用している為、3.2 インチ LCD で搭載している SD カードを利用する場合、nRF24L01 は利用できません。

## 2.28 315MHz無線モジュールインタフェース

本ボードは 315MHz 無線微弱モジュール (オプション) のインタフェースを提供しております。リモコンで制御できます。

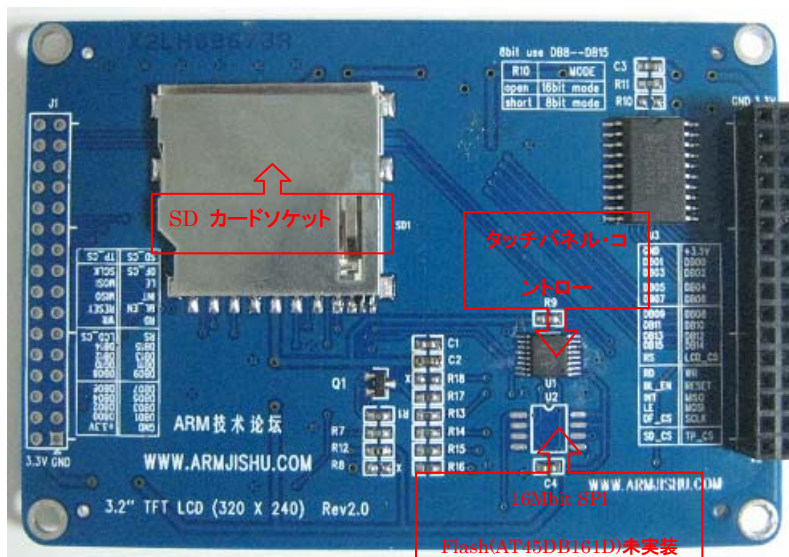


## 第三章 タッチパネル付き 3.2 インチ/2.8 インチTFT 液晶

表面：



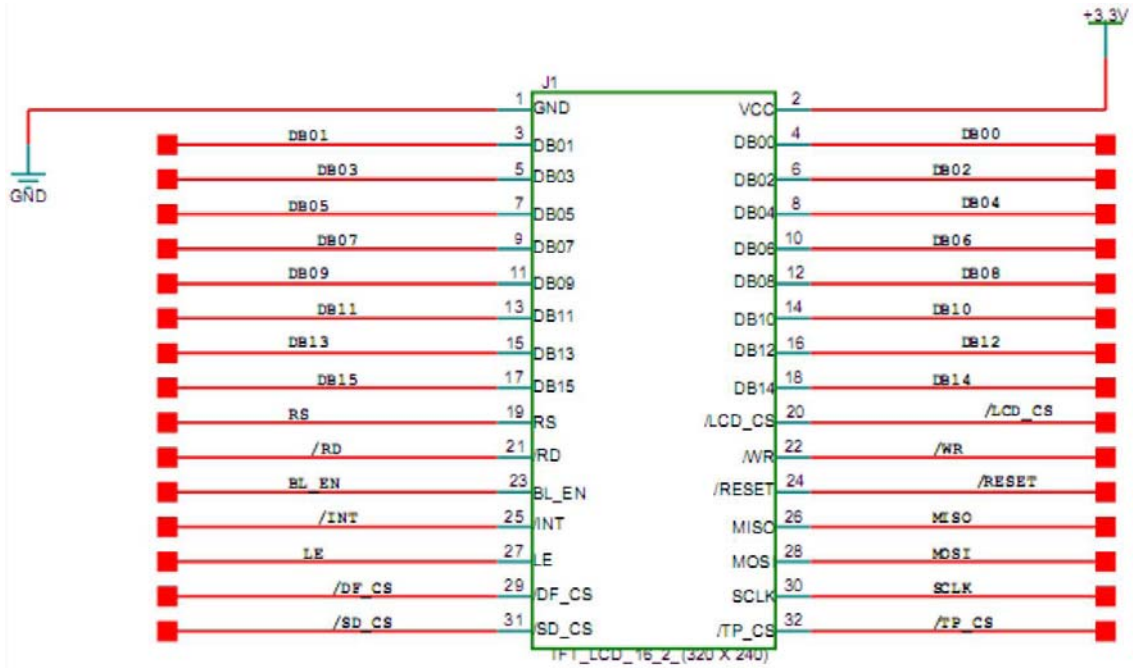
裏面：



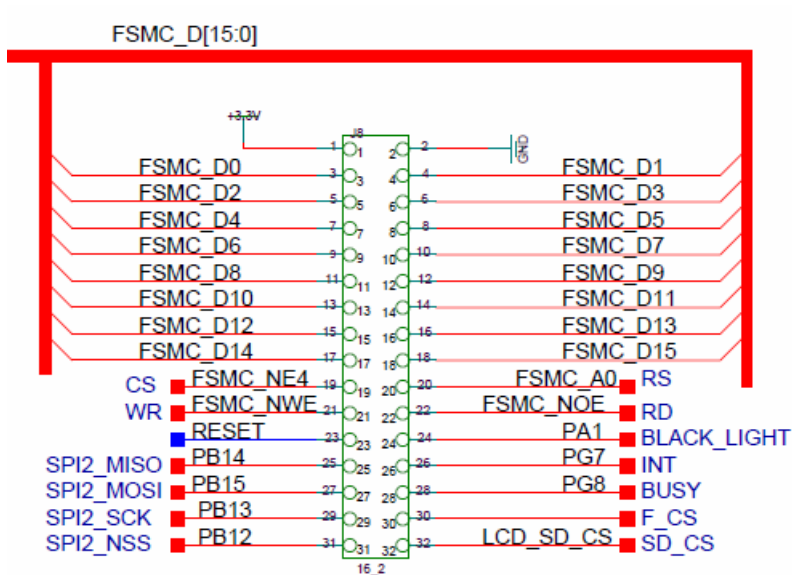
- ・ 3.2 インチTFT 液晶、解像度は240(W)\*320(H)
- ・ 8/16bit パラレルインタフェース
- ・ タッチパネル・コントローラADS7843 或いはTSC2046 (SPI インタフェース)
- ・ 16Mbit SPI Flash(AT45DB161D)未実装
- ・ SD カードソケット

- ・ 使いやすい2.54mm コネクタ。
- ・ 外形寸法： 3.2 インチ、95×62(mm) ※突起物は除く

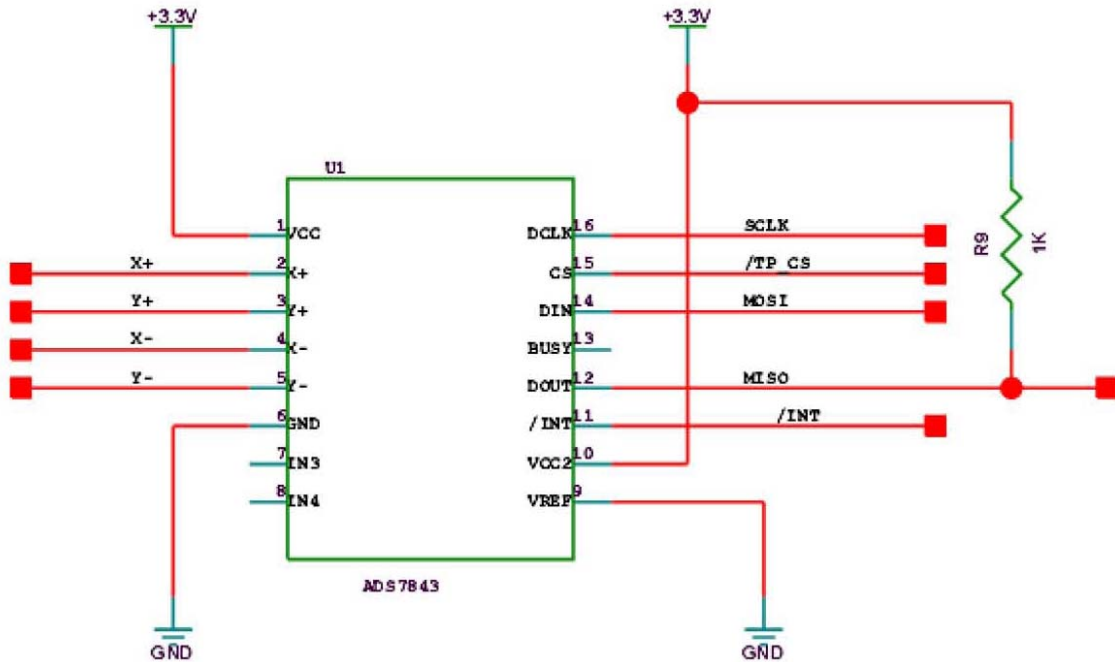
2.8 インチ/3.2 インチ TFT LCD インタフェース：



MCU とのインタフェース：



タッチパネル：



MCU と 2.8 インチ/3.2 インチタッチパネル TFT LCD のピン対応：

GPIO ピン	信号名	TFT LCD 信号	説明
PD14	FSMC_D0	D0	
PD15	FSMC_D1	D1	
PD0	FSMC_D2	D2	
PD1	FSMC_D3	D3	
PE7	FSMC_D4	D4	
PE8	FSMC_D5	D5	
PE9	FSMC_D6	D6	
PE10	FSMC_D7	D7	
PE11	FSMC_D8	D8	
PE12	FSMC_D9	D9	
PE13	FSMC_D10	D10	
PE14	FSMC_D11	D11	
PE15	FSMC_D12	D12	
PD8	FSMC_D13	D13	
PD9	FSMC_D14	D14	
PD10	FSMC_D15	D15	
PG12	FSMC_NE4	CS	LCD チップセレクト信号
PF0	FSMC_A0	RS	コマンド/データフラグ





			(1 : データ、0 : コマンド)
PD5	FSMC_NWE	WR	LCD に書き込む
PD4	FSMC_NOE	RD	LCD から読み出し
PA1	BLACK LIGHT	BLACK LIGHT	バックライトコントロール
PG7	INT		バックライトコントロール
PG8	BUSY		
-	-	F_CS	
PG15	SD_CS	SD_CS	TFT LCD 搭載している SD カードの SPI CS 信号、2.4G モジュールの IRQ と共用
PB14	MISO	MISO	タッチパネル SPI MISO 信号
PB15	MOSI	MOSI	タッチパネル SPI MOSI 信号
PB13	SCK	SCK	タッチパネル SPI SCK 信号
PB12	NSS	NSS	タッチパネル SPI CS 信号

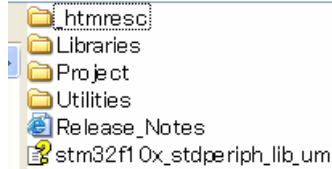
## 第四章 サンプルプログラムについて

### 4.1 サンプルプログラムの構造

提供しているサンプルプログラムは全て標準の STM32F10x\_StdPeriph\_Lib\_V3.3.0 を基づいて作成しております。LED サンプルを例として、サンプルソースの構造を説明します。

STM32F10x\_StdPeriph\_Lib\_V3.3.0 フォルダの構成は下記のとおりです：

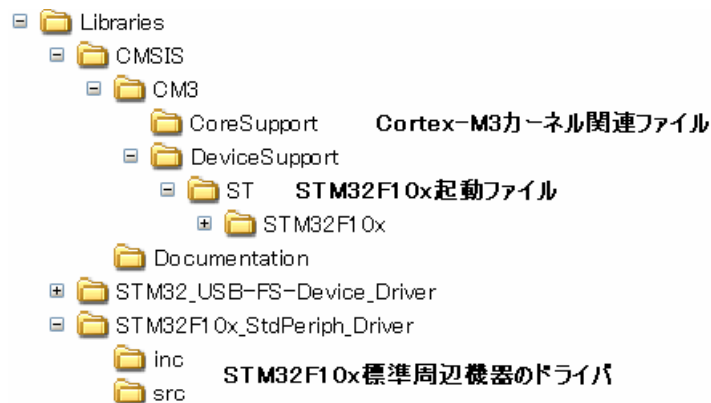
¥STM32F10x\_StdPeriph\_Lib\_V3.3.0



各フォルダの内容：

**\_htmresc** : Html ページで使用する画像ファイル

**Libraries** : Cortex-M3 カーネル関連ファイル、stm32f10x 起動ファイルなど。詳細内容



**Utilities フォルダ** : 共有フォルダ、ST 社評価版関連ヘッダファイルも含まれている

**Release\_Notes.html** : STM32F10x\_StdPeriph\_Lib\_V3.3.0 説明ファイル

**stm32f10x\_stdperiph\_lib\_um.chm** : STM32F10x\_StdPeriph\_Lib\_V3.3.0 ヘルプファイル

**Project フォルダ** :

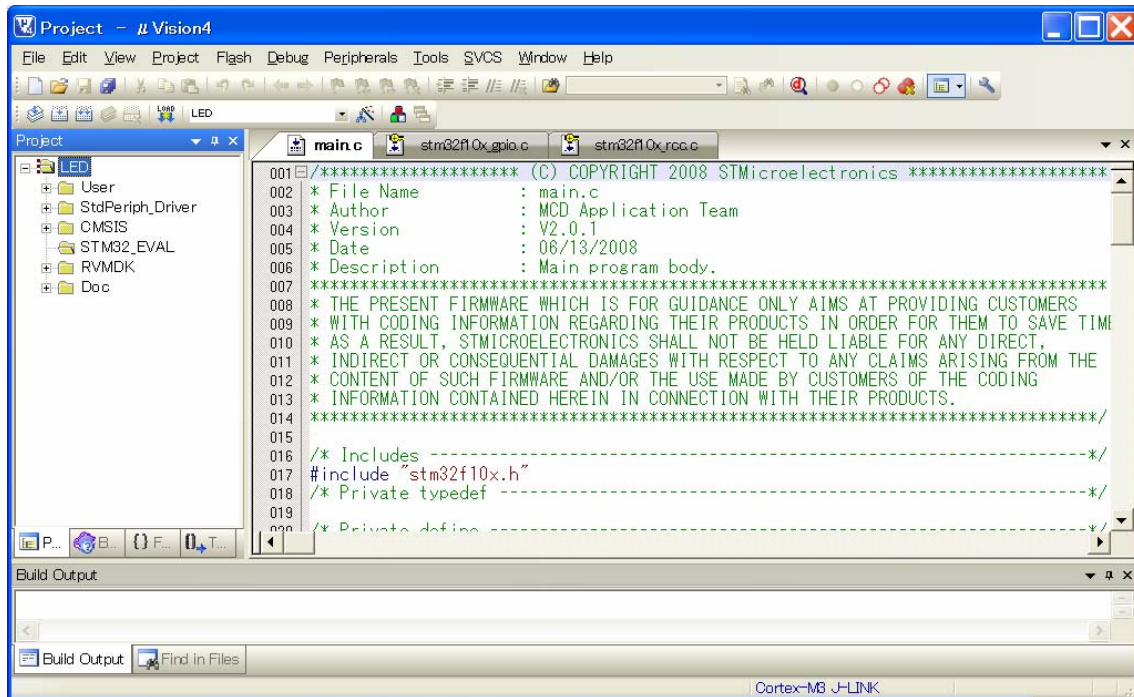


- Project
  - 01.LED
  - 02.BEEPER
  - 03.KEY\_LEDand315Mwireless
  - 04.USART-COM1
  - 05.COM
  - 06.RS485
  - 07.CAN(LoopBack)
  - 08.ADC
  - 09.EEPROM
  - 10.SPI FLASH
  - 11.SysTick
  - 12.SRAM
  - 13.NOR FLASH
  - 14.NAND FLASH
  - 15.FMTuner
  - 16.2.4GWireless
  - 17.Calendar
  - 18.ENC28J60
  - 19.TFTLCD\_2.8inch\_9320and8989
  - 19.TFTLCD\_3.2inch\_9320and8989
  - 20.SD
  - 21.MP3Player
  - 22.uCOS+uCGUI(FSMC2.8inch\_9320\_8989)
  - 22.uCOS+uCGUI(FSMC3.2inch\_9320\_8989)
  - STM32F10x\_StdPeriph\_Examples
  - STM32F10x\_StdPeriph\_Template

Project¥01.LED を開くイメージ :



各フォルダに各開発環境のプロジェクトファイルがあります。本ボードは MDK 環境を基  
づいたサンプルを提供しております。MDK-ARM フォルダにあるプロジェクトファイルを  
クリックしてプロジェクトを開けます。開いたイメージ :



## 4.2 サンプルプログラム紹介

### 4.2.1 LED

本試験によって STM32 の基本 IO ポートの制御方法を把握できます。

- ・ 原理

STM32 の IO は 8 種のモードに設定できます。

ライブラリファイルに定義があります。

```
typedef enum
{
    GPIO_Mode_AIN = 0x0,
    GPIO_Mode_IN_FLOATING = 0x04,
    GPIO_Mode_IPD = 0x28,
    GPIO_Mode_IPU = 0x48,
    GPIO_Mode_Out_OD = 0x14,
    GPIO_Mode_Out_PP = 0x10,
    GPIO_Mode_AF_OD = 0x1C,
    GPIO_Mode_AF_PP = 0x18
}GPIO_Mode_TypeDef;
```

GPIO を利用する際、GPIO のクロック、モード及びスピードを設定する必要があります。

アウトプットとして利用する時、サポートする最大クロックは 10MHz、2MHz、50MHz があります。

ライブラリファイル上の定義：

```
typedef enum
{
    GPIO_Speed_10MHz = 1,
    GPIO_Speed_2MHz,
    GPIO_Speed_50MHz
}GPIO_Speed_TypeDef;
```

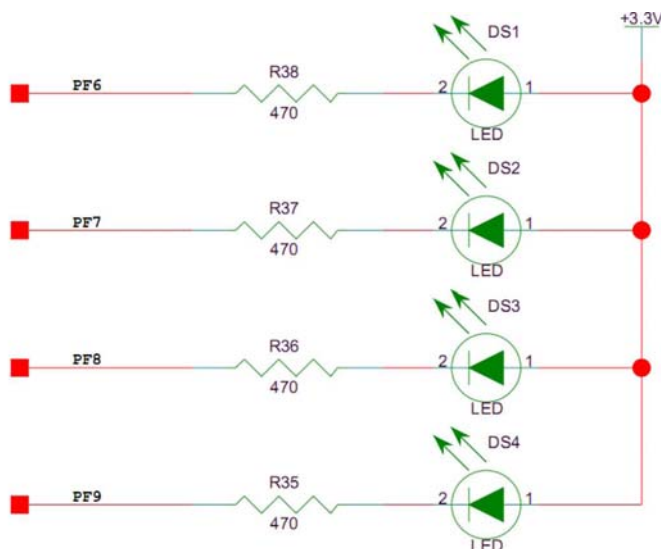
STM32 プロセッサの消費電力を下げる為に、実際の必要によって適当なクロックを設定します。クロックが低くなると消費電力も低くなります。

下記コードは本試験で4つのLEDを制御するGPIOの初期化の例です：

```
RCC_APB2PeriphClockCmd(RCC_GPIO_LED, ENABLE); /*ENABLE LED GPIO CLOCK*/  
GPIO_InitStructure.GPIO_Pin = DS1_PIN|DS2_PIN|DS3_PIN|DS4_PIN; /*ENABLE LED GPIO PIN*/  
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP; /*SET LED GPIO MODE*/  
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz; /*SET LED GPIO CLOCK*/  
GPIO_Init(GPIO_LED, &GPIO_InitStructure); /*INITIALIZE LED GPIO*/
```

### ・ハードウェア設計

本ボードは4つのLEDを搭載しております。それぞれGPIOのPF6-9の4つのピンと繋いで、Lowレベルの時点灯します。以下は回路です。GPIOのPINと繋いでいる抵抗は電流過大を防ぐ為です。



GPIOのPINとLEDの対応関係：

DS1	PF6
DS2	PF7
DS3	PF8
DS4	PF9

### ・ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0¥Project¥01.LED¥main.c をご参照ください。

### ・結果

.hex ファイルをボードの書き込んで実行すると、4つのLED(DS1~4)が順次に点滅します。

## 4.2.2 BEEP

本試験もLEDと同じ様にSTM32のIOの出力の制御で実現しております。

### ・ 原理

本ボードのブザーは両側の電圧が4Vより大きく、標準では5Vになると入れると、固定周波数の音を出します。

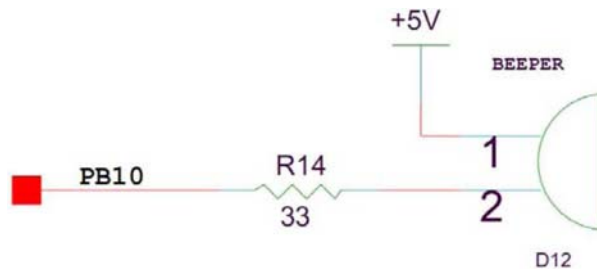
本試験ではGPIOのPINがHighレベルになるとブザーを閉じて、Lowレベルになると発声します。

### ・ ハードウェア設計

本ボードは一つのブザーを搭載しております。GPIOのPB10のピンと繋いで、Lowレベルの時発声します。以下は回路です。GPIOのPINと繋いでいる抵抗R14は電流過大を防ぐ為です。STM32のIOの最大電流は25mAとなっております。

$I_{IO}$	Output current sunk by any I/O and control pin	25
	Output current source by any I/Os and control pin	- 25

なお、ブザーのワーク電流は30mAなので、R14を利用してあります。



### ・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0¥Project¥02.BEEP¥main.c をご参照ください。

### ・ 結果

.hex ファイルをボードの書き込んで実行すると、一定間隔ある音が鳴ります。

## 4.2.3 KEY検出と315MHz無線モジュール

本試験ではSTM32のIOの入力制御で実現しております。

### ・ 原理

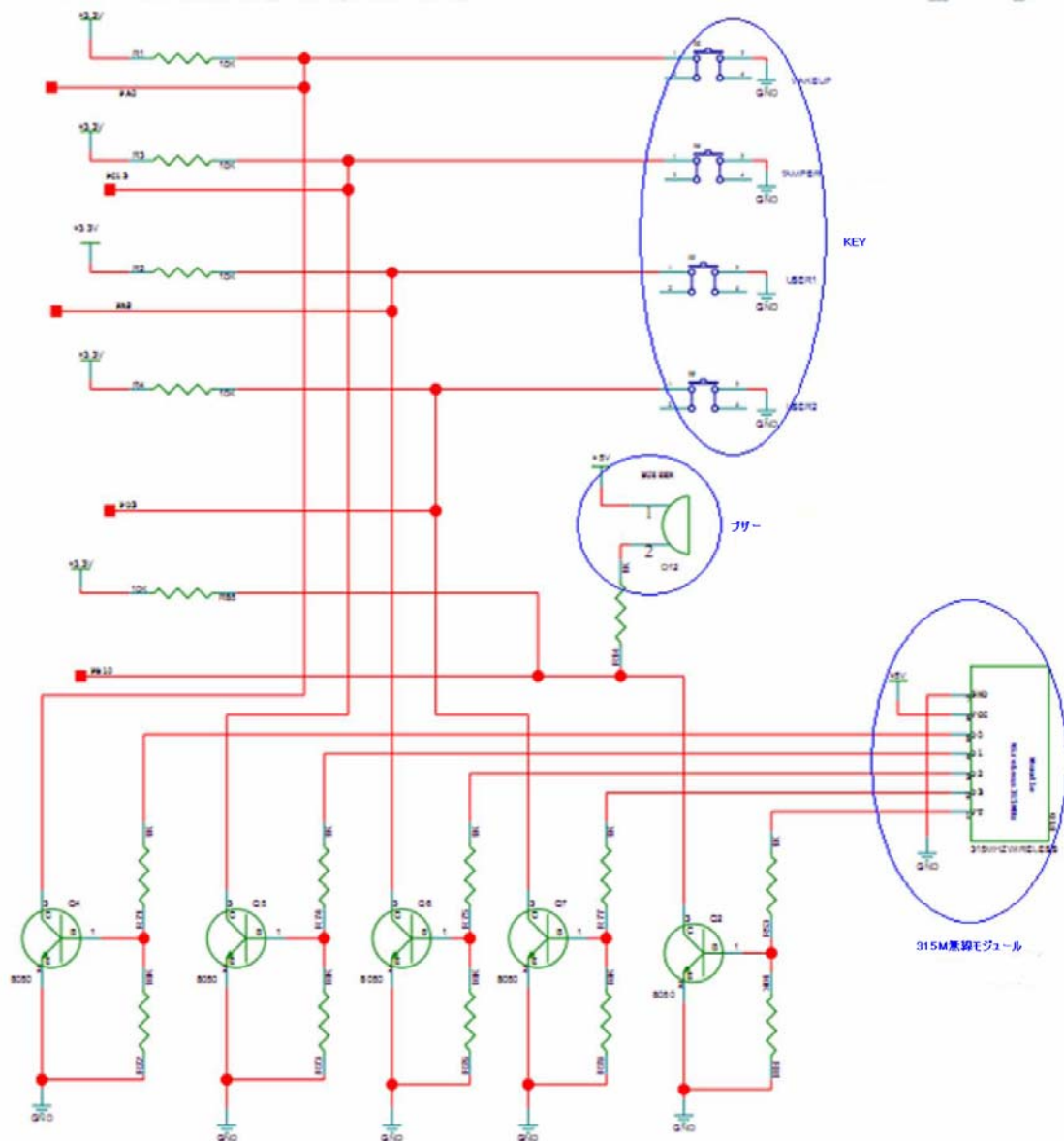
ボードの四つのKEY (WAKEUP、TAMPER、USER1 と USER2) で4つのLEDを点滅させる。また315M無線モジュールと対応しているリモコンのKEYを押すと、315MHzの無線信号によってボード搭載しているモジュールのアンテナから受信し、PT2272で解析しVT端をHighレベルに設定します。そして解析結果をD0~D3のGPIOピンからSTM32に入力し、対応のLEDを点滅させます。

対応関係：

DS1点灯 USER2キー押下或いはリモコンのDキー押下

DS2 点灯 USER1 キー押下或いはリモコンの C キー押下  
 DS3 点灯 TAMPER キー押下或いはリモコンの B キー押下  
 DS4 点灯 WAKEUP キー押下或いはリモコンの A キー押下  
 リモコン上のキーを押下した場合、ブザーが鳴ります。  
 ボード上のキーを押下した場合、ブザーは鳴りません。

・ハードウェア設計



無線とボード搭載のキーは MCU の同じ GPIO を共用しております。どちらからの操作で GPIO の信号が変化するかは二つの判断方法があります：

- 一、無線モジュールの VT 信号が有効かを見る。リモコンのキー操作する時無線モジュールの VT は High レベルになります。またキーと対応している D0<sup>4</sup> ピンも High レベル

になります (315M 無線の VT と D0~4 は High レベル有効です)。

二、無線モジュールでリモコン信号を受信すると、ブザーが鳴ります。

GPIO ピンと対応 LED :

LED	GPIO ピン
DS2	PF6
DS3	PF7
DS4	PF8
DS1	PF9

GPIO ピンとキー、無線信号の対応関係 :

キーとブザー	315M 無線信号	GPIO ピン
WAKEUP	D0	PA0
TAMPER	D1	PC13
USER1	D2	PA8
USER2	D3	PD3
ブザー	VT	PB10

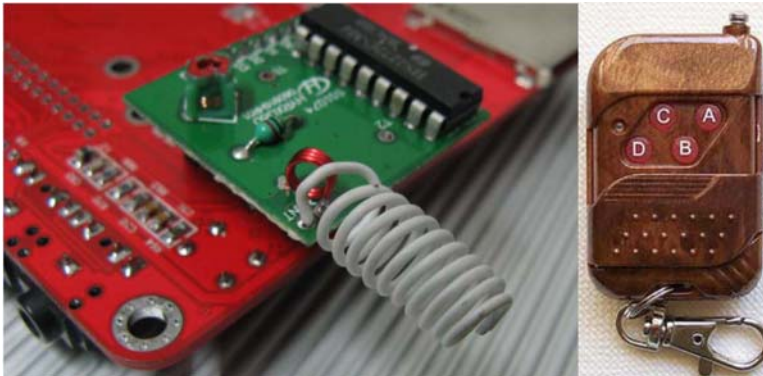
#### ・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0¥Project¥ 03.KEY\_LEDand315Mwireless¥main.c をご参照ください。

#### ・ 結果

無線モジュールの試験にはオプションのモジュールとリモコンが必要です。

ボードに挿入したイメージ :



.hex ファイルをボードの書き込んで実行する。キー操作ない場合、全ての LED が点灯します。ボード上のキー押下すると対応 LED は消灯します。リモコンのキーを押下すると対応 LED は消灯し、同時にブザーが鳴ります。

#### 4.2.4 COM1 のprintfテスト

本試験では COM の出力機能を利用して実現しております。



## ・ 原理

STM32 マイコンでは最大5つのCOMを提供していますが、ボードではCOM1とCOM2を引き出しております。COMを利用するには次の設定が必要です：COMのクロックの起動、相応IOモードの設定、ボーレート/データレングス/パリティビットなどの情報の設定。

主な関連レジスター：

クロックのイネーブル：APB2ENRの14ビット（COM1、その他のCOMはAPB1ENR）

COMのリセット：APB2RSTRの14ビット（COM1、その他のCOMはAPB1RSTR）

ボーレート：各COMは専用のボーレートレジスターがあります。USART\_BRR

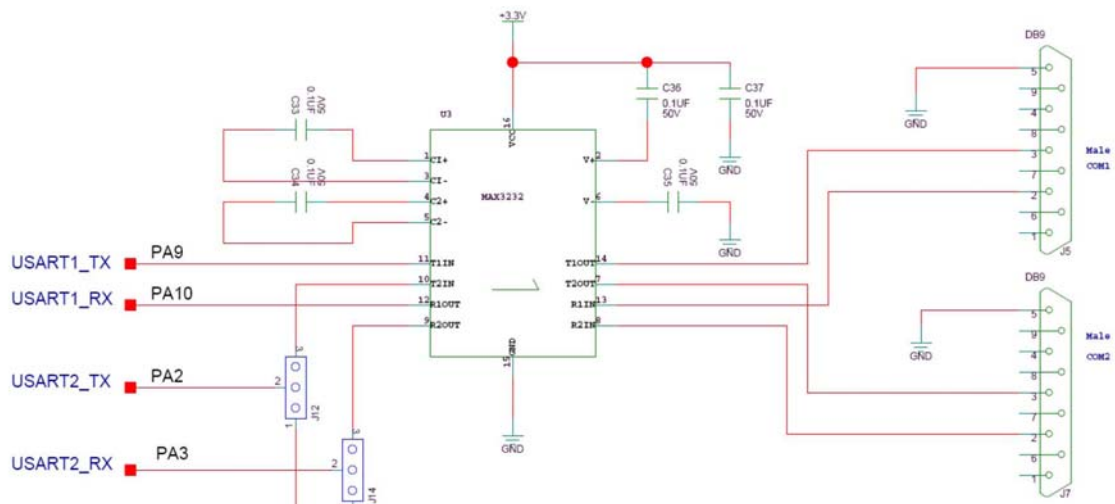
COMのコントロール：各COMは5つのコントロールレジスターUSART\_CR1~5があります。本試験ではUSART\_CR1だけ利用しています。

データ送信と受信：USART\_DR、Read/Write両方の機能を持っています。

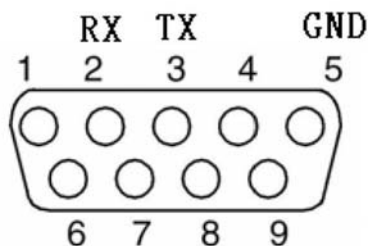
ステータスレジスター：USART\_SR

## ・ ハードウェア設計

STM32から出力するのはTTL/CMOS信号で、PCのCOMはRS-232信号なので、変換用にMAX3232を利用してあります。



ボードに搭載している二つのDB9のインターフェースはPCと同じです。なので、メス-メスクロスシリアルケーブルで二つのCOMを繋いだり、PCと繋いだりすることができます。



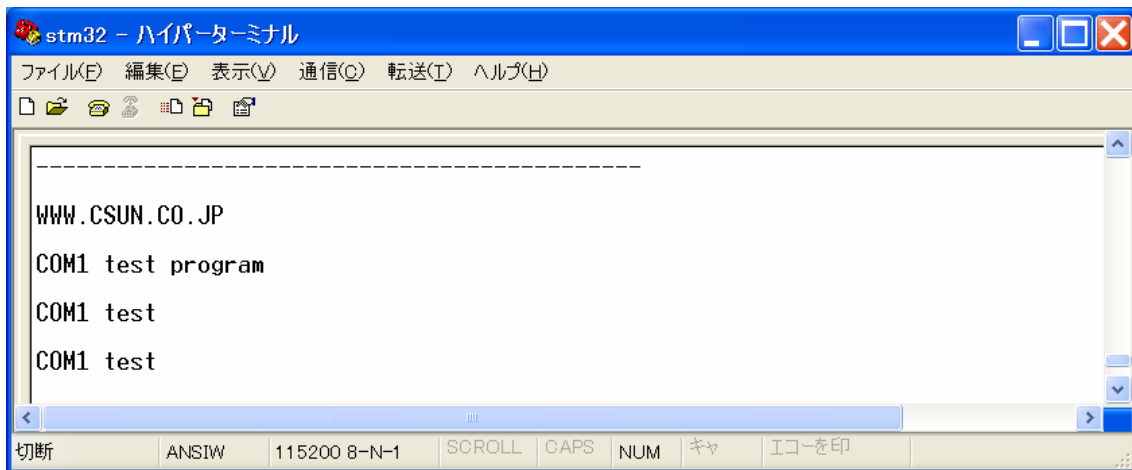
- ・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Libraries\STM32F10x\_StdPeriph\_Driver に USART のライブラリ関数 `stm32f10x_usart.c` と `stm32f10x_usart.h` があります。

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\04\_USART-COM1\main.c をご参照ください。

- ・ 結果

ボードの COM1 と PC の COM をクロスシリアルケーブルで繋ぎいで、ハイパーターミナルを起動して、下記のパラメータを設定します：115200、8、なし、1、なし。 .hex ファイルをボードの書き込んで実行する。ハイパーターミナル画面上下記情報が出力されます：



#### 4.2.5 COMの送受信テスト

本試験では COM の割込みモードで入力と出力機能を利用して実現しております。

- ・ 原理

本試験では COM1 と COM2 を同じパラメータを設定し、割込みモードで動作する様設定します。またクロスシリアルケーブルで繋がります。COM2 で COM1 から発送したデータを受信し、COM1 で COM2 から発送したデータを受信します。

- ・ ハードウェア設計

4.2.4 節をご参照ください。

- ・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Libraries\STM32F10x\_StdPeriph\_Driver に USART のライブラリ関数 `stm32f10x_usart.c` と `stm32f10x_usart.h` があります。

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\05\_COM\main.c をご参照ください。

##### クロックイネーブル

本試験では割込みモードで、COM2 で COM1 から発送したデータを受信し、COM1 で COM2 から発送したデータを受信します。また DS1~4 の LED でその結果を表します。なので、本試験で利用しているハードのリソースは LED、COM1 と COM2 です。利用前にこれらのリソースのクロックを設定する必要があります。これは `RCC_Configuration` 関数で実現しており

ます。

#### COM の割込みモード設定

割込みモード及び優先度を設定する必要があります。COM1 と COM2 二つの割込みソースがあって、割込みグループは一つで、COM1 の優先度を 0、COM2 の優先度を 1 に設定します (COM1 の優先度は COM2 の優先度より高い)。これは NVIC\_Configuration 関数で実現しております。

#### GPIO ピン設定

COM と LED 関連ピン設定。これは GPIO\_Configuration 関数で実現しております。

#### COM パラメータ設定

COM1 と COM2 を同じパラメータを設定します。

```
/* USART1 and USART2 configured as follow:
   - BaudRate = 9600 baud
   - Word Length = 8 Bits
   - One Stop Bit
   - No parity
   - Hardware flow control disabled (RTS and CTS signals)
   - Receive and transmit enabled
*/
```

#### COM の送受信割込み関数

上記の設定した後、COM から送信或いは受信割込みが発生する場合、割込み関数が呼ばれます。Stm32f19x\_it.c ファイルにある USART1\_IRQHandle() と USART2\_IRQHandle() です。

##### ・ 結果

ボードの COM2 のジャンパを確認します。J12 と J14 は 2-3 ショートし、RS232 インタフェースに設定します。クロスシリアルケーブルで COM1 と COM2 を繋ぎます。

.hex ファイルをボードの書き込んで実行します。正常の場合、DS1 と DS3 が常に点灯します。

各 LED の意味：

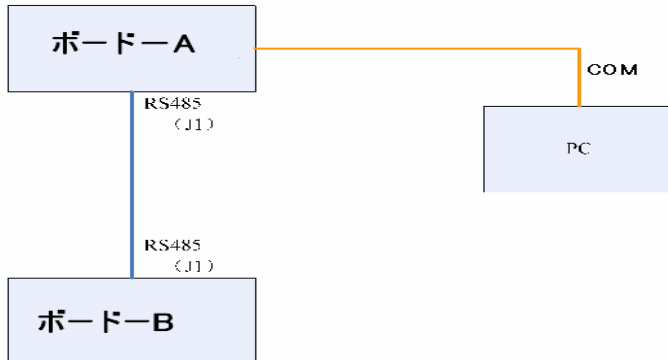
- DS1 点灯 COM1 受信データと COM2 送信データが一致
- DS2 点灯 COM1 受信データと COM2 送信データが不一致
- DS3 点灯 COM2 受信データと COM1 送信データが一致
- DS4 点灯 COM2 受信データと COM1 送信データが不一致

#### 4.2.6 RS-485 の送受信テスト

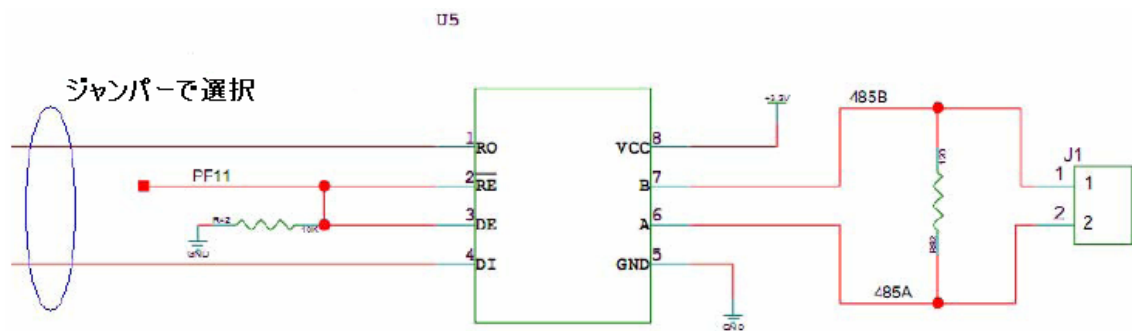
本試験では RS485 の入力と出力機能を利用して実現しております。二つのボードは必要です。一つは送信側として、もう一つは受信側となります。

##### ・ 原理

RS485 は RS232 より通信速度が高い、信号安定、通信距離長いなどのメリットがあります。下記接続図でボード-B は RS485 送信側で、ボード-A は RS485 受信側となって、受信後 COM1 からデータを PC 側に送信します。



## ・ハードウェア設計



U5 は Sipex 社の SP3485 で RS-485 の送受信機です。

主な特徴：

電源：3.3V

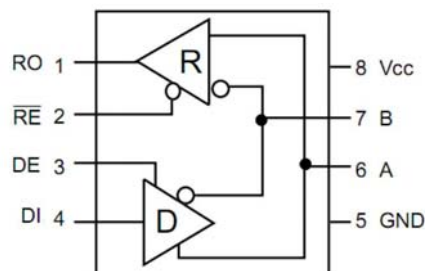
5V レベルと交換性あり

送信/受信イネーブルコントロール

最大 32 個のノードをサポートする

出力ショート保護回路あり

ロジック図：



RE と DE ピンで RS-485 のイネーブル制御を行います。本ボードではこの二つのピンは MCU の PF11 ピンと繋いで、PF11 ピンでコントロールしております。PF11 が High レベルの時、SP3485 は發送モードになり、Low レベルの時、受信モードになります。

## ・ソフトウェア設計



STM32F10x\_StdPeriph\_Lib\_V3.3.0\Libraries\STM32F10x\_StdPeriph\_Driver に USART のライブラリ関数 `stm32f10x_usart.c` と `stm32f10x_usart.h` があります。

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\06\_RS485\main.c をご参照ください。

#### ・ 結果

二つのボードの COM2 のジャンパを確認します。J12 と J14 は 1-2 ショートし、RS485 インタフェースに設定します。前述の接続図の様に二つのボードの RS485 インタフェース (J1) をウェアラインで接続します。ピン 1 とピン 1 を接続し、ピン 2 とピン 2 を接続します。またボード A をクロスシリアルケーブルで PC と繋いで、ハイパーターミナルを開いて、下記パラメータを設定します：115200、8、なし、1、なし。

二つのボードに .hex ファイルを書き込んで実行します。ハイパーターミナルの画面上に提示情報が表示されます。設置方法は：

USER1 キー押下 RS485 受信側に設定

USER2 キー押下 RS485 送信側に設定

LED の状態からボードの運行が分かります：

DS1 点灯 正常 (電源入って、DS1 は一度点滅してずっと点灯します)

DS2 点灯 ボードは RS485 送信側として、周期的にデータを送信します

DS3 点灯 ボードは RS485 受信側として、データを受信します。

ハイパーターミナル上も受信或いは送信データが表示されます。

```
RS485 test
--Press USER2, set RS485 sender
--Press USER1, set RS485 receiver
-----
RS485 receiving mode OK
Waiting for receiving data
Received data: [www.csun.co.jp RS485 test]
Received data: [www.csun.co.jp RS485 test]
Received data: [www.csun.co.jp RS485 test]
-----
RS485 sending mode OK
Sending data:www.csun.co.jp RS485 test
Sending data:www.csun.co.jp RS485 test
Sending data:www.csun.co.jp RS485 test
```

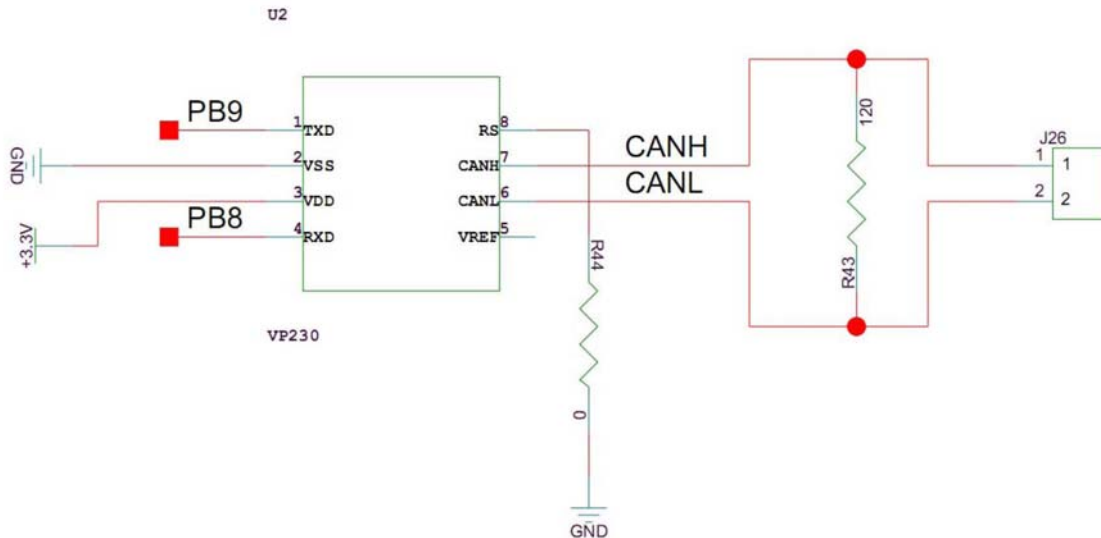
#### 4.2.7 CAN通信テスト

本試験では CAN バスの基本的な使用方法を利用して実現しております。CAN バスを初期化して、ポーリングモードと割込みモードで通信を行います。また LED で結果を表します。

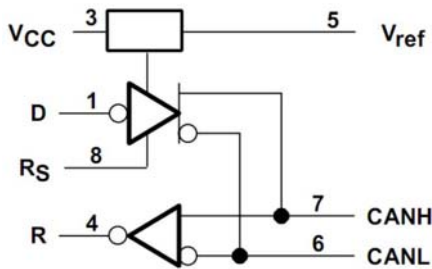
STM32 の CAN バスは 2.0A と 2.0B をサポートします。

・ハードウェア設計

本ボードではTI社のVP230を搭載してSTM32のCANと一緒にCANバスを実現しております。



VP230 のロジック図：



各ピンの機能：

TERMINAL NAME	NO.	DESCRIPTION
CANL	6	Low bus output
CANH	7	High bus output
D	1	Driver input
GND	2	Ground
R	4	Receiver output
RS	8	Standby/slope control
VCC	3	Supply voltage
Vref	5	Reference output

・ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0¥Project¥07.CAN (LoopBack) ¥Source¥main.c をご参照ください。

・結果

.hex ファイルを書き込んで実行します。

LED の状態から実行結果が分かります：

DS1 点灯 ポーリングモードで CAN バス受信成功

DS2 点灯 割込みモードで CAN バス受信成功

DS3 点灯 ポーリングモードで CAN バス受信失敗

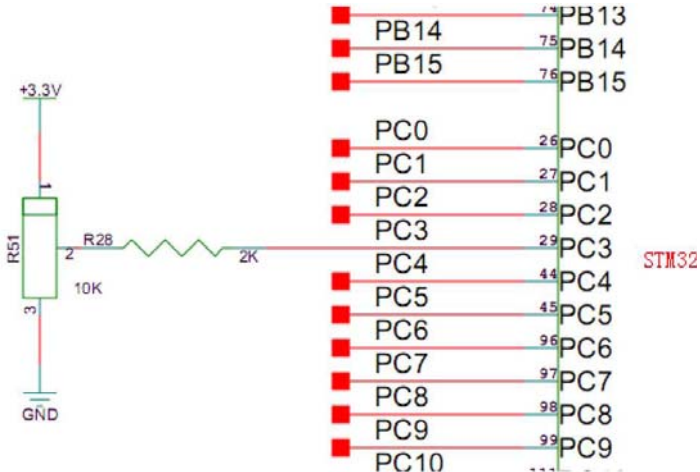
DS4 点灯 割込みモードで CAN バス受信失敗

#### 4.2.8 ADCテスト

本試験ではSTM32のADCを利用して可変抵抗器からサンプリングしたデータをADC変換後、シリアルポートから出力します。

##### ・ハードウェア設計

本ボードの可変抵抗器はPC3ピンに繋いでいます。



##### ・ソフトウェア設計

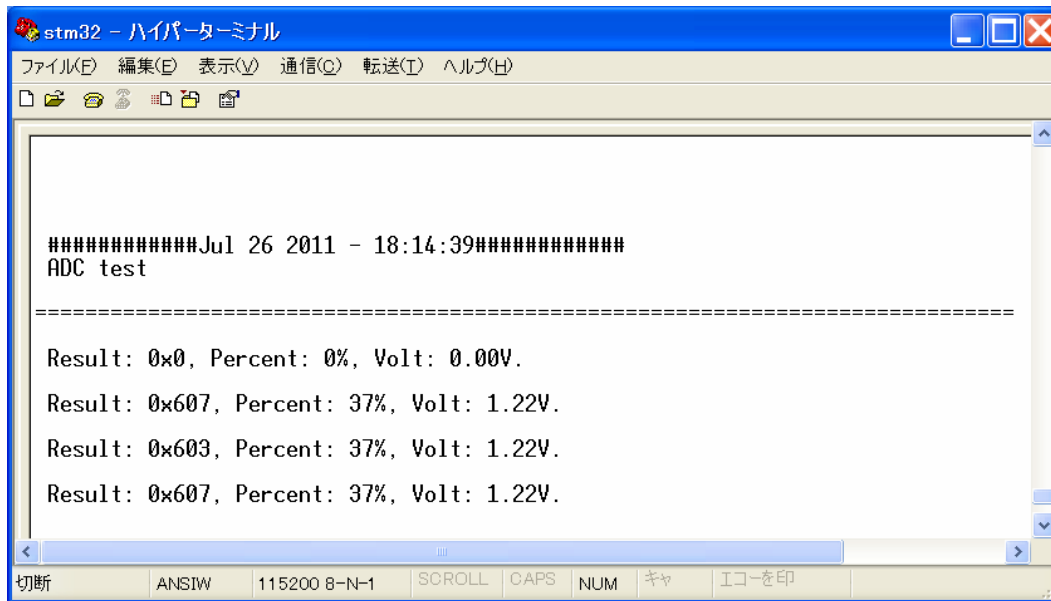
STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\08\_ADC\main.c をご参照ください。

##### ・結果

クロスシリアルケーブルでボードのCOM1とPCを繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル画面にADC変換結果が表示されます。可変抵抗器を調整すると、結果が変更します。



```
stm32 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
#####Jul 26 2011 - 18:14:39#####
ADC test
=====
Result: 0x0, Percent: 0%, Volt: 0.00V.
Result: 0x607, Percent: 37%, Volt: 1.22V.
Result: 0x603, Percent: 37%, Volt: 1.22V.
Result: 0x607, Percent: 37%, Volt: 1.22V.
切断 ANSIW 115200 8-N-1 SCROLL CAPS NUM キャ Iコーを印
```

#### 4.2.9 I2C EEPROMテスト

本ボードで搭載している EEPROM チップは 24C02、容量は 256Byte。STM32 の I2C インタフェースで 24C02 と接続しております。

本試験では先ず EEPROM 中にデータを書き込んで、また読み出して、シリアルポートから出力します。書き込んだデータと一致するか比較して EEPROM を正常にアクセスできるか判断します。

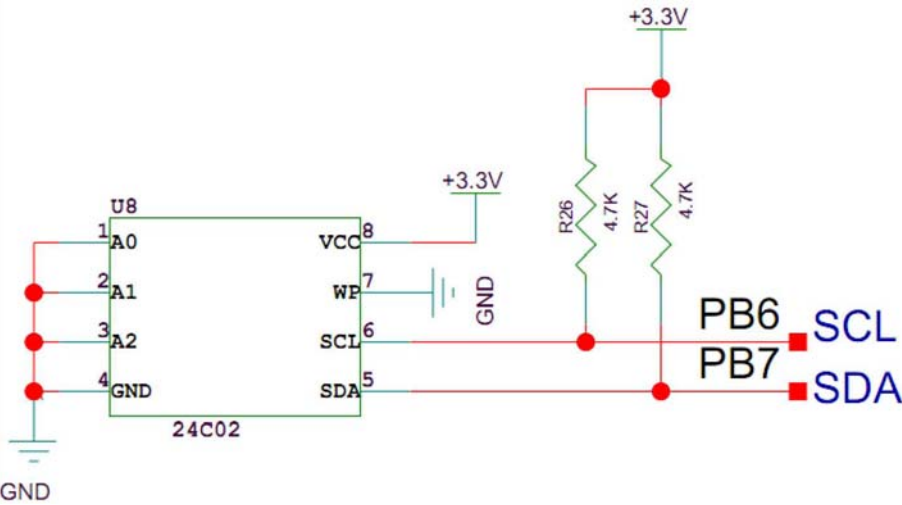
##### ・ハードウェア設計

STM32F103ZET6 は二つの I2C インタフェースがあります。

I2C インタフェース	ピン	GPIO	機能
I2C1	I2C1_SCL	PB6	I2C1 のクロック
	I2C1_SDA	PB7	I2C1 のデータ
I2C2	I2C2_SCL	PB10	I2C2 のクロック
	I2C2_SDA	PB11	I2C2 のデータ

本ボードでは I2C1 のインタフェースと EEPROM 24C02 を繋いでおります。回路図：





・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\09\_EEPROM\main.c をご参照ください。

・ 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル画面にテスト結果が表示されます。

Write data

0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1a	0x1b	0x1c	0x1d	0x1e	0x1f
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27	0x28	0x29	0x2a	0x2b	0x2c	0x2d	0x2e	0x2f
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x3a	0x3b	0x3c	0x3d	0x3e	0x3f
0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47	0x48	0x49	0x4a	0x4b	0x4c	0x4d	0x4e	0x4f
0x50	0x51	0x52	0x53	0x54	0x55	0x56	0x57	0x58	0x59	0x5a	0x5b	0x5c	0x5d	0x5e	0x5f
0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x67	0x68	0x69	0x6a	0x6b	0x6c	0x6d	0x6e	0x6f
0x70	0x71	0x72	0x73	0x74	0x75	0x76	0x77	0x78	0x79	0x7a	0x7b	0x7c	0x7d	0x7e	0x7f
0x80	0x81	0x82	0x83	0x84	0x85	0x86	0x87	0x88	0x89	0x8a	0x8b	0x8c	0x8d	0x8e	0x8f
0x90	0x91	0x92	0x93	0x94	0x95	0x96	0x97	0x98	0x99	0x9a	0x9b	0x9c	0x9d	0x9e	0x9f
0xa0	0xa1	0xa2	0xa3	0xa4	0xa5	0xa6	0xa7	0xa8	0xa9	0xaa	0xab	0xac	0xad	0xae	0xaf
0xb0	0xb1	0xb2	0xb3	0xb4	0xb5	0xb6	0xb7	0xb8	0xb9	0xba	0xbb	0xbc	0xbd	0xbe	0xbf
0xc0	0xc1	0xc2	0xc3	0xc4	0xc5	0xc6	0xc7	0xc8	0xc9	0xca	0xcb	0xcc	0xcd	0xce	0xcf
0xd0	0xd1	0xd2	0xd3	0xd4	0xd5	0xd6	0xd7	0xd8	0xd9	0xda	0xdb	0xdc	0xdd	0xde	0xdf
0xe0	0xe1	0xe2	0xe3	0xe4	0xe5	0xe6	0xe7	0xe8	0xe9	0xea	0xeb	0xec	0xed	0xee	0xef
0xf0	0xf1	0xf2	0xf3	0xf4	0xf5	0xf6	0xf7	0xf8	0xf9	0xfa	0xfb	0xfc	0xfd	0xfe	0xff

Read data

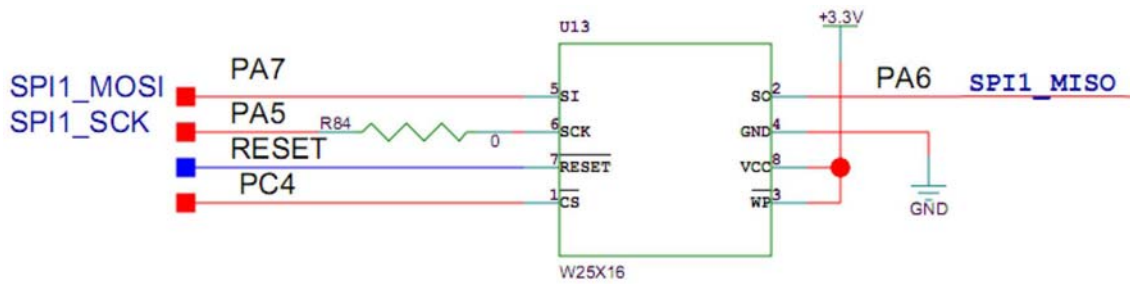
0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27	0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x3A	0x3B	0x3C	0x3D	0x3E	0x3F
0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47	0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F
0x50	0x51	0x52	0x53	0x54	0x55	0x56	0x57	0x58	0x59	0x5A	0x5B	0x5C	0x5D	0x5E	0x5F
0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x67	0x68	0x69	0x6A	0x6B	0x6C	0x6D	0x6E	0x6F
0x70	0x71	0x72	0x73	0x74	0x75	0x76	0x77	0x78	0x79	0x7A	0x7B	0x7C	0x7D	0x7E	0x7F
0x80	0x81	0x82	0x83	0x84	0x85	0x86	0x87	0x88	0x89	0x8A	0x8B	0x8C	0x8D	0x8E	0x8F
0x90	0x91	0x92	0x93	0x94	0x95	0x96	0x97	0x98	0x99	0x9A	0x9B	0x9C	0x9D	0x9E	0x9F
0xA0	0xA1	0xA2	0xA3	0xA4	0xA5	0xA6	0xA7	0xA8	0xA9	0xAA	0xAB	0xAC	0xAD	0xAE	0xAF
0xB0	0xB1	0xB2	0xB3	0xB4	0xB5	0xB6	0xB7	0xB8	0xB9	0xBA	0xBB	0xBC	0xBD	0xBE	0xBF
0xC0	0xC1	0xC2	0xC3	0xC4	0xC5	0xC6	0xC7	0xC8	0xC9	0xCA	0xCB	0xCC	0xCD	0xCE	0xCF
0xD0	0xD1	0xD2	0xD3	0xD4	0xD5	0xD6	0xD7	0xD8	0xD9	0xDA	0xDB	0xDC	0xDD	0xDE	0xDF
0xE0	0xE1	0xE2	0xE3	0xE4	0xE5	0xE6	0xE7	0xE8	0xE9	0xEA	0xEB	0xEC	0xED	0xEE	0xEF
0xF0	0xF1	0xF2	0xF3	0xF4	0xF5	0xF6	0xF7	0xF8	0xF9	0xFA	0xFB	0xFC	0xFD	0xFE	0xFF

#### 4.2.10 SPI Flashテスト

本試験ではSPIを利用してボードで搭載しているFLASH(W25X16)に対して書き込みと読み出し操作を実行して、結果をシリアルポートからPCのハイパーターミナル画面に表示します。

・ ハードウェア設計

STM32F103ZET6 のSPI1 インタフェースとボード搭載している W25X16 と繋いでいます。



GPIO ピンと SPI ピンの対応関係：

W25X16 ピン	GPIO ピン	SPI 信号	説明
SCK	PA5	SPI1_SCK	SPI1 インタフェース信号
SO	PA6	SPI1_MISO	
SI	PA7	SPI1_MISI	
/CS	PC4	-	W25X16 以外、ENC28J60 のイーサネットも SPI インタフェースを利用しているので、ここで GPIO ピンの PC4 をチップセレクト信号として利用します。

SPI1 で W25X16 をアクセスする時、DS1 の LED も点滅します。

- ソフトウェア設計

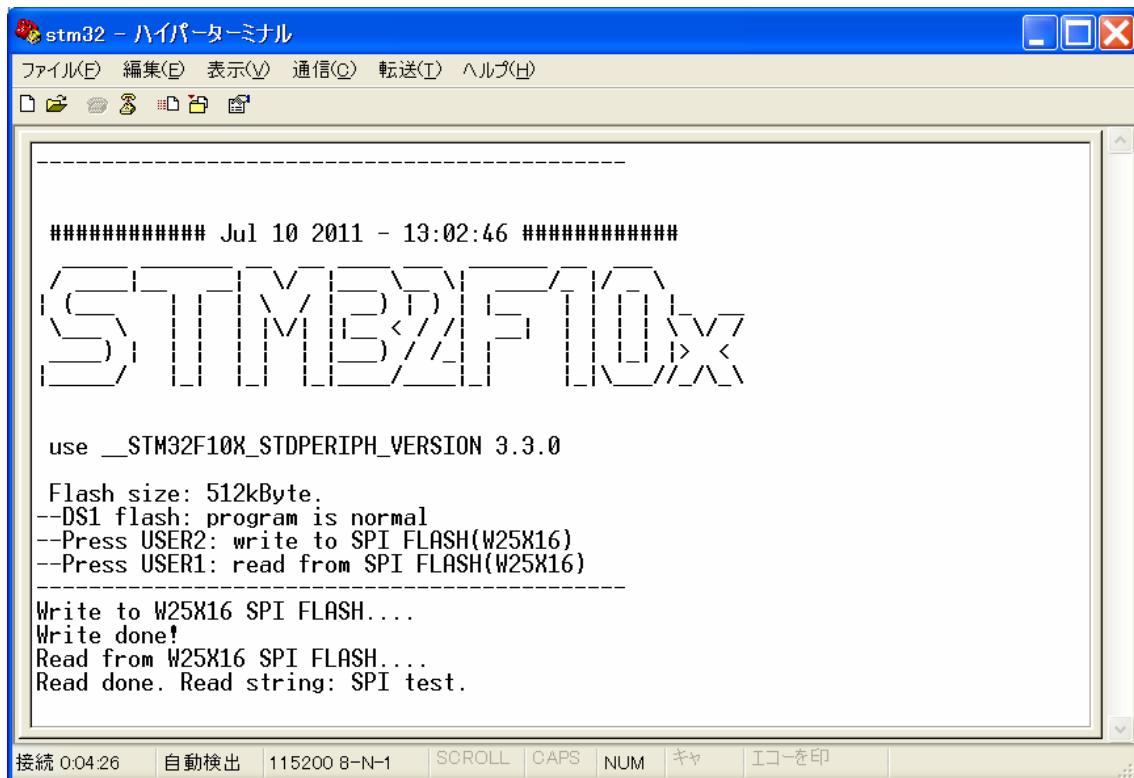
STM32F10x\_StdPeriph\_Lib\_V3.3.0¥Project¥10.SPI\_FLASH¥main.c をご参照ください。

- 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル画面に提示情報が表示されます。USER2 キーを押して FLASH にデータを書き込んで、USER1 キーを押して書き込んだデータを読み出して、テスト結果が表示されます。



```
stm32 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
-----
##### Jul 10 2011 - 13:02:46 #####
STM32F10X

use __STM32F10X_STDPERIPH_VERSION 3.3.0

Flash size: 512kByte.
--DS1 flash: program is normal
--Press USER2: write to SPI FLASH(W25X16)
--Press USER1: read from SPI FLASH(W25X16)
-----
Write to W25X16 SPI FLASH....
Write done!
Read from W25X16 SPI FLASH....
Read done. Read string: SPI test.
-----
接続 0:04:26 自動検出 115200 8-N-1 SCROLL CAPS NUM キャンセル エコーを印
```

#### 4.2.11 SysTickテスト

SysTick はシステムタイマーであり、ハードウェアの割り込みを発生します。本試験では SysTick タイマーで発生した割り込みで LED のコントロールを行います。

SysTick は 24 ビットのカウンタダウンタイマーで、0 になると、STK\_LOAD レジスタから初期値をリセットします。SysTick のコントロール及びステータスレジスタのイネーブルをクリアしないとずっと行います。

- ・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\11.SysTick\main.c をご参照ください。

- ・ 結果

.hex ファイルを書き込んで実行します。ボード上の 4 つの LED が順次点滅します。

#### 4.2.12 SRAMテスト

本試験では STM32 の FSMC バスで SRAM をアクセスします。SRAM を STM32F103ZET6 の FSMC バスの Bank3 にマッピングします。

FSMC バスを初期化後、SRAM の固定アドレスにデータを書き込んだ後読み出します。書き込んだデータと一致するか判断して、LED より結果を表します。

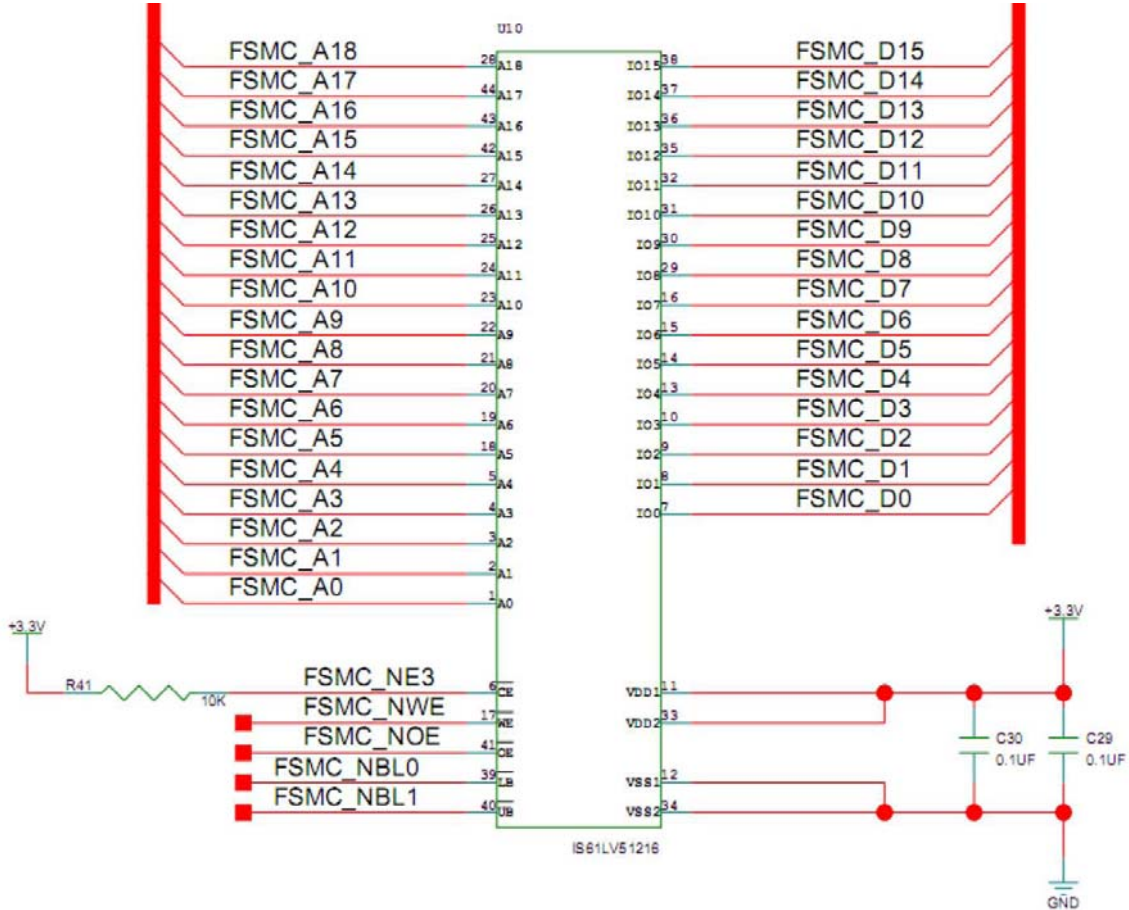
DS1 点滅 プログラム実行中

DS2 点灯 SRAM に書き込んだデータと読み出したデータが一致、SRAM アクセス成功

DS3 点灯 SRAM に書き込んだデータと読み出したデータが不一致、SRAM アクセス失敗

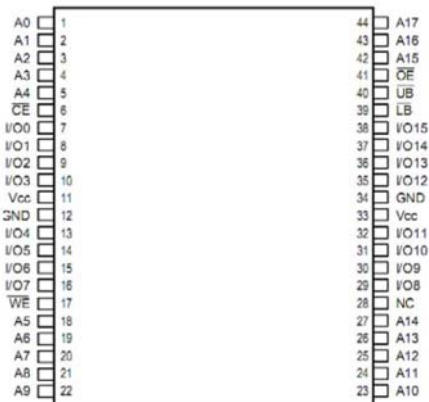
・ハードウェア設計

STM32F103ZET6 の FSMC バスは IS61LV25616LL SRAM と繋いでおります。



本ボードでは 4M の IS61LV25616LL SRAM を搭載していますが、pin-to-pin 完全交換の 8M の IS61LV51216LL SRAM に交換しても良いです。

IS61LV25616 のピン配置：



A0-A17	Address Inputs	$\overline{LB}$	Lower-byte Control (I/O0-I/O7)
I/O0-I/O15	Data Inputs/Outputs	$\overline{UB}$	Upper-byte Control (I/O8-I/O15)
$\overline{CE}$	Chip Enable Input	NC	No Connection
$\overline{OE}$	Output Enable Input	Vcc	Power
$\overline{WE}$	Write Enable Input	GND	Ground

#### ・ ソフトウェア設計

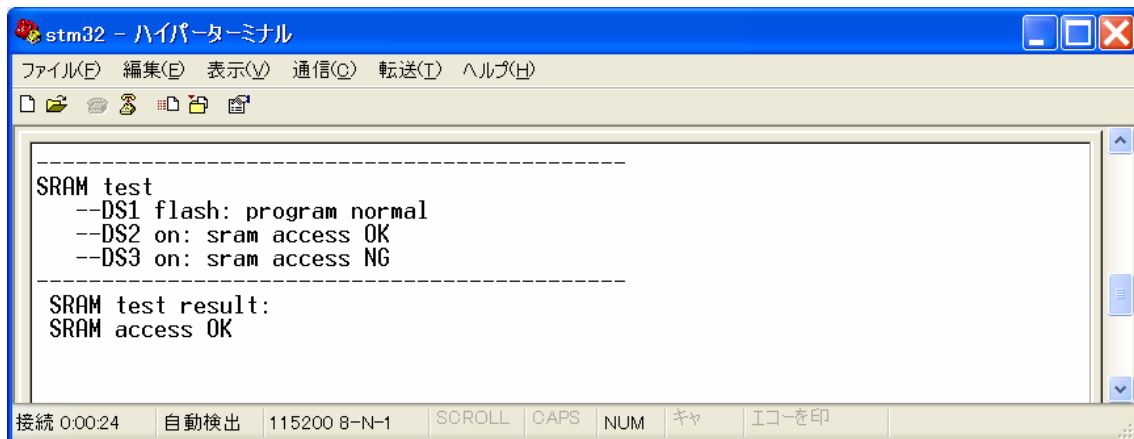
STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\12. SRAM\main.c をご参照ください。

#### ・ 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル画面に提示情報が表示されます。



```
stm32 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
SRAM test
--DS1 flash: program normal
--DS2 on: sram access OK
--DS3 on: sram access NG
-----
SRAM test result:
SRAM access OK
接続 0:00:24 自動検出 115200 8-N-1 SCROLL OAPS NUM キャ エコーを印
```

#### 4.2.13 Nor Flashテスト

本試験ではSTM32のFSMCバスでNor Flashをアクセスします。Nor FlashをSTM32F103ZET6のFSMCバスのBank3にマッピングします。

FSMCバスを初期化後、Nor Flashの固定アドレスにデータを書き込んだ後読み出します。書き込んだデータと一致するか判断して、LEDより結果を表します。

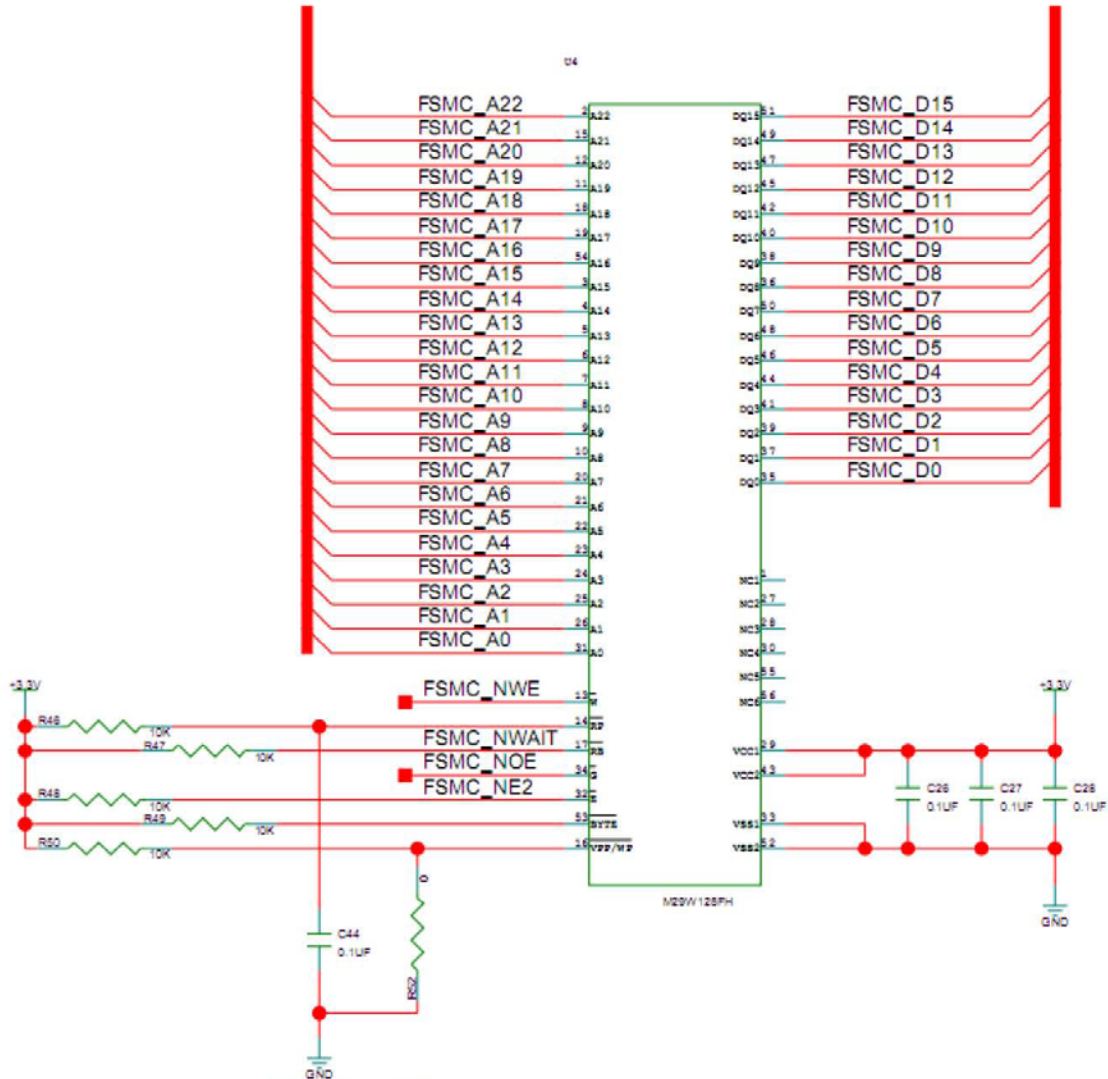
DS1 点滅 プログラム実行中

DS2 点灯 Nor Flashに書き込んだデータと読み出したデータが一致、Nor Flashアクセス成功

DS3 点灯 Nor Flashに書き込んだデータと読み出したデータが不一致、Nor Flashアクセス失敗

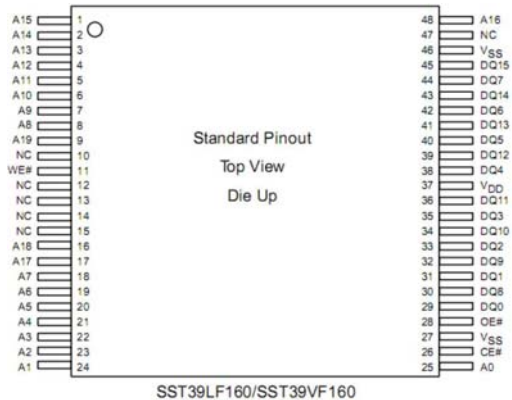
#### ・ ハードウェア設計

STM32F103ZET6のFSMCバスは39VF1601 NOR Flashと繋がります。



本ボードでは SST 社の 14M ビットの SST39VF160 Nor Flash を搭載していますが、pin-to-pin 完全交換のもっと大きい容量の SST39VF3201/SST39VF6401 に交換しても良いです。

SST39VF160 のピン配置：



PIN DESCRIPTION

Symbol	Pin Name	Functions
A <sub>19</sub> -A <sub>0</sub>	Address Inputs	To provide memory addresses. During Sector-Erase A <sub>19</sub> -A <sub>11</sub> address lines will select the sector. During Block-Erase, A <sub>19</sub> -A <sub>15</sub> address line will select the block.
DQ <sub>15</sub> -DQ <sub>0</sub>	Data Input/output	To output data during Read cycles and receive input data during Write cycles. Data is internally latched during a Write cycle. The outputs are in tri-state when OE# or CE# is high.
CE#	Chip Enable	To activate the device when CE# is low
OE#	Output Enable	To gate the data output buffers
WE#	Write Enable	To control the Write operations
V <sub>DD</sub>	Power Supply	To provide power supply voltage: 3.0-3.6V for SST39LF160 2.7-3.6V for SST39VF160
V <sub>SS</sub>	Ground	
NC	No Connection	Unconnected pins

・ ソフトウェア設計

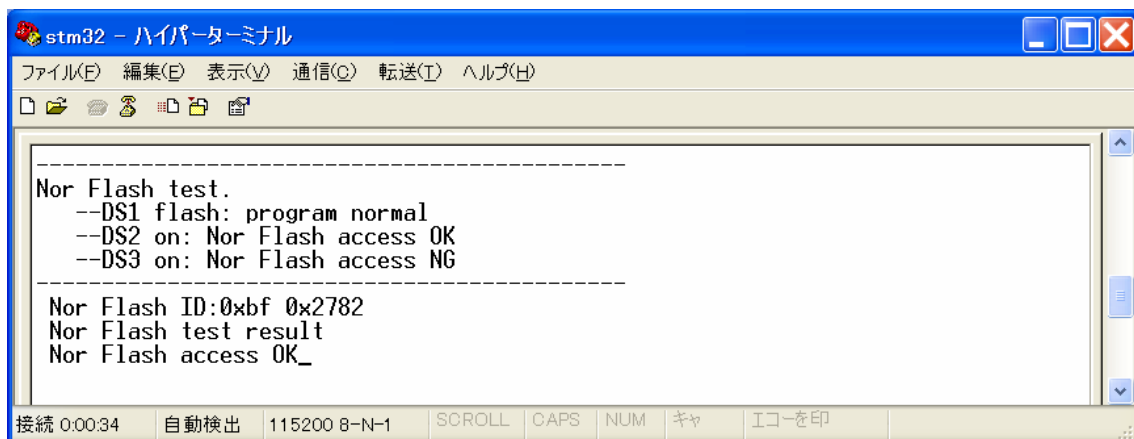
STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\13. NOR FLASH\main.c をご参照ください。

・ 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル画面に提示情報が表示されます。



```

stm32 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
-----
Nor Flash test.
--DS1 flash: program normal
--DS2 on: Nor Flash access OK
--DS3 on: Nor Flash access NG
-----
Nor Flash ID:0xbf 0x2782
Nor Flash test result
Nor Flash access OK_
接続 0:00:34   自動検出   115200 8-N-1   SCROLL OAPS NUM   キャ   エコーを印
  
```

4.2.14 Nand Flashテスト

本試験ではSTM32のFSMCバスでNand Flashをアクセスします。

FSMCバスを初期化後、Nand Flashの固定アドレスにデータを書き込んだ後読み出します。書き込んだデータと一致するか判断して、LEDより結果を表します。

DS1 点滅 プログラム実行中

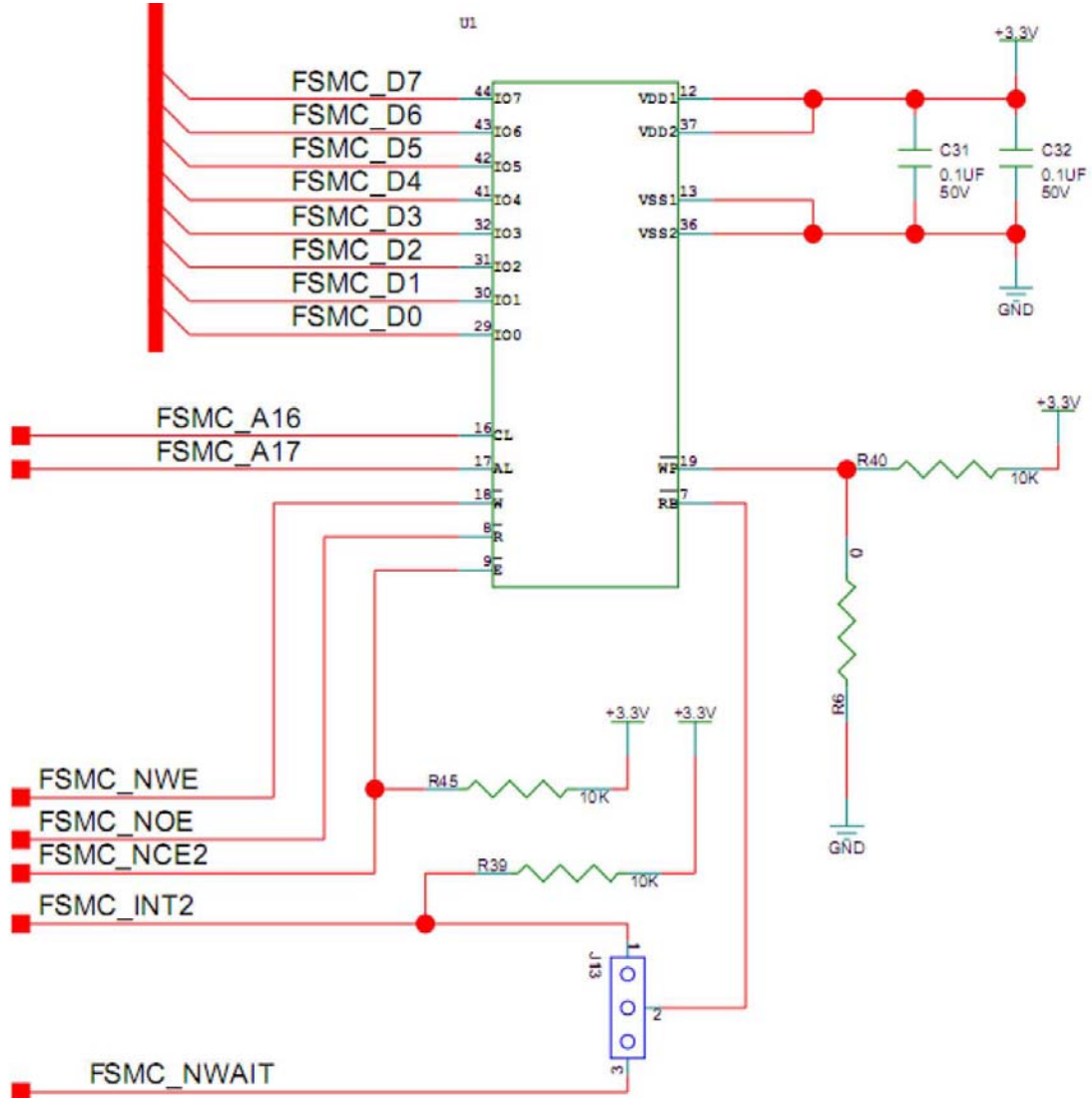
DS2 点灯 Nand Flashに書き込んだデータと読み出したデータが一致、Nand Flashアクセス成功

DS3 点灯 Nand Flashに書き込んだデータと読み出したデータが不一致、Nand Flashアクセス失敗

DS4 点灯 Nand Flash の ID の読み出し失敗

・ハードウェア設計

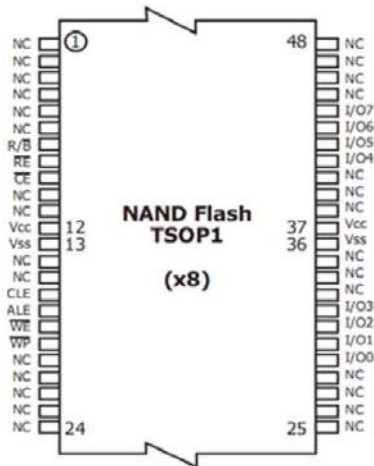
STM32F103ZET6 の FSMC バスは HY27UF081G2A Nand Flash と繋がります。



HY27UF081G2A は Hynix 社製の 1Gbit Nand Flash。

HY27UF081G2A のピン配置：





IO15 - IO8	Data Input / Outputs (x16 only)
IO7 - IO0	Data Inputs / Outputs
CLE	Command latch enable
ALE	Address latch enable
CE	Chip Enable
RE	Read Enable
WE	Write Enable
WP	Write Protect
R/B	Ready / Busy
Vcc	Power Supply
Vss	Ground
NC	No Connection

・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\14.NAND\_FLASH\main.c をご参照ください。

・ 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル画面に提示情報が表示されます。

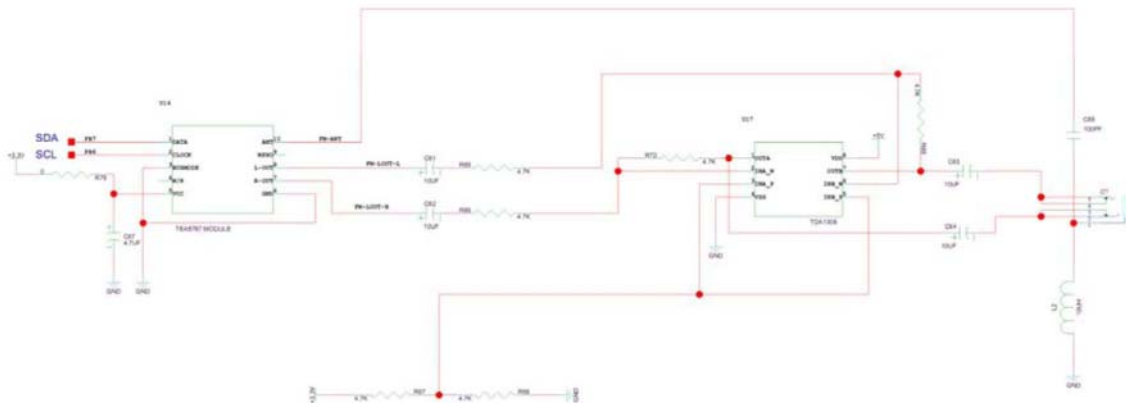
```
stm32 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
-----
Nand Flash test.
--DS1 flash: program normal
--DS2 on: Nand Flash access OK
--DS3 on: Nand Flash access NG
--DS4 on: fail to get Nand Flash ID
-----
Nand Flash ID:0xad    0xf1    0x80    0x1d
Nand Flash test result
Nand Flash access OK
-----
接続 0:00:18   自動検出   115200 8-N-1   SCROLL  OAPS  NUM  キャ  エコーを印
```

#### 4.2.15 FM Tunerテスト

本試験では STM32 で TEA5767 チューナーモジュールをコントロールして、自動サーチして一つのバンドを選んで受信した音声をイヤリングから再生します。

##### ・ハードウェア設計

STM32 の PB6、PB7 で TEA5767 をアクセスします。PB6、PB7 は STM32 の I2C バスの GPIO ピンです。本試験では PB6、PB7 ソフトモード I2C インタフェースで TEA5767 をアクセスします。



チューナーモジュールの ANT ピンはオーディオブロックと繋いでいるので、イヤホンを挿入するとアンテナとして利用できます。

TEA5767 外形図：



- ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\15.FMTuner\main.c をご参照ください。

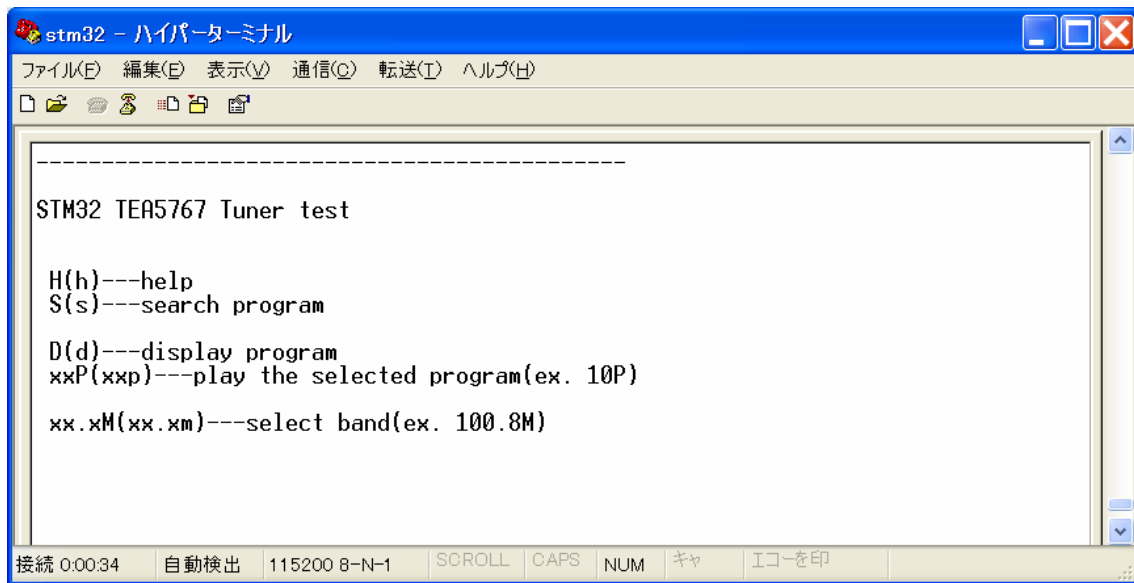
- 結果

イヤホンをボードの J27 に挿入します。

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル画面に提示情報が表示されます。



```
STM32 TEA5767 Tuner test

H(h)---help
S(s)---search program

D(d)---display program
xxP(xxP)---play the selected program(ex. 10P)

xx.xM(xx.xM)---select band(ex. 100.8M)
```

画面提示通り各操作できます。例えば、PC のキーボードから S キーを押下すると自動サーチします。1P を押すと一番目のチャンネルを選択します。

#### 4.2.16 2.4G無線通信テスト

本試験では 2.4G 無線モジュール nRF24L01 (オプション) 接続している二つのボードが必要です。電源入れて nRF24L01 モジュールが繋いでいるか測定して、なかったら、接続確認の提示が表示されます。あれば、次にワークモードが送信或いは受信かの選択を提示します。ボードの USER1 キーを押すと受信モードに設定し、USER2 キーを押すと送信モードに設定します。

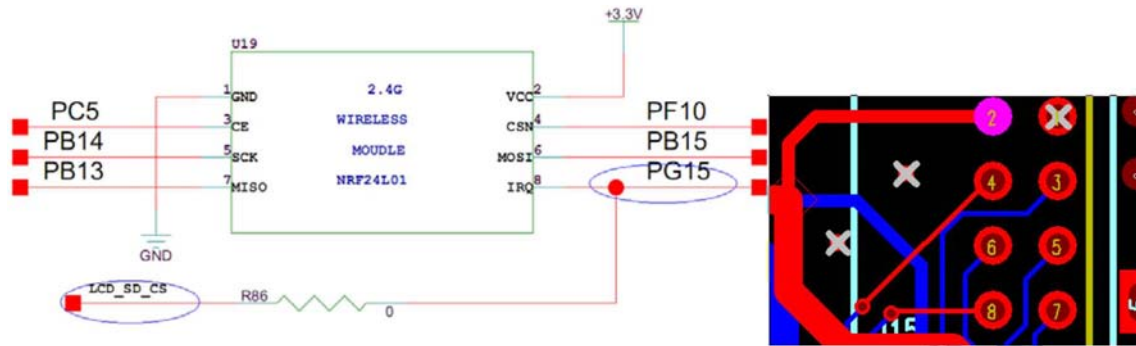
2.4G 無線モジュール nRF24L01 については、下記 URL をご参照ください。

<http://www.csun.co.jp/SHOP/2009061924.html>

- ハードウェア設計

STM32 の SPI バスで nRF24L01 をコントロールします。

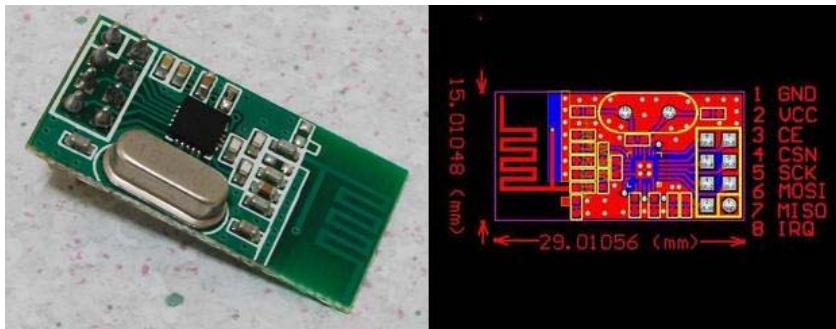
回路図：



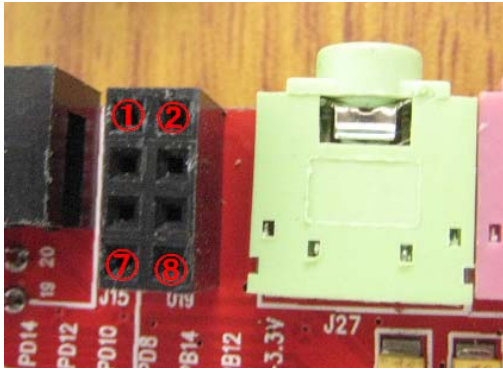
GPIO ピンと 2.4G 無線モジュール nRF24L01 ピン対応関係：

nRF24L01 ピン	GPIO ピン	PCB ピン	SPI 信号	説明
SCK	PB14	5	SPI2_SCK	SPI2 インタフェース信号
MISO	PB13	7	SIP2_MISO	
MISI	PB15	6	SIP2_MISI	
CE	PC5	3	-	SPI2 は 2.4G 無線モジュール以外、DA チップの PCM1770 または TFT のタッチパネルも共用していますので、PC5 を nRF24L01 の SPI2 の CS 信号として利用します。
CSN	PF10	4	-	nRF24L01 のモジュール選択信号
IRQ	PG15	8	-	nRF24L01 の割り込み信号 ※PG15 は nRF24L01 の割り込み出力以外、LCD モジュール上の SD カードの CS 信号とも繋いでおりますので、nRF24L01 と LCD モジュール上の SD カードは同時に利用できません。
GND	-	1	-	GND 信号
VCC	-	2	-	電源入力

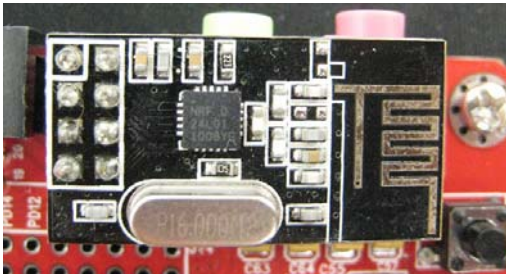
nRF24L01 無線モジュールのピン配列：



ボードのピン配列：



nRF24L01 とボード接続した様子：



- ・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0¥Project¥16.2.4GWireless¥main.c をご参照ください。

- ・ 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

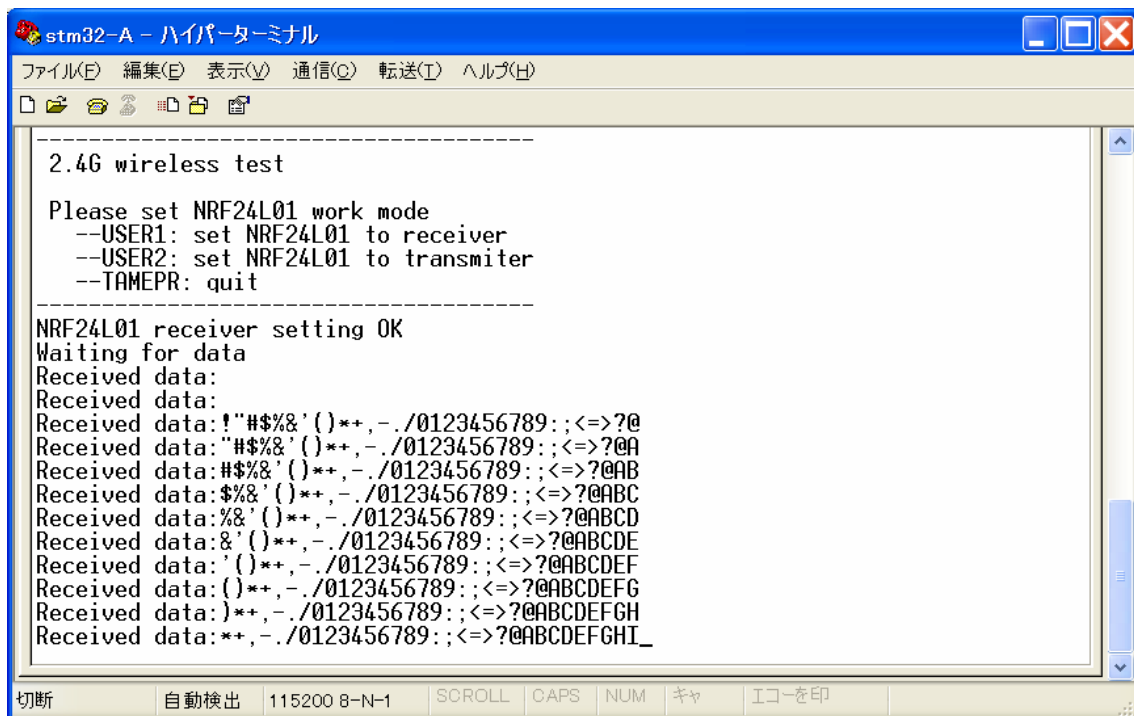
.hex ファイルを書き込んで実行します。

ハイパーターミナル画面に提示情報が表示されます。

一つのボードの USER1 キーを押して受信モードに設定します。もう一つのボードでは USER2 キーを押して送信モードに設定します。

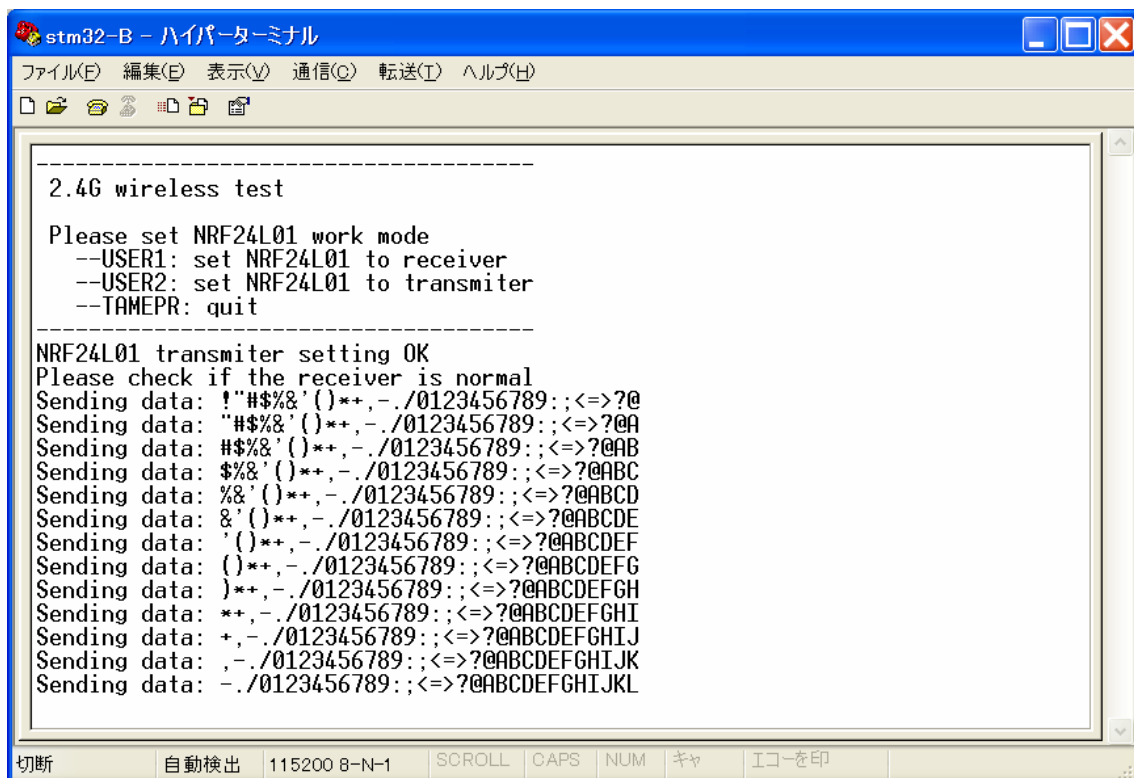
※本試験実行時は LCD を取り外してください。

それぞれの画面提示は下記：



```
stm32-A - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
-----
2.4G wireless test

Please set NRF24L01 work mode
--USER1: set NRF24L01 to receiver
--USER2: set NRF24L01 to transmitter
--TAMEPR: quit
-----
NRF24L01 receiver setting OK
Waiting for data
Received data:
Received data:
Received data: !"#%&'()*+,-./0123456789:;<=>?@
Received data: "#$%&'()*+,-./0123456789:;<=>?@A
Received data: #$%&'()*+,-./0123456789:;<=>?@AB
Received data: %&'()*+,-./0123456789:;<=>?@ABC
Received data: &'()*+,-./0123456789:;<=>?@ABCD
Received data: '()*+,-./0123456789:;<=>?@ABCDE
Received data: ()+,-./0123456789:;<=>?@ABCDEF
Received data: )+,-./0123456789:;<=>?@ABCDEFG
Received data: *+,-./0123456789:;<=>?@ABCDEFGH
Received data: **,-./0123456789:;<=>?@ABCDEFGHI_
-----
切断 自動検出 115200 8-N-1 SCROLL CAPS NUM キャ Iコーを印
```



```
stm32-B - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
-----
2.4G wireless test

Please set NRF24L01 work mode
--USER1: set NRF24L01 to receiver
--USER2: set NRF24L01 to transmitter
--TAMEPR: quit
-----
NRF24L01 transmitter setting OK
Please check if the receiver is normal
Sending data: !"#%&'()*+,-./0123456789:;<=>?@
Sending data: "#$%&'()*+,-./0123456789:;<=>?@A
Sending data: #$%&'()*+,-./0123456789:;<=>?@AB
Sending data: %&'()*+,-./0123456789:;<=>?@ABC
Sending data: &'()*+,-./0123456789:;<=>?@ABCD
Sending data: '()*+,-./0123456789:;<=>?@ABCDE
Sending data: ()+,-./0123456789:;<=>?@ABCDEF
Sending data: )+,-./0123456789:;<=>?@ABCDEFG
Sending data: *+,-./0123456789:;<=>?@ABCDEFGH
Sending data: **,-./0123456789:;<=>?@ABCDEFGHI
Sending data: +,-./0123456789:;<=>?@ABCDEFGHIJ
Sending data: ,-./0123456789:;<=>?@ABCDEFGHIJK
Sending data: -./0123456789:;<=>?@ABCDEFGHIJKL
-----
切断 自動検出 115200 8-N-1 SCROLL CAPS NUM キャ Iコーを印
```

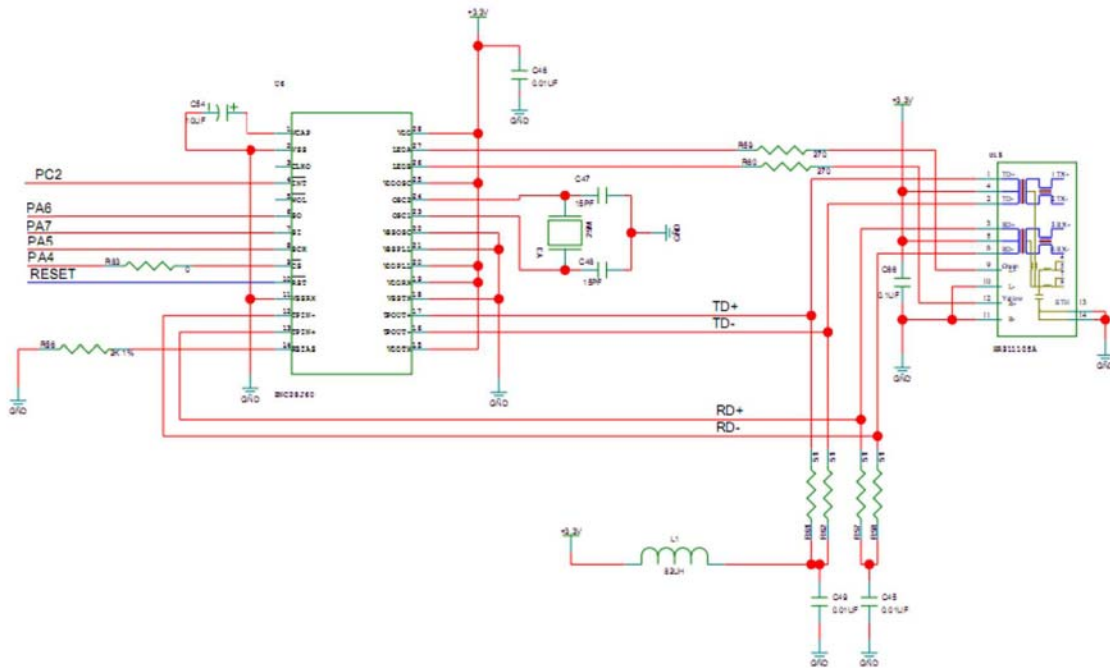
#### 4. 2. 17 LAN通信テスト

本ボードではSPI インタフェースの ENC28J60 コントローラーで 10M イーサネット機能を実

現しております。ENC28J60 と STM32F103ZET6 は SPI1 でアクセスし、10Mbps をサポートします。

・ハードウェア設計

回路図：



ENC28J60 ピンとボードの GPIO ピン対応：

ENC28J60 ピン	GPIO ピン	SPI 信号	説明
/CS	PA4	SPI1_NSS	SPI1 インタフェース信号
SCK	PA5	SPI1_	
SO	PA6	SPI1_	
SI	PA7	SPI1_	
/INT	PC2	-	ENC28J60 の割込み入力

・ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\18\_ENC28J60\main.c をご参照ください。

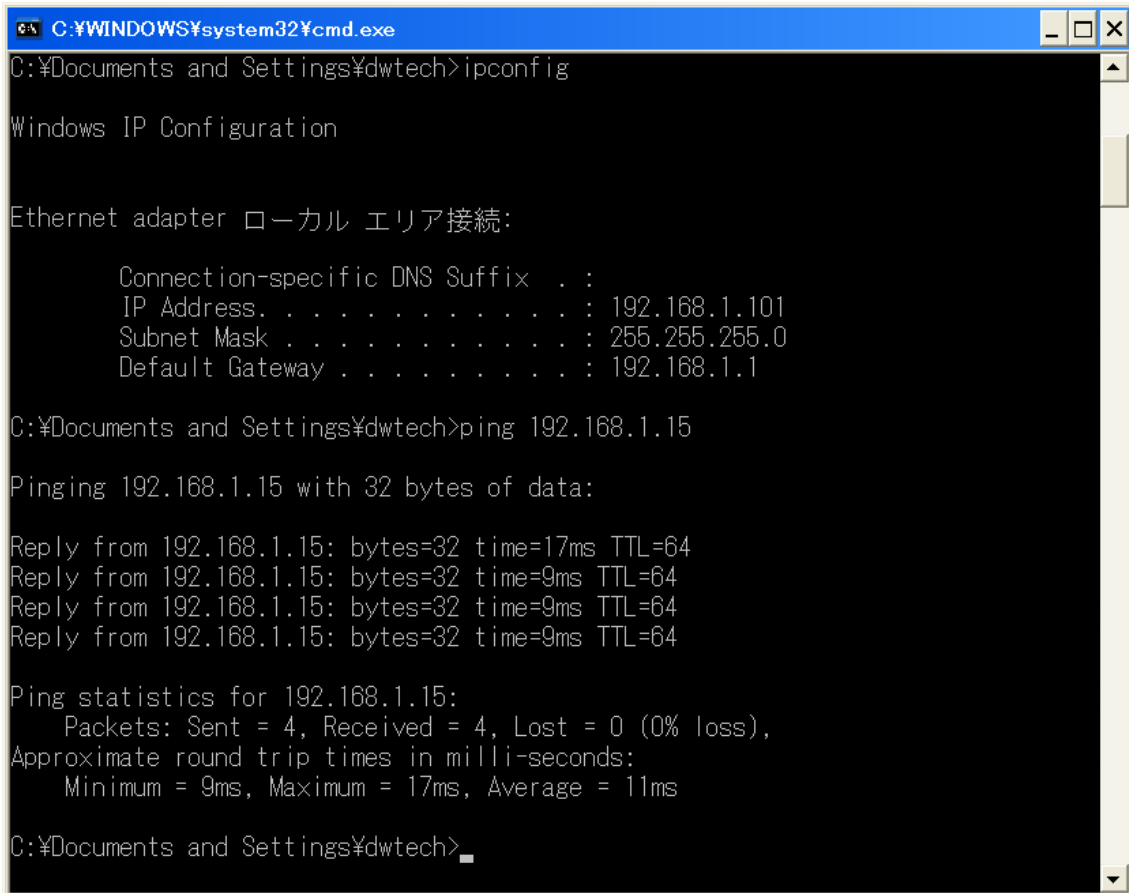
・結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

クロス LAN ケーブルでボードと PC を繋ぎます（或いは同じローカルエリアに接続する）。本試験では例として、PC の IP アドレスを 192.168.1.101 に設定し、ボードの IP アドレスは 192.168.1.15 に設定します。

.hex ファイルを書き込んで実行します。

PC 側で IP アドレスを確認し、ボードとの接続を Ping コマンドで確認します。



```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\dwtech>ipconfig

Windows IP Configuration

Ethernet adapter ローカル エリア接続:

    Connection-specific DNS Suffix  . :
    IP Address. . . . . : 192.168.1.101
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.1.1

C:\Documents and Settings\dwtech>ping 192.168.1.15

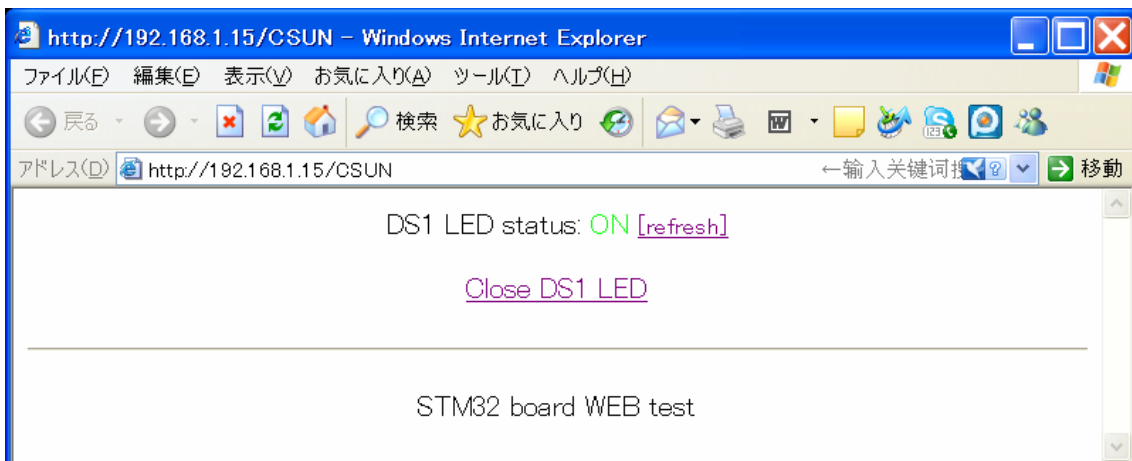
Pinging 192.168.1.15 with 32 bytes of data:

Reply from 192.168.1.15: bytes=32 time=17ms TTL=64
Reply from 192.168.1.15: bytes=32 time=9ms TTL=64
Reply from 192.168.1.15: bytes=32 time=9ms TTL=64
Reply from 192.168.1.15: bytes=32 time=9ms TTL=64

Ping statistics for 192.168.1.15:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 9ms, Maximum = 17ms, Average = 11ms

C:\Documents and Settings\dwtech>
```

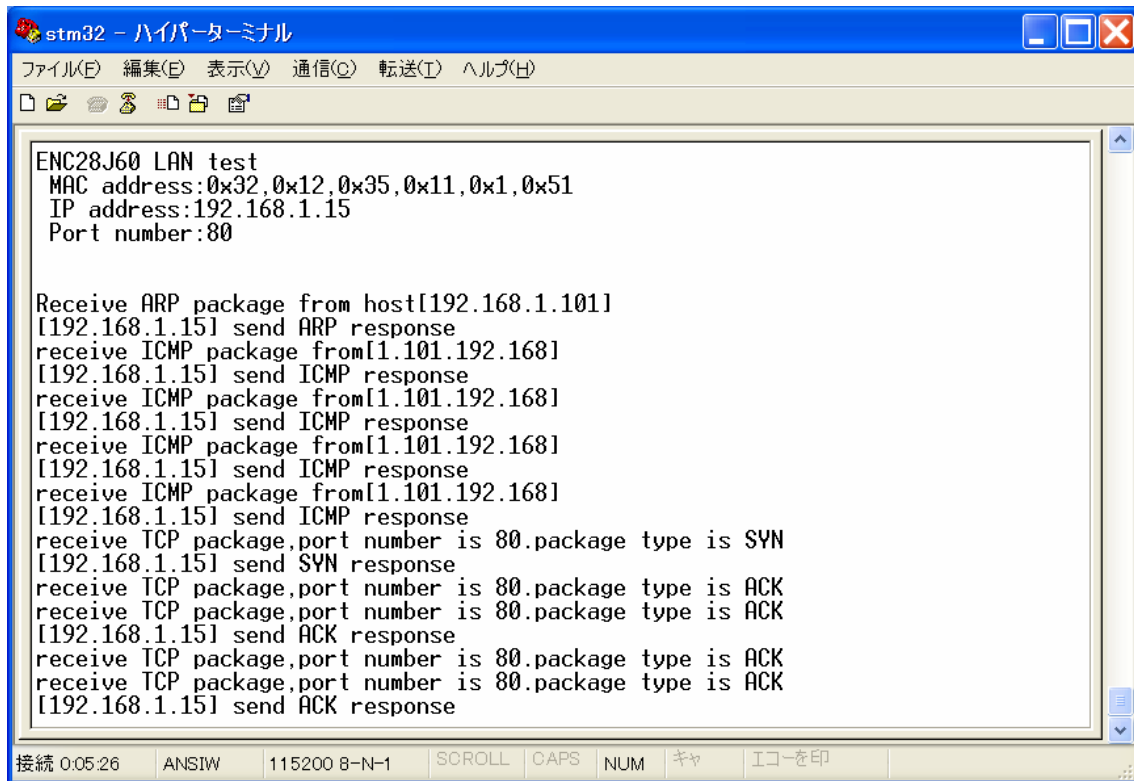
PC側のIEを開いて、<http://192.168.1.15/CSUN> を入力してHTTPアクセスができます。



画面上の操作で DS1 LED を点灯、消灯の制御ができます。

上記操作する時、ハイパーターミナル画面に提示情報が表示されます。





```
stm32 - ハイパーターミナル
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)
ENC28J60 LAN test
MAC address:0x32,0x12,0x35,0x11,0x1,0x51
IP address:192.168.1.15
Port number:80

Receive ARP package from host[192.168.1.101]
[192.168.1.15] send ARP response
receive ICMP package from[1.101.192.168]
[192.168.1.15] send ICMP response
receive ICMP package from[1.101.192.168]
[192.168.1.15] send ICMP response
receive ICMP package from[1.101.192.168]
[192.168.1.15] send ICMP response
receive ICMP package from[1.101.192.168]
[192.168.1.15] send ICMP response
receive TCP package,port number is 80.package type is SYN
[192.168.1.15] send SYN response
receive TCP package,port number is 80.package type is ACK
receive TCP package,port number is 80.package type is ACK
[192.168.1.15] send ACK response
receive TCP package,port number is 80.package type is ACK
receive TCP package,port number is 80.package type is ACK
[192.168.1.15] send ACK response

接続 0:05:26 ANSIW 115200 8-N-1 SCROLL CAPS NUM キャ Iコーを印
```

#### 4.2.18 SDカードテスト

本試験では USB インタフェースでボードに挿入した SD カードをアクセスします。STM32 で SD カードリーダーの機能を実現します。本試験は ST の Mass\_Storage のサンプルを参照して実現しております。SDIO モードで SD カードをアクセスします。

試験の簡単なプロセス：シリアルポートと GPIO ピンを初期化して、SDIO インタフェースと USB インタフェースを初期化します。SD カード挿入しているか検査して挿入したら USB の配置を始まり、終わりましたら PC 上 USB ディスクが発見できます。

DS2 と DS3 LED は初期化中を示して、DS1 LED は USB から SD カードをアクセス中を示します。

##### ・ ハードウェア設計

本試験で利用するハードウェアソース：

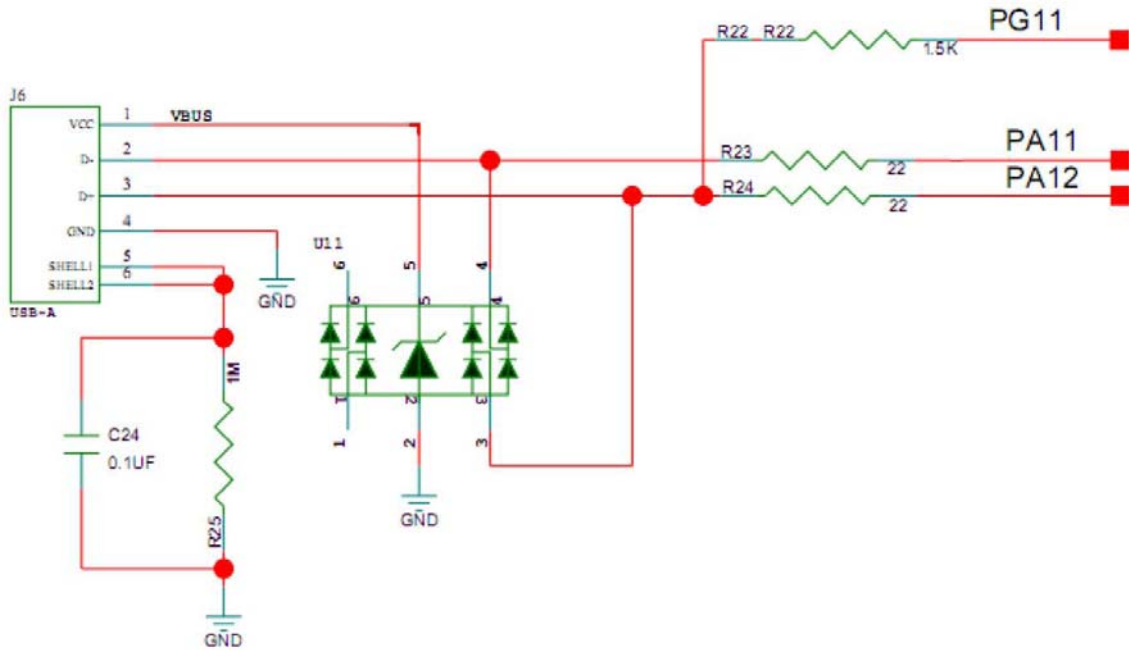
シリアルポート 1：試験中の提示情報の出力。

LED：プログラムの実行状態の表示。

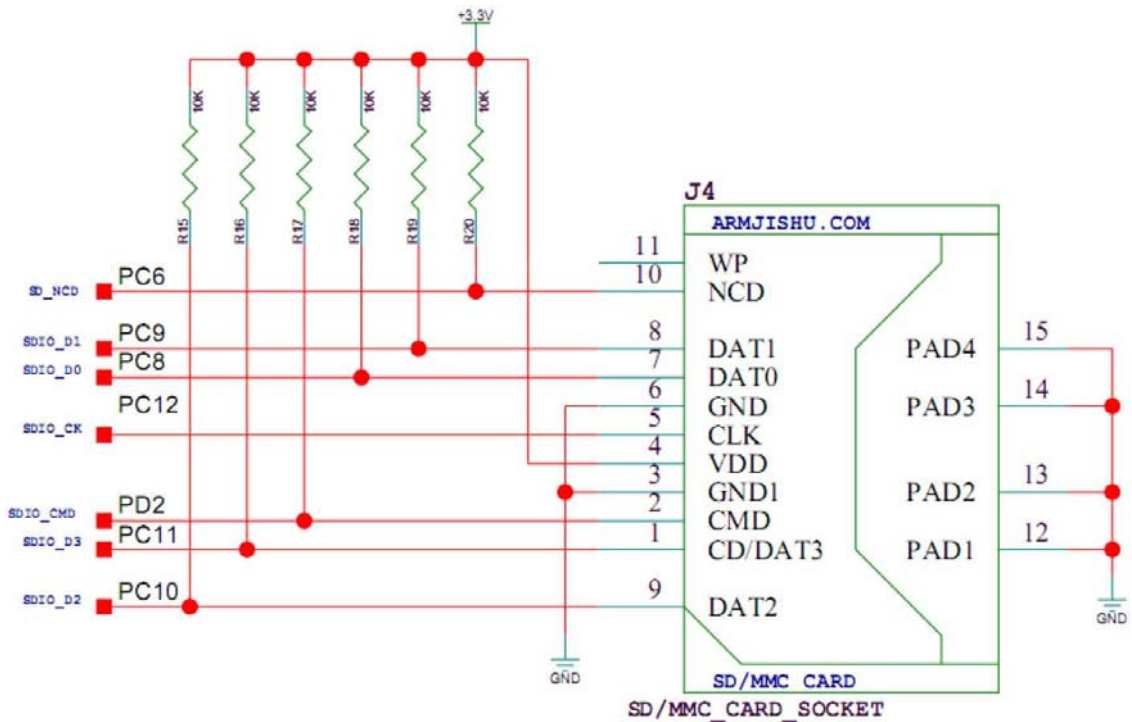
SD カード：最大 2G 容量の SD カードをサポート。

USB インタフェース：ボードの USB ポートと PC を繋ぎ、SD カードをボードに挿入して、本試験を実行すると、PC 上 USB ディスクが発見され、読み出し、書き込みの操作ができます。

USB インタフェース回路図：



SD カードインタフェース回路図：



本ボードでは、SDIO インタフェースで SD カードをアクセスしております。SDIO は 4 ビットデータモードで実現しております。各ピンの機能は：

ピン	機能	説明
PC6	SC_NCD	SD カード検査
PC8	SDIO_D0	メディアカード/SD/SDIO カードデータ

PC9	SDIO_D1	
PC10	SDIO_D2	
PC11	SDIO_D3	
PC12	SDIO_CK	メディアカード/SD/SDIO カードクロック
PD2	SDIO_CMD	メディアカード/SD/SDIO カードコマンド、レスポンス信号

- ソフトウェア設計

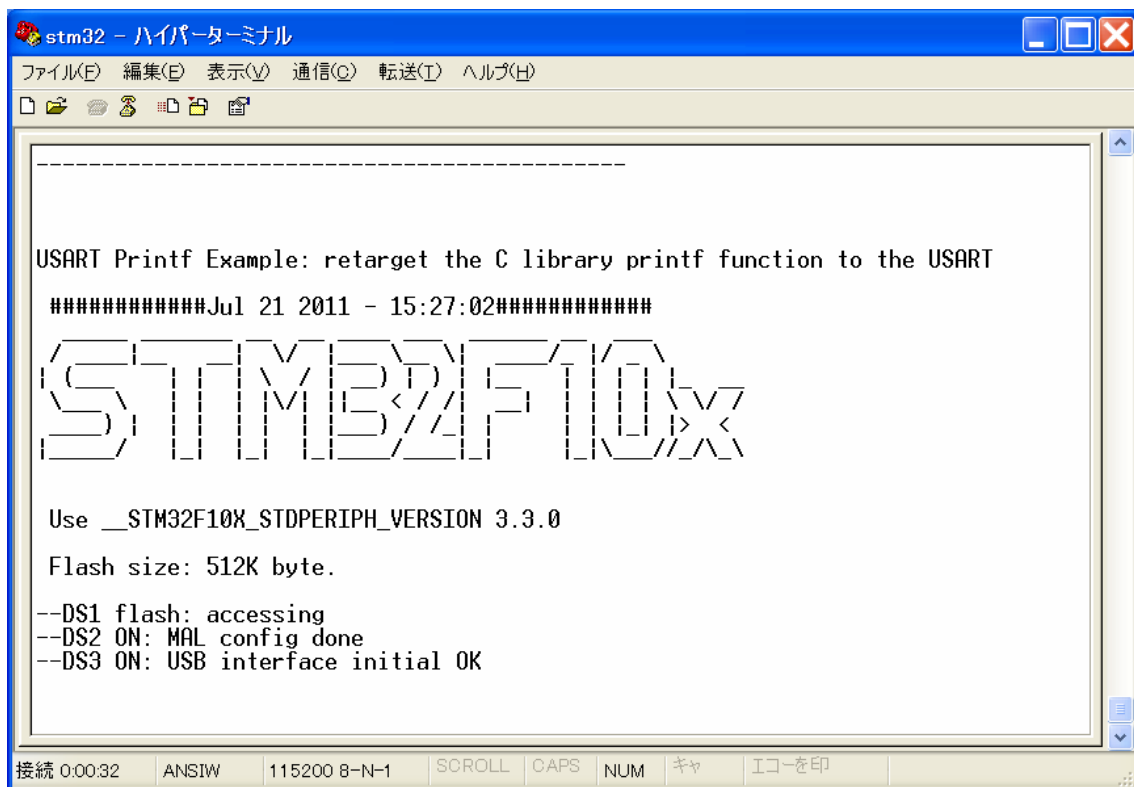
STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\20\_SDI\main.c をご参照ください。

- 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

.hex ファイルを書き込んで実行します。

ハイパーターミナル上の提示画面は下記：



```
-----  
USART Printf Example: retarget the C library printf function to the USART  
#####Jul 21 2011 - 15:27:02#####  
STM32F10X  
Use __STM32F10X_STDPERIPH_VERSION 3.3.0  
Flash size: 512K byte.  
--DS1 flash: accessing  
--DS2 ON: MAL config done  
--DS3 ON: USB interface initial OK
```

SD カードをボードに挿入し、USB ケーブルでボードと PC を接続します。PC 上 USB 設備が発見され、書き込みと読み出し操作ができます。

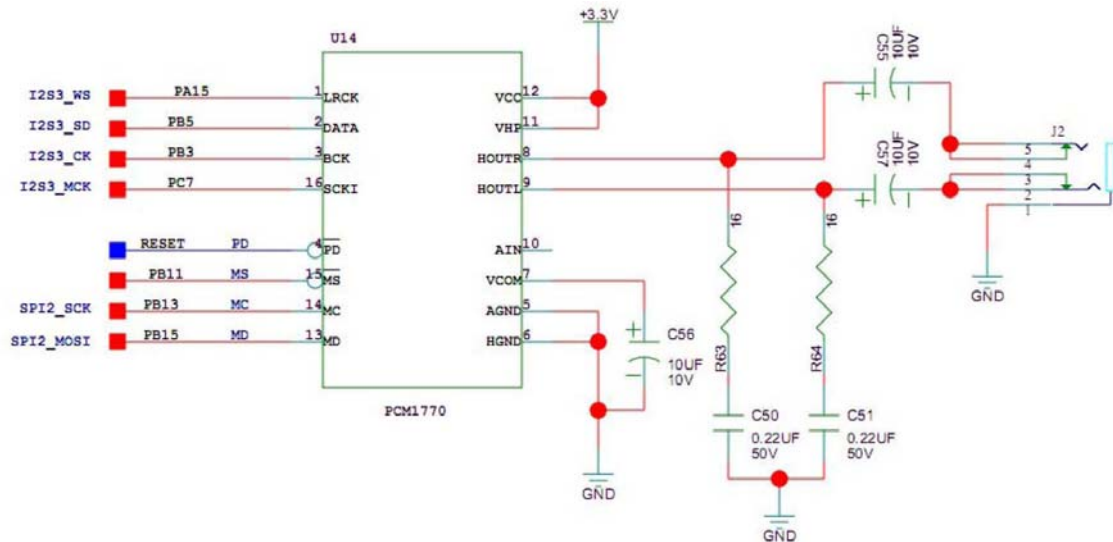


※SD\_CARD は試験中ボードに挿入した SD カードのラベルです。

#### 4.2.19 Audio Playテスト

本試験では DA チップ PCM1770 を初期化後、事前に MCU 内部に保存している音楽ファイルを読み出して、フォーマットより I2S3 インタフェースのパラメータを設定し、割込みモードで I2S3 インタフェースから音楽ファイルを繰り返してプレイします。

- ハードウェア設計



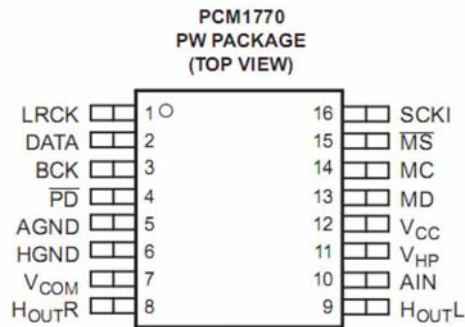
PCM1770 は I2S3 インタフェースで STM32F103ZET6 と繋いでいます。音声信号は I2S3 インタフェースで PCM1770 に転送されて音声信号に変換して出します。関連ピンについて：

GPIO ピン	機能	説明
PA15	I2S3_WS	左右チャンネルクロック、周波数はサンプリング周波数と同じ。この信号は JTAG の JTDI 信号と同じピンを共用しております。
PB3	I2S3_CK	シリアルビットクロック。この信号は JTAG の JTDO 信号と同じピンを共用しております。
PB5	I2S3_SD	シリアル音声信号
PC7	I2S3_MCK	システムクロック入力

STM32F103ZET6 は SPI2 インタフェースで PCM1770 をアクセスして、内部のデータを読み出し、配置します。関連ピンについて：

GPIO ピン	機能	説明
PB11	SPI2_NSS	SPI2 インタフェースのセレクト信号
PB13	SPI2_SCK	SPI2 インタフェースの SCK クロック信号
PB15	SPI2_MOSI	SPI2 インタフェースの MOSI 信号

PCM1770 チップについて：



TERMINAL NAME	NO.	I/O	DESCRIPTION
AGND	5	-	Analog ground. This is a return for V <sub>CC</sub> .
AIN	10	I	Monaural analog signal mixer input. The signal can be mixed with the outputs of the L- and R-channel DACs.
BCK	3	I/O	Serial bit clock. Clocks the individual bits of the audio data input, DATA. In the slave interface mode, this clock is input from an external device. In the master interface mode, the PCM1770 device generates the BCK output to an external device.
DATA	2	I	Serial audio data input
HGND	6	-	Analog ground. This is a return for V <sub>HP</sub> .
H <sub>OUTL</sub>	9	O	L-channel analog signal output of the headphone amplifiers
H <sub>OUTR</sub>	8	O	R-channel analog signal output of the headphone amplifiers
LRCK	1	I/O	Left and right clock. Determines which channel is being input on the audio data input, DATA. The frequency of LRCK must be the same as the audio sampling rate. In the slave interface mode, this clock is input from an external device. In the master interface mode, the PCM1770 device generates the LRCK output to an external device.
MC	14	I	Mode control port serial bit clock input. Clocks the individual bits of the control data input, MD.
MD	13	I	Mode control port serial data input. Controls the operation mode on the PCM1770 device.
MS	15	I	Mode control port select. The control port is active when this terminal is low.
PD	4	I	Reset input. When low, the PCM1770 device is powered down, and all mode control registers are reset to default settings.
SCKI	16	I	System clock input
V <sub>CC</sub>	12	-	Power supply for all analog circuits except the headphone amplifier.
V <sub>COM</sub>	7	-	Decoupling capacitor connection. An external 10-μF capacitor connected from this terminal to analog ground is required for noise filtering. Voltage level of this terminal is 0.5 V <sub>HP</sub> nominal.
V <sub>HP</sub>	11	-	Analog power supply for the headphone amplifier circuits. The voltage level must be the same as V <sub>CC</sub> .

### ・ ソフトウェア設計

STM32F10x\_StdPeriph\_Lib\_V3.3.0\Project\21.MP3Player\main.c をご参照ください。

### ・ 結果

クロスシリアルケーブルでボードの COM1 と PC を繋いで、ハイパーターミナルを起動して、下記パラメータを設定します：115200、8、なし、1、なし。

※本テストで利用する I2S3 は JTAG のインタフェースを共用している部分がありますので、ダウンロード及びデバッグの時 JTAG ではなく、SWD インタフェースをご利用ください。

.hex ファイルを書き込んで実行します。

イヤホンをボードの J2 に挿入すると、繰り返している音楽が聞こえます。

ハイパーターミナル上の提示画面は下記：



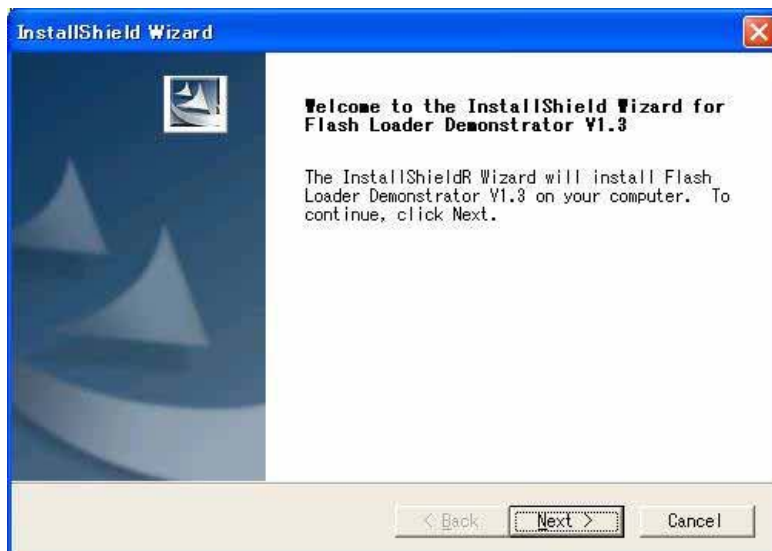
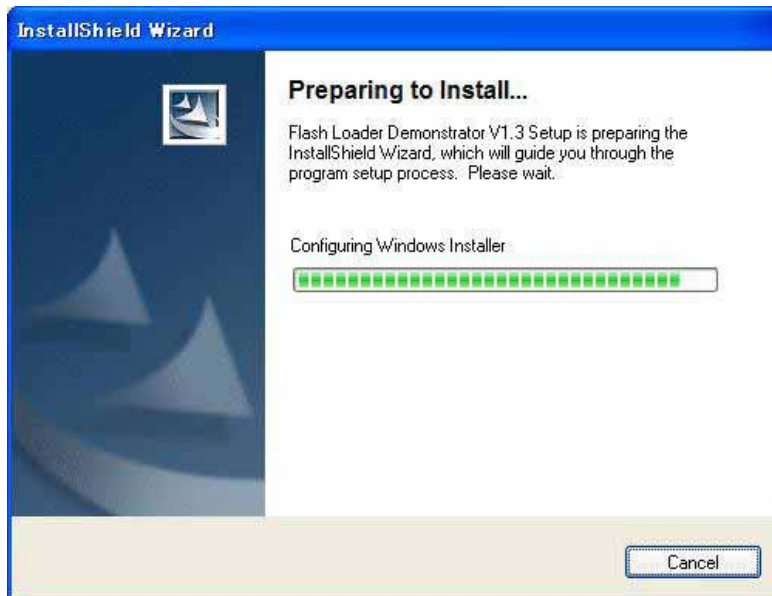
```
-----  
Audio play test  
-----  
Serial port initial finish  
PCM1770 SPI interface initial finish  
AudioFile_Init  
WaveFileStatus Valid_WAVE_File  
I2S_Config  
  
CODEC_Config  
ResetVar_SendDummyData  
GetVar_SendDummyData=0  
  
MusicBufferCurrentPos = 0!  
  
MusicBufferCurrentPos = 0!
```

## 第五章 実行ファイルの書き込み

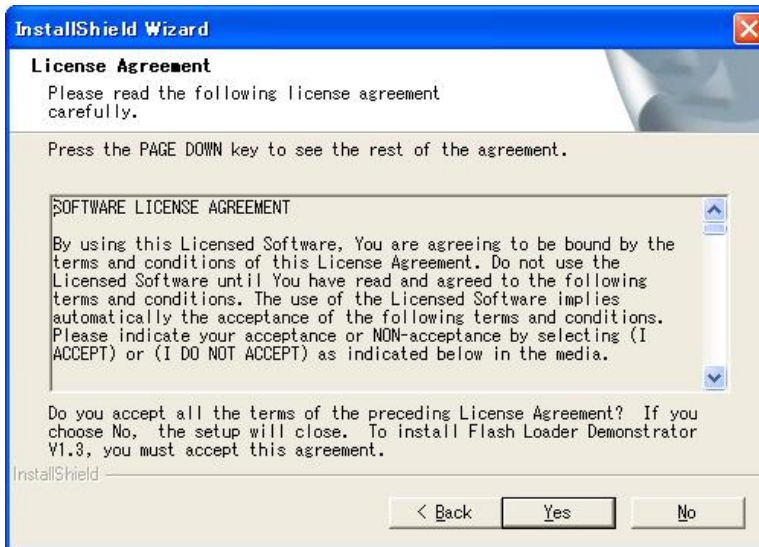
### 5.1 シリアルポートで書き込む

Flash Loader Demonstrator\_V1.3\_Setup.exeはシリアルポートでSTM32マイコンのFlashを更新するツールである。

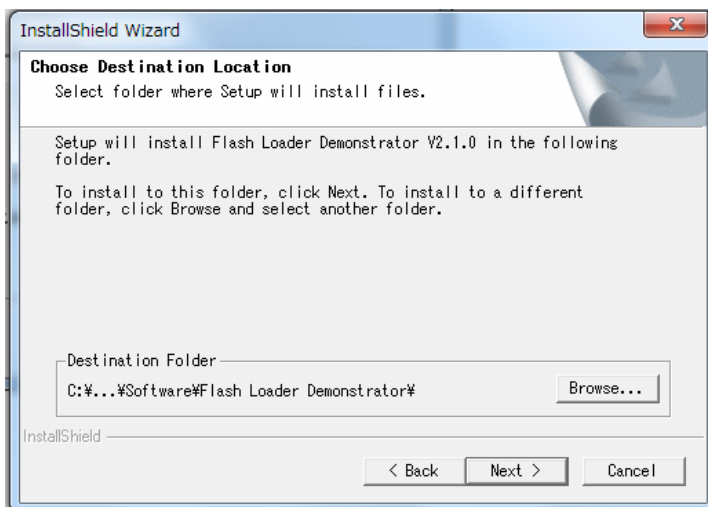
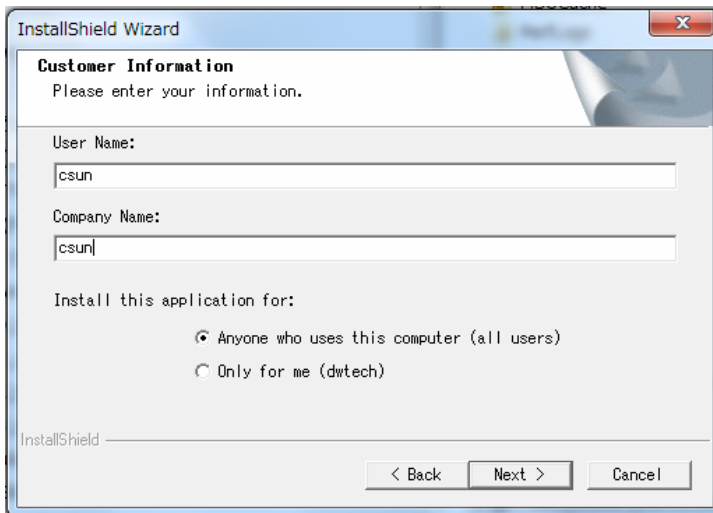
Flash Loader Demonstrator\_V1.3\_Setup.exeを実行する。



「Next」ボタンを押すと、英文のライセンスが出てきます。同意できる場合は、「Yes」ボタンを押す。

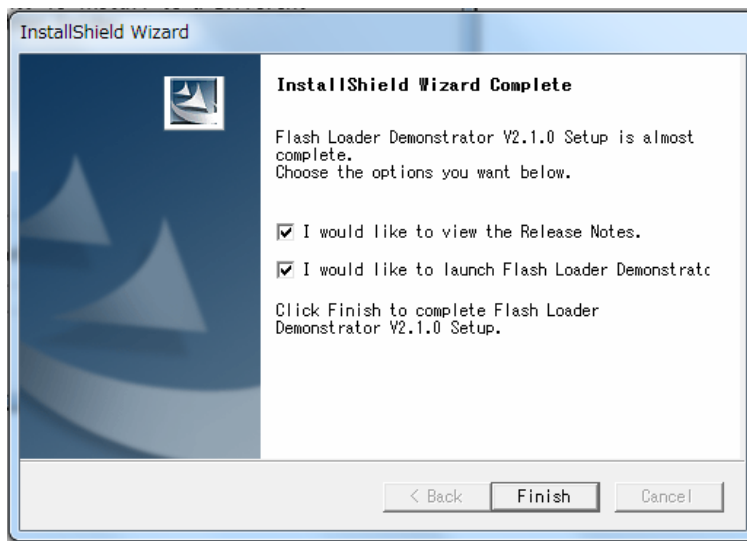


ユーザー名と会社名を入力して、「Next」ボタンを押す。





インストール先フォルダを変更せず、そのまま進んでください。



最後に「Finish」をクリックすると、ウィザードが閉じてインストールが終了。

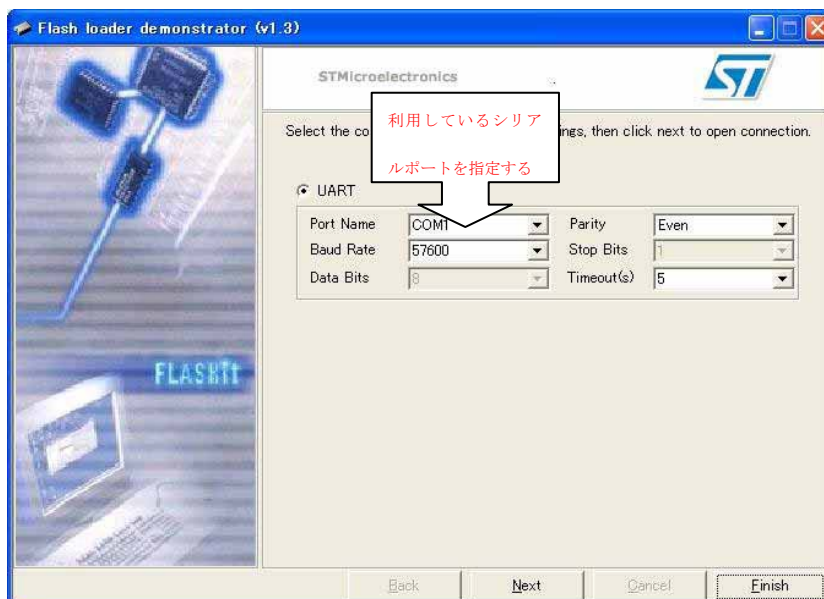
書き込む前にボードのJ10(Boot0)を1-2に設置する。

直接RS232 ケーブルでボードのCOM1 をパソコンと接続して、電源を入れる。

パソコン側に RS232 インタフェースがない場合は USB RS232 変換ケーブルで接続する。

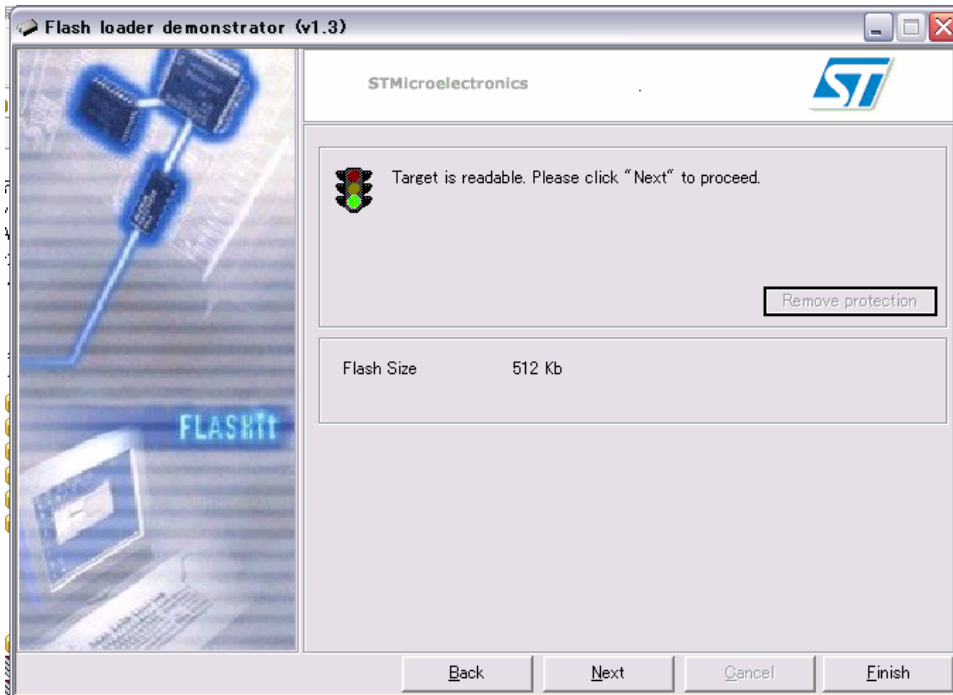
USB RS232変換ケーブル製品紹介URL : ( <http://www.csun.co.jp/SHOP/2010040601.html> )

上記準備終わったら、Windowsのメニュー「スタート」→「STMicroelectronics」→「Flash Loader Demonstrator」→「Flash Loader Demo」を選択して起動する。

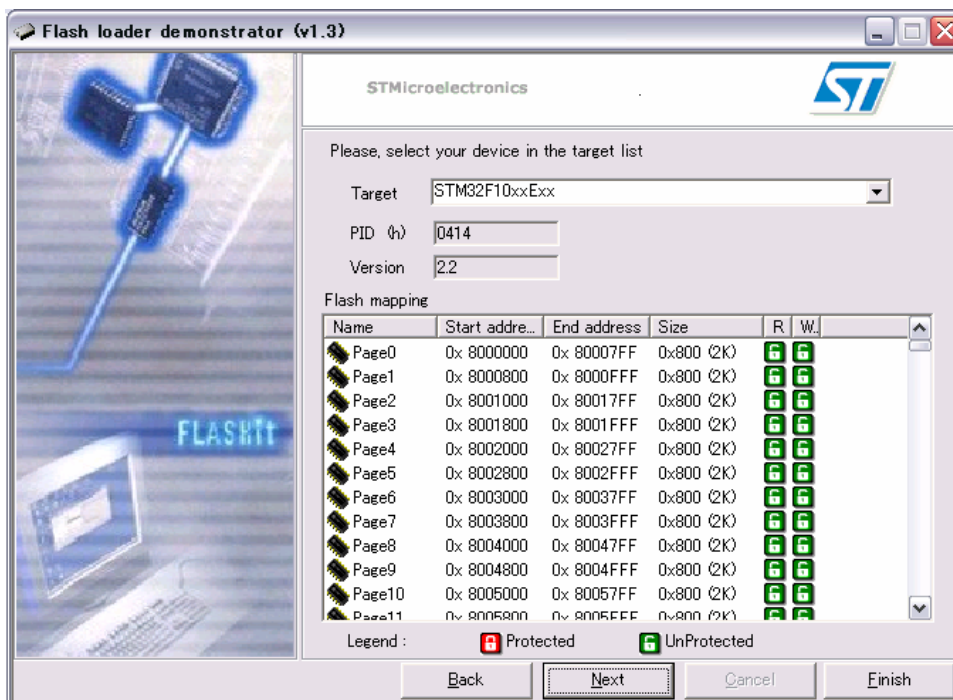


書き込み用のシリアルポートを選択して、「Next」ボタンを押す。

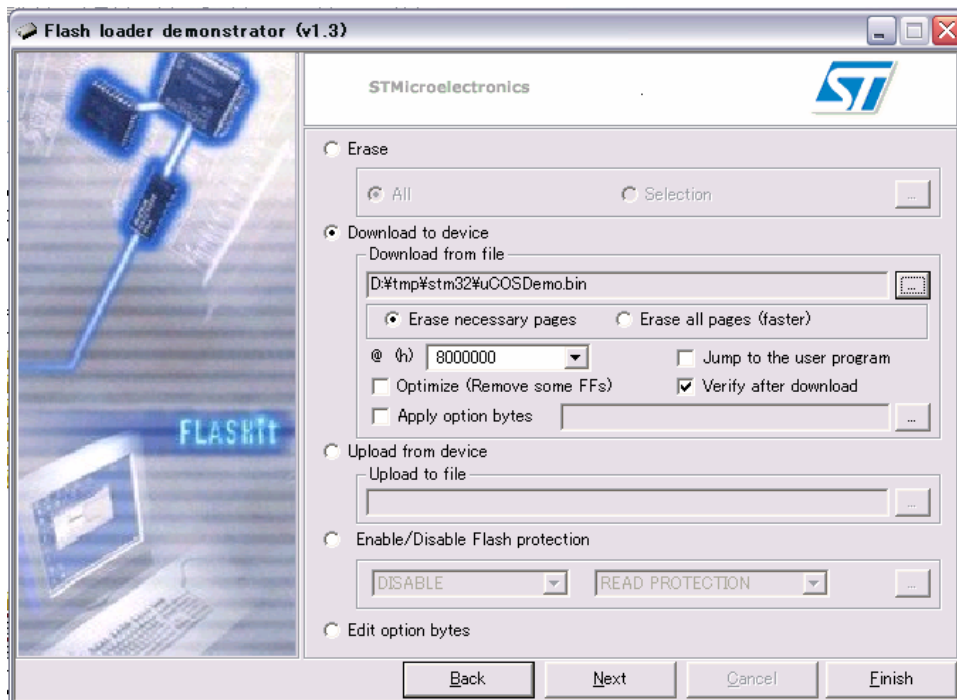
(USB-RS232ケーブルを利用している場合は、そちらの設定と合わせて設定する。)



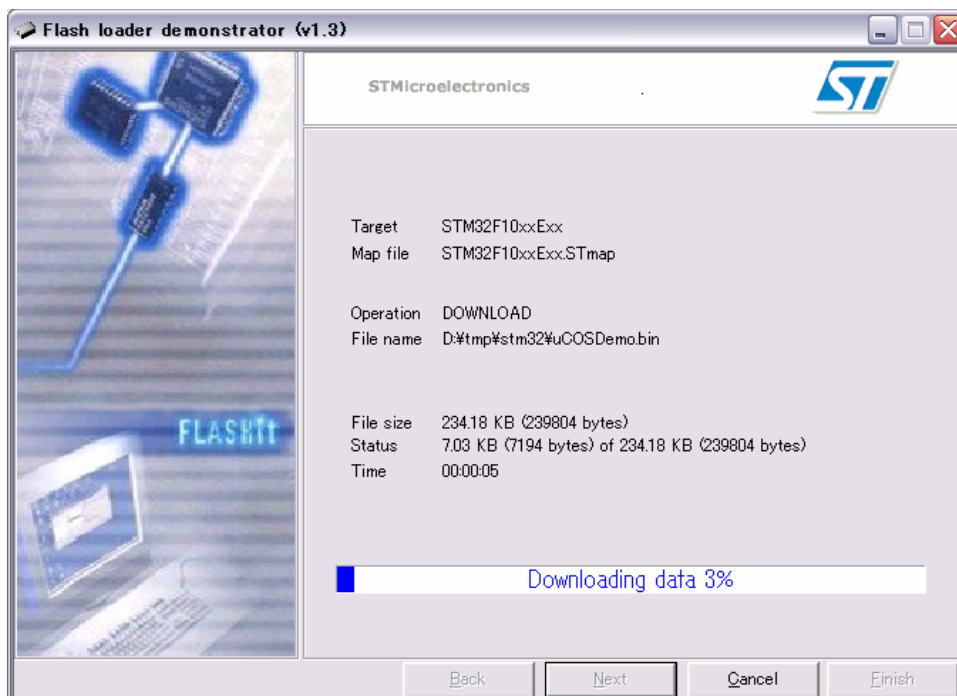
「Next」ボタンを押す。

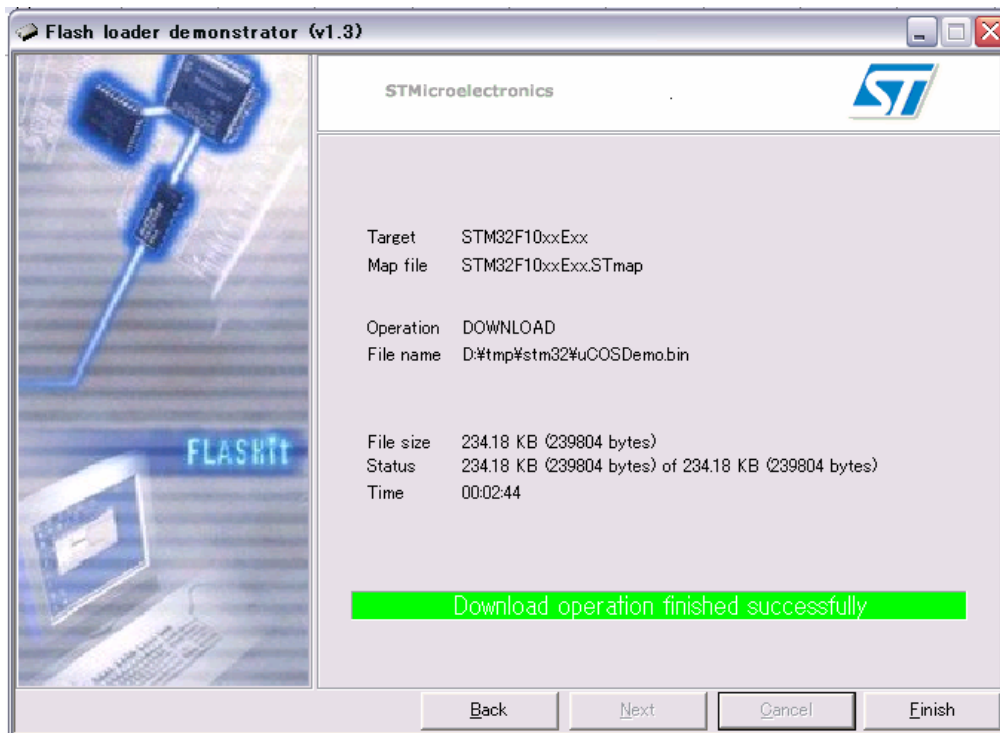


そのまま「Next」ボタンを押す。



書き込む\*. Binファイルを選択して、「Next」ボタンを押す。





最後に「Finish」をクリックすると、ウィザードが閉じて書き込みが終了。

## 5.2 OpenLinkで書き込む

弊社は OpenLink のハードウェアを提供しております（製品紹介 URL: <http://www.csun.co.jp/SHOP/2009121901.html>）。

### 5.2.1 ドライバのインストール

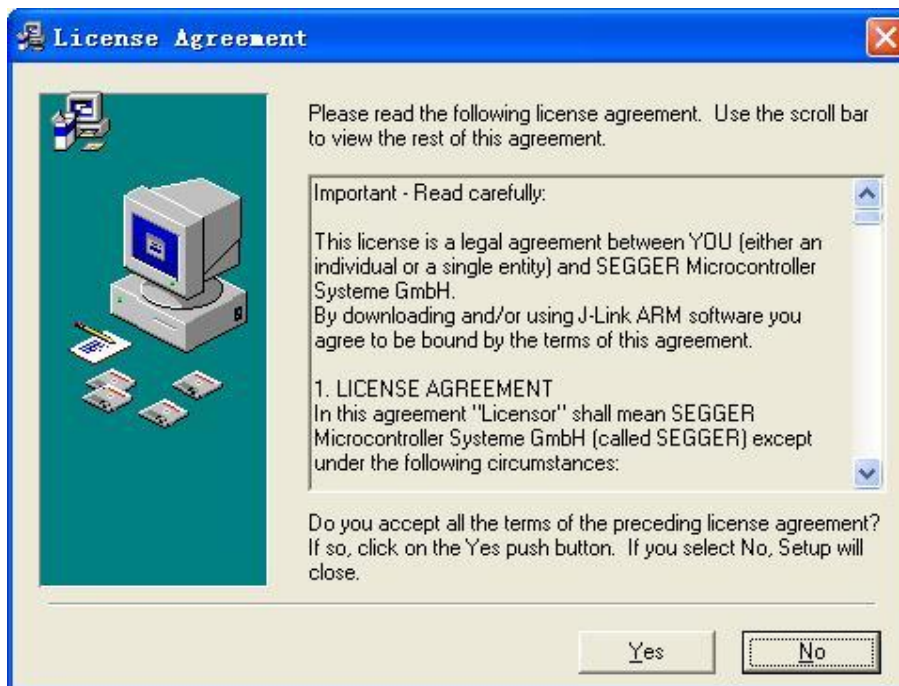
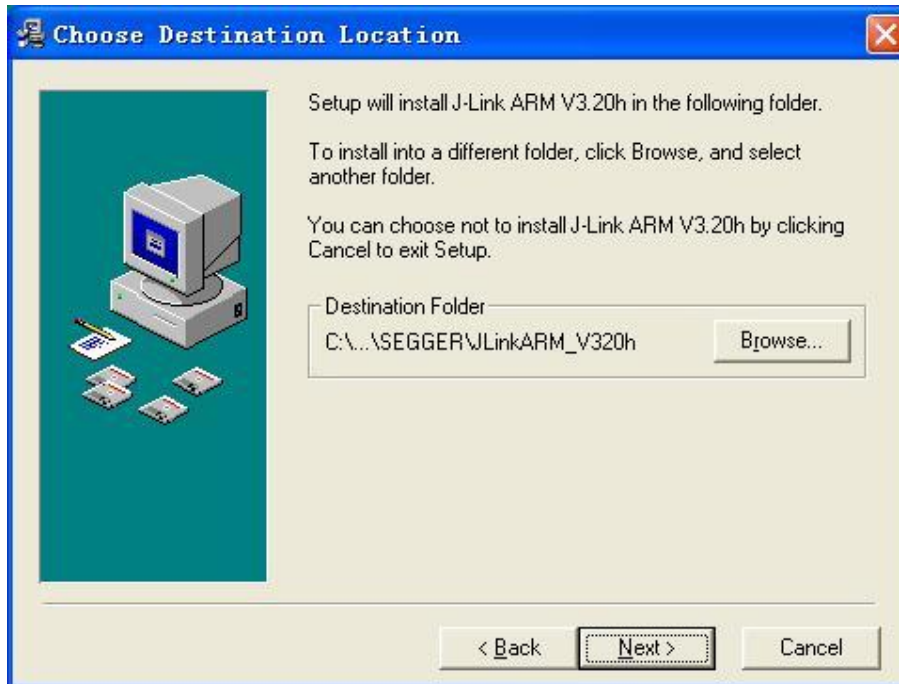
ドライバインストール用のファイルは弊社ホーム下記 URL からダウンロードできる。

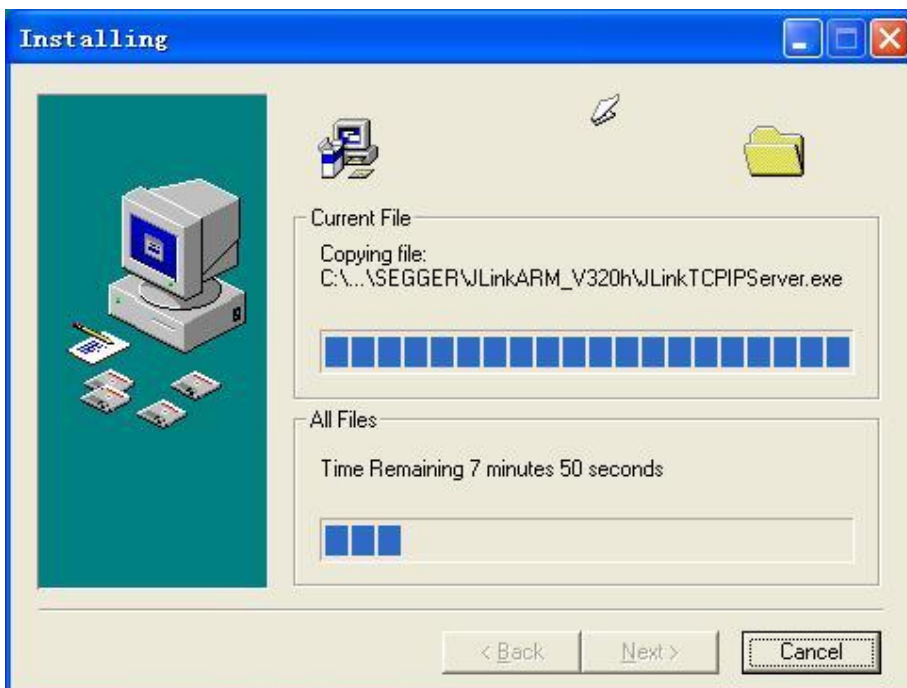
[http://www.dragonwake.com/download/open-link/Setup\\_OpenLinkARM.zip](http://www.dragonwake.com/download/open-link/Setup_OpenLinkARM.zip)

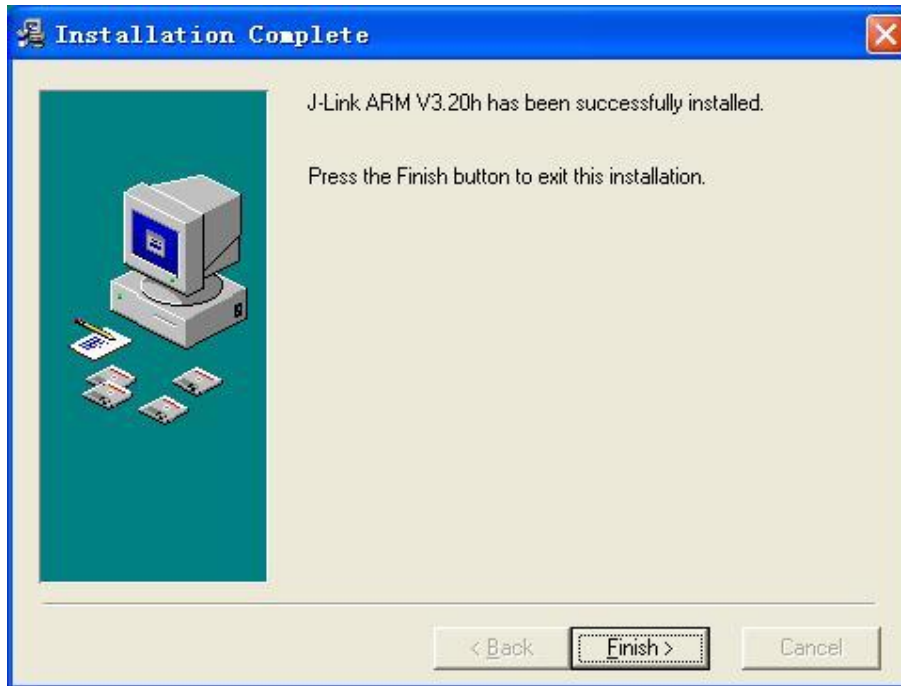
SEGGER 社様のソフトウェアを利用しておりますので、直接 SEGGER 社様ホームページから最新の USB ドライバをダウンロードできる。

<http://www.segger.com/cms/jlink-software.html>

インストールの際に、ダウンロードした ZIP ファイルを解凍し、デフォルトのままで行ってください。

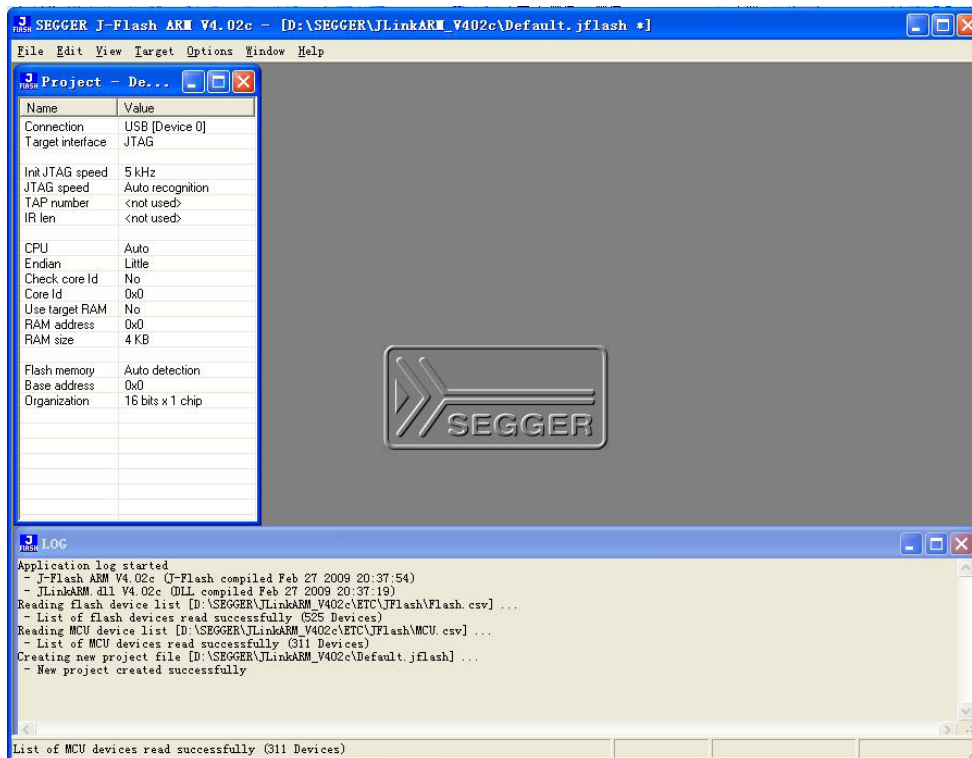




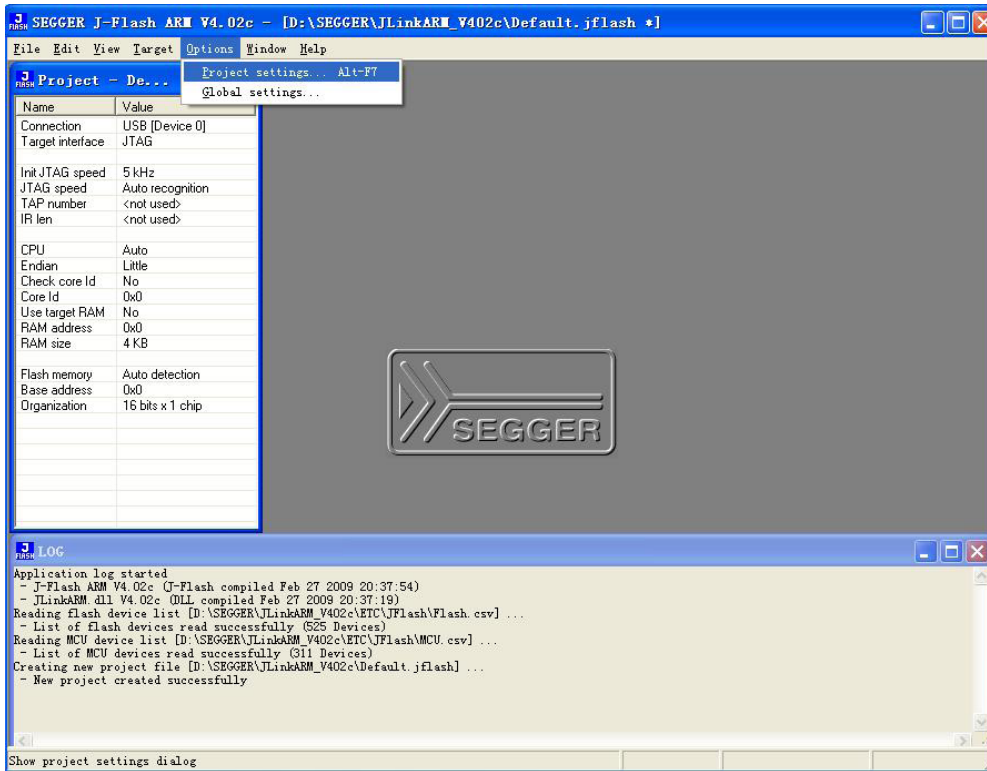


### 5.2.2 J-FLASH ARMで実行ファイルを書き込む

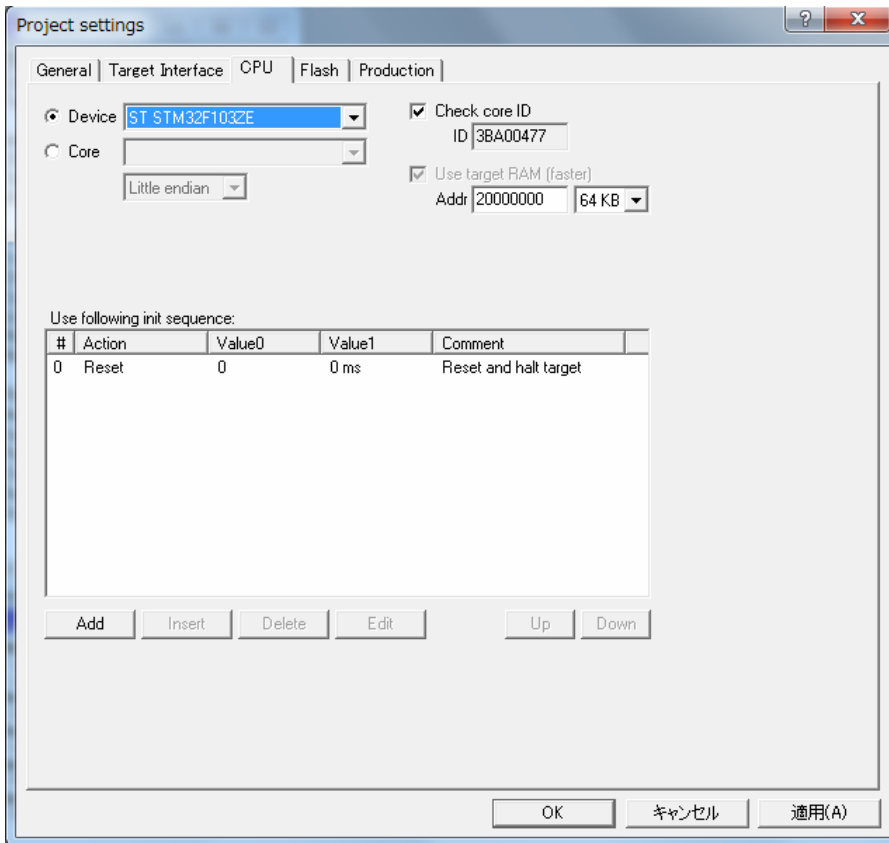
J-FLASH ARM を実行する。



書き込む前に必要な設定 (Options->project settings...) :

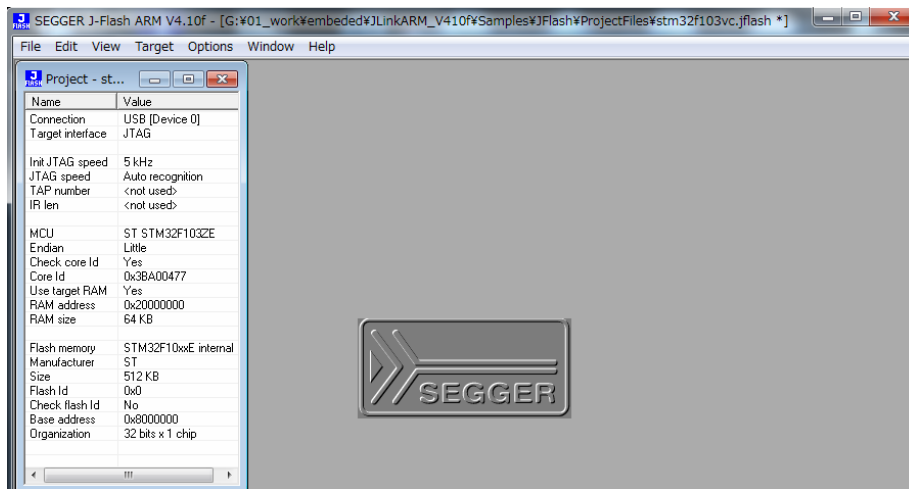


STM32 ボードの CPU 型番を選択する。

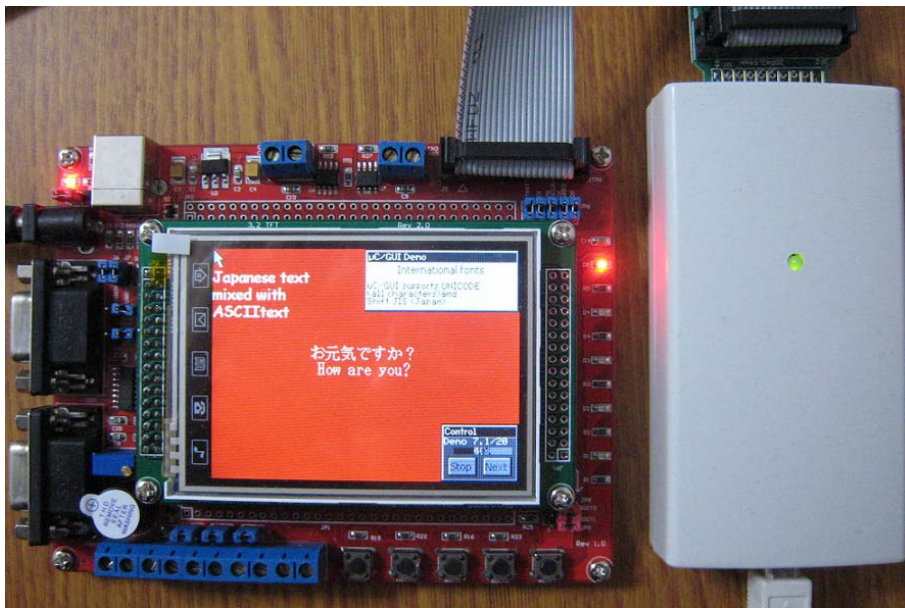




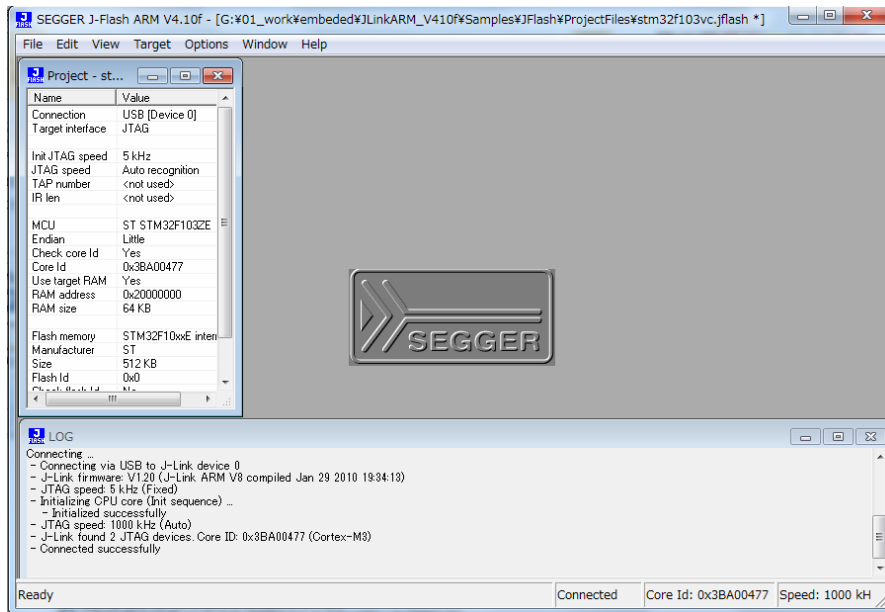
設定後、左側に書き込み情報が表示される。



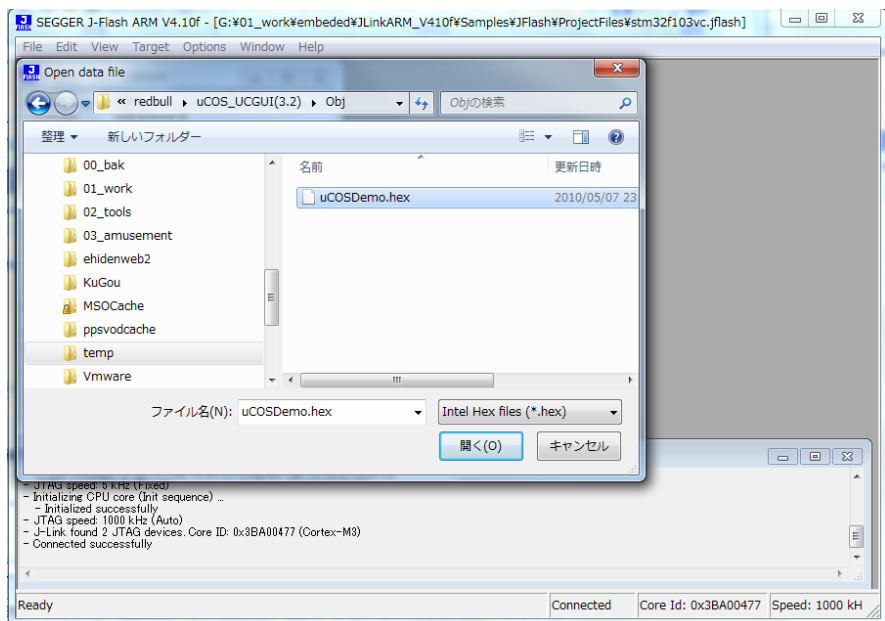
ボードを接続する。



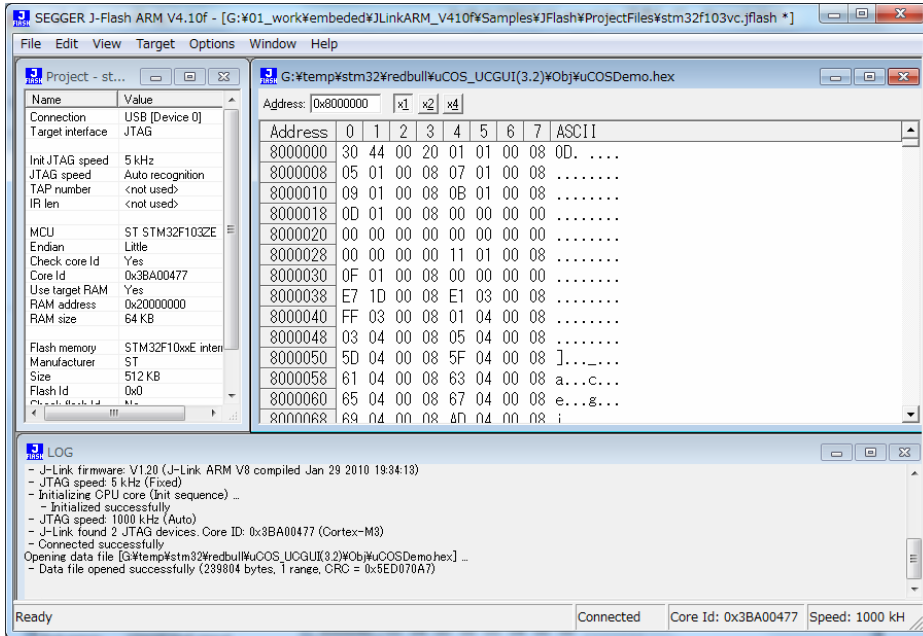
ソフト側も接続する (Target→Connect)。



File->open で実行ファイルを選択する。

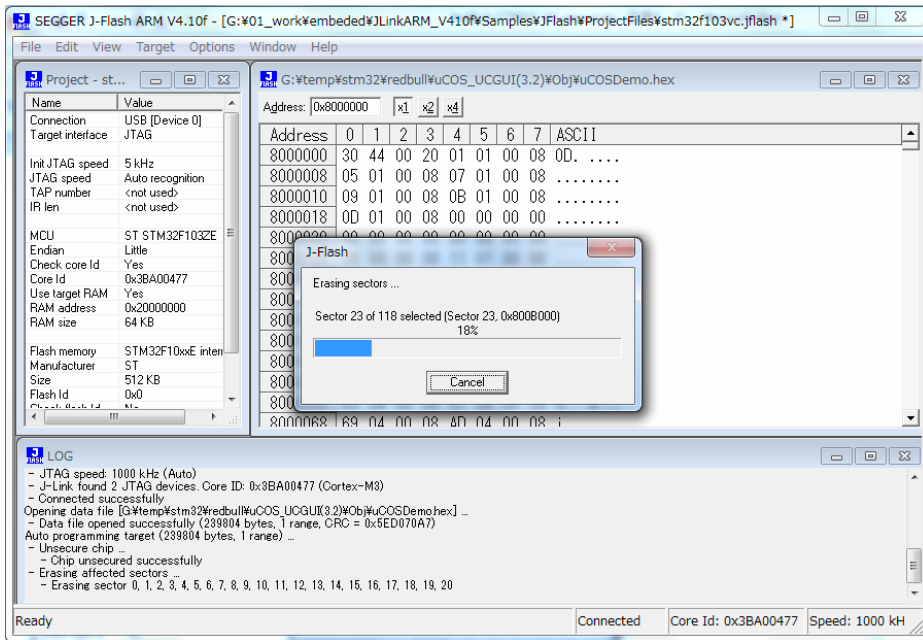


「開く (O)」をクリックする。

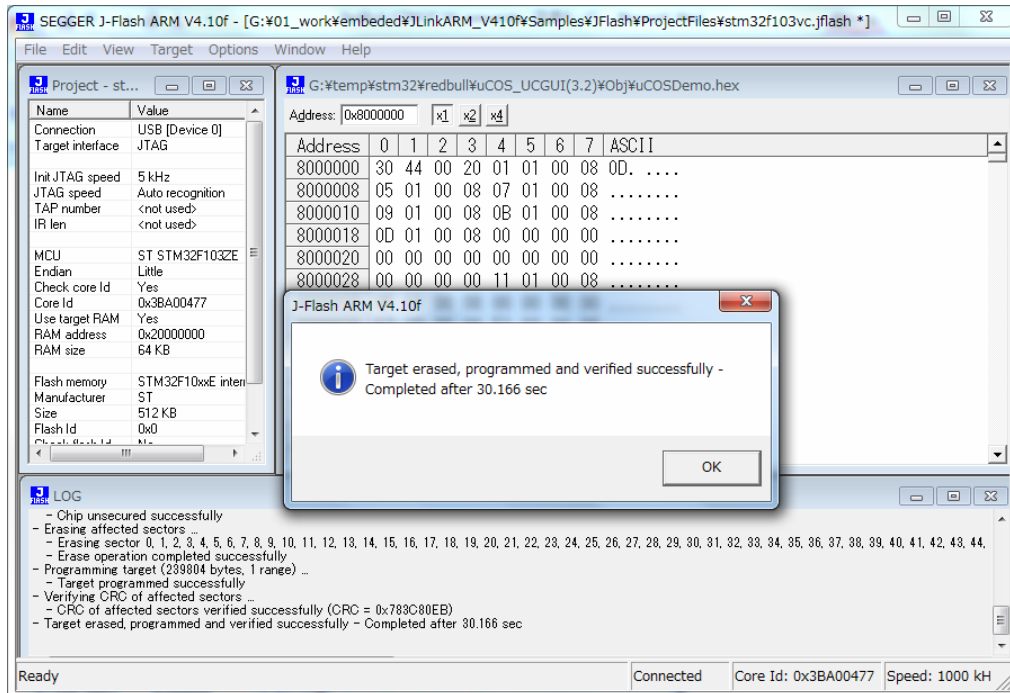


Target->Auto あるいは F7 で書き込み開始する。

書き込み中 :



書き込み完了 :



### 5.3 H-JTAGで実行ファイルを書き込む

H-JTAGはARMの為のJTAGエミュレータです。AXD又はkeilをサポートします。デバッグのスピードも速いです。詳しい情報はこちらです。

<http://www.hjtag.com>

弊社はH-JTAGのハードウェアを提供しております（製品紹介URL：<http://www.csun.co.jp/SHOP/200806151.html>）。パソコン側にはLTPが必要です。

(1) H-JTAG をダウンロードしてインストールする。

ホームページ<http://www.hjtag.com>から最新版をダウンロードできます。

H-JTAGの特性：

- RDI 1.5.0 & 1.5.1 をサポートします；
- ARM7 & ARM9（ARM9E-SとARM9EJ-Sを含む）；
- thumb & arm 命令；
- little-endian & big-endian；
- semihosting；
- 実行環境WINDOWS 9.X/NT/2000/XP；
- flashの書き込み

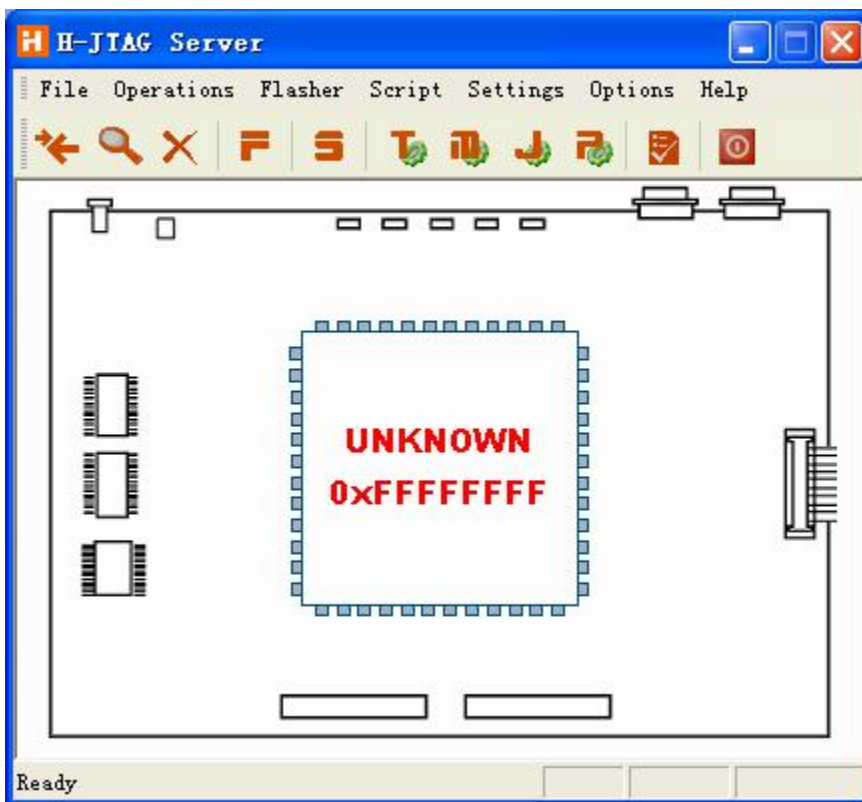
デフォルト設定のままインストール完了させて、デスクトップでH-JTAG と H-Flasher が生成される。

H-JTAG を実行する前に、まず、H-JTAGでSTM32ボードとパソコンを接続する。STM32 ボードに電源を入れてください。

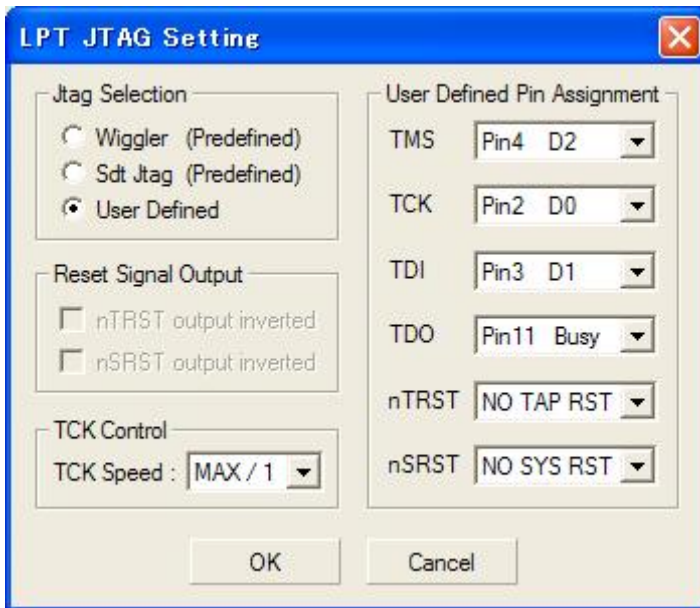
初めてH-JTAG を実行する時、次の画面のエラーメッセージが出て来る。



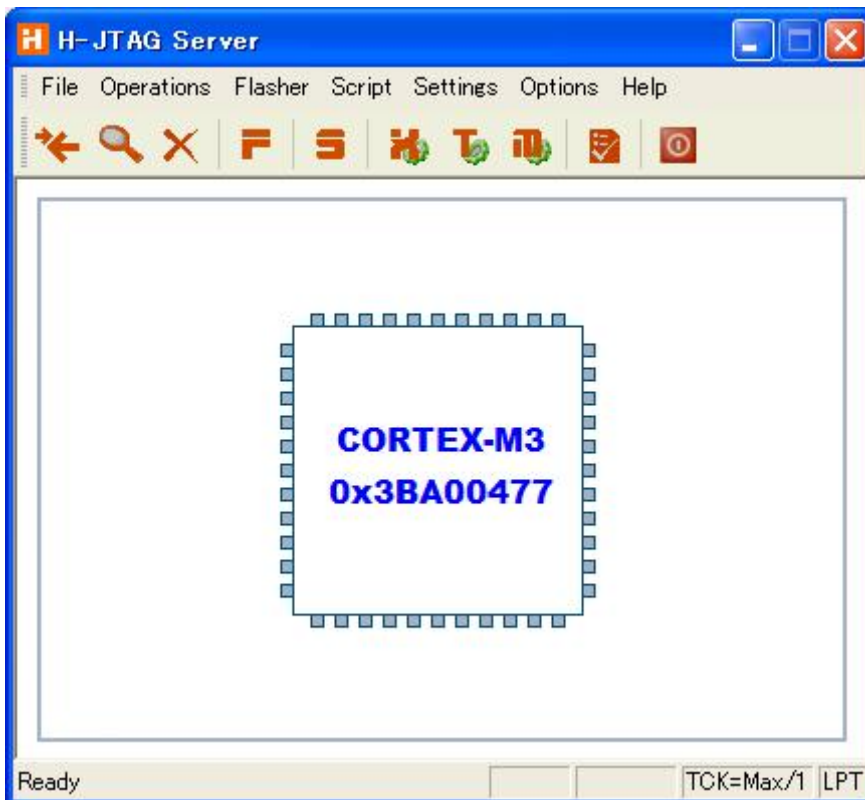
“確定”ボタンをクリックすると、初の画面が出て来る。



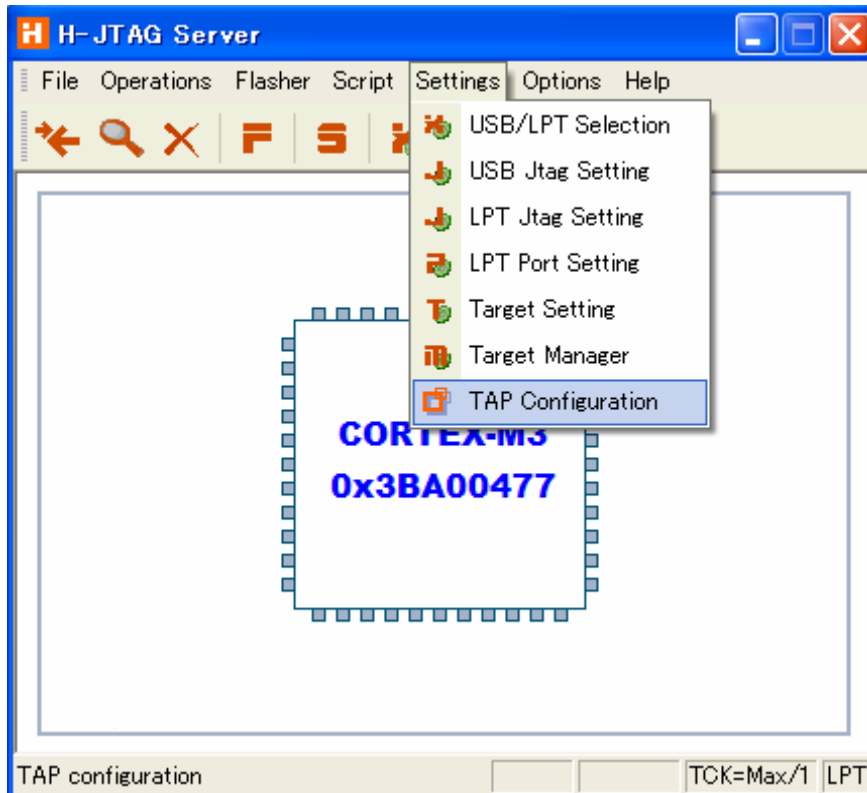
H-JTAG のメニュー : Setting → LPT Jtag Setting



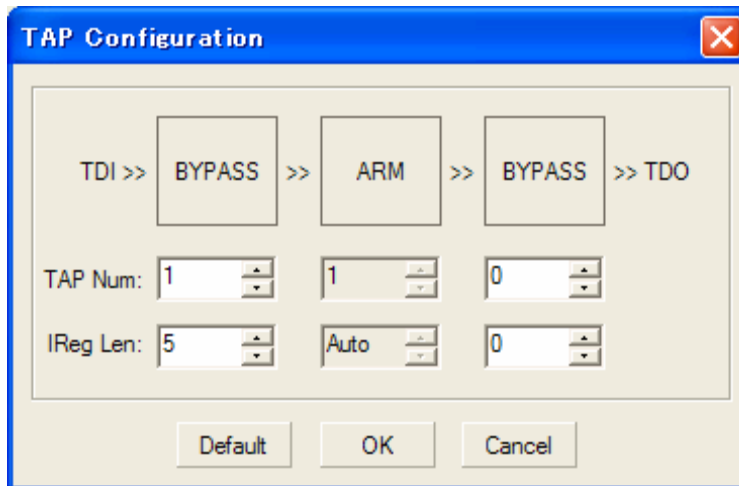
上記画面の様に設定して、“Ok”ボタンをクリックすると CORTEX-M3 が認識される。



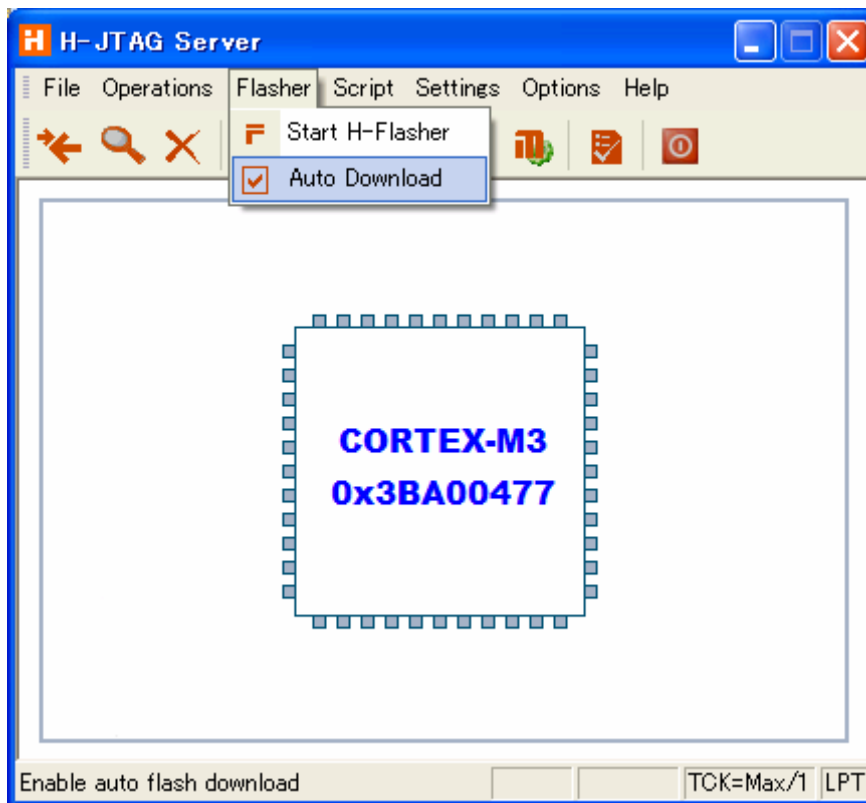
メニュー「Settings」→「TAP Configuration」を選択する。



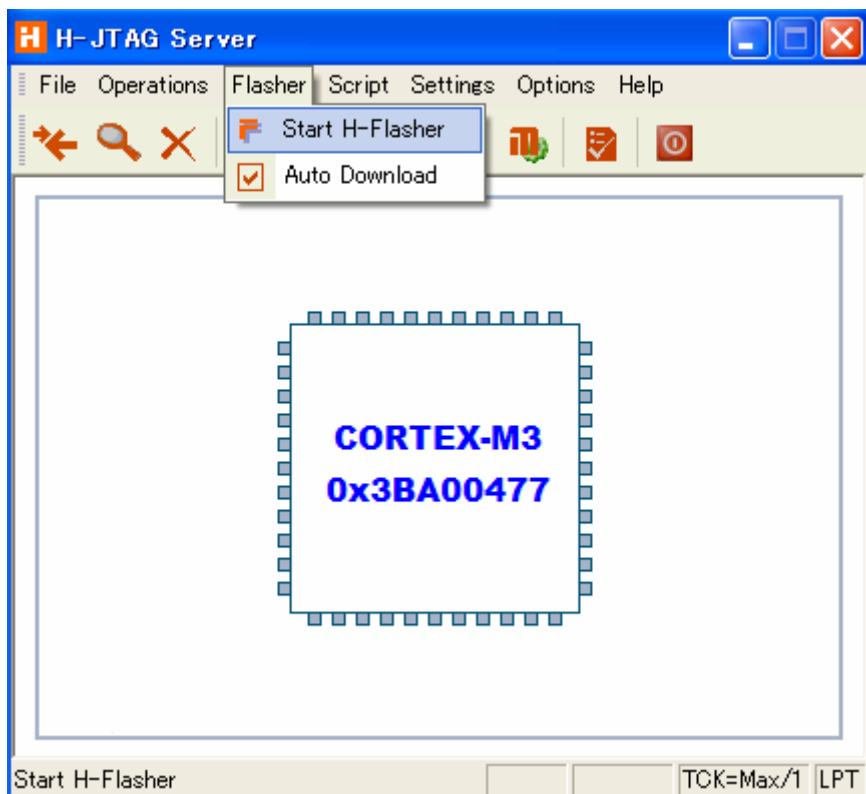
下記画面の通りに設定する。



メニュー「Flasher」→「Auto Download」にチェックを入れる。

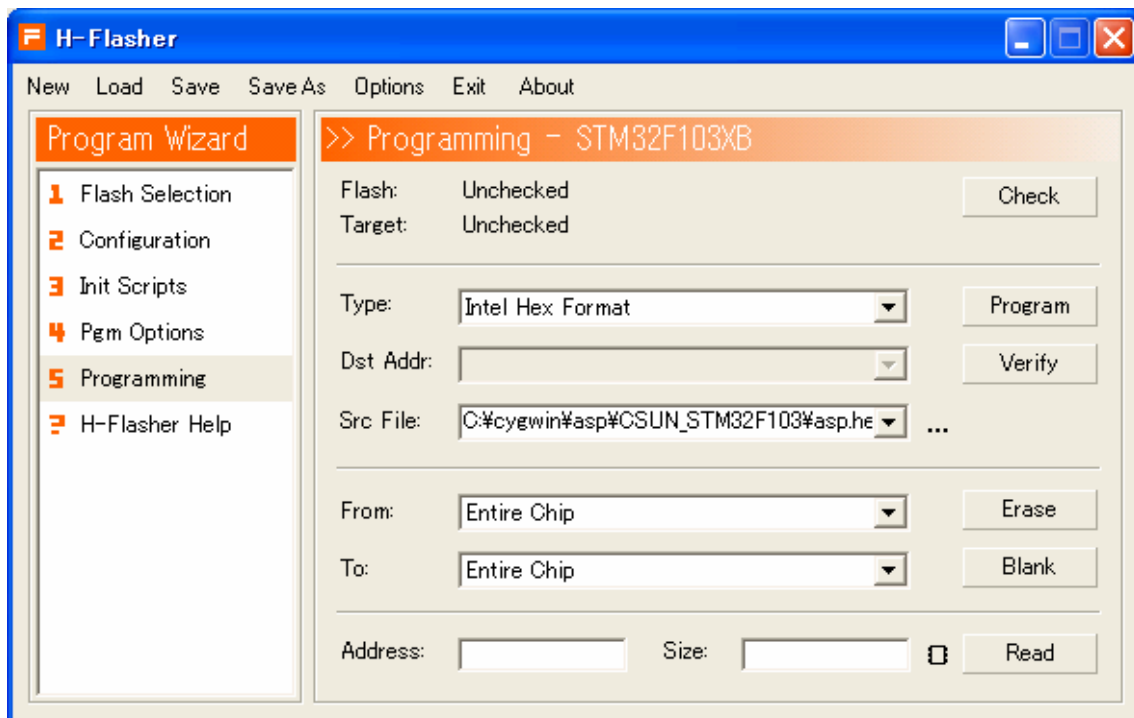


メニュー「Flasher」→「Start H-Flasher」を選択する。



STM32F103ZE を選択する。





ファイルのフォーマットを「Intel Hex Format」を設定して、実行ファイル\*.hex を選択して、「Program」ボタンをクリックする。

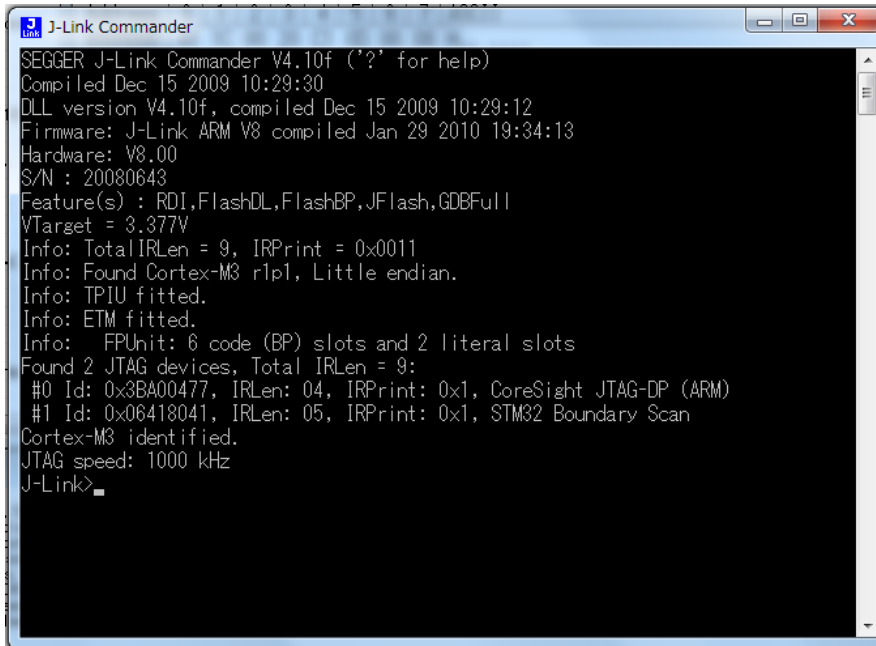
## 第六章 OpenLinkでデバッグ

OpenLink エミュレータ使い環境のインストール手順は「5.2.1 ドライバのインストール」をご参照ください。

### 6.1 J-Link commandでデバッグ

コマンドラインでコマンドを入力して実行する。

J-Link command を起動すると、JLINK のバージョン情報が表示される。ターゲットを接続している場合は、ターゲットの状態と CPU などの情報が表示される。



```
SEGGER J-Link Commander V4.10f ('?' for help)
Compiled Dec 15 2009 10:29:30
DLL version V4.10f, compiled Dec 15 2009 10:29:12
Firmware: J-Link ARM V8 compiled Jan 29 2010 19:34:13
Hardware: V8.00
S/N : 20080643
Feature(s) : RDI,FlashDL,FlashBP,JFlash,GDBFull
VTarget = 3.377V
Info: TotalIRLen = 9, IRPrint = 0x0011
Info: Found Cortex-M3 r1p1, Little endian.
Info: TPIU fitted.
Info: ETM fitted.
Info: FPUunit: 6 code (BP) slots and 2 literal slots
Found 2 JTAG devices, Total IRLen = 9:
#0 Id: 0x3BA00477, IRLen: 04, IRPrint: 0x1, CoreSight JTAG-DP (ARM)
#1 Id: 0x06418041, IRLen: 05, IRPrint: 0x1, STM32 Boundary Scan
Cortex-M3 identified.
JTAG speed: 1000 kHz
J-Link>
```

J-Link command では豊富なデバッグ、検索などのコマンドを持っている。詳しい内容は J-Link command で?を入力してエンタリすると説明が表示される。

## 第七章 開発ツールKEILの応用

MDK315B.exe は開発ツール KEIL の無償評価版です。

Keil社のHP (<http://www.keil.com/>) から最新版がダウンロード出来ます。

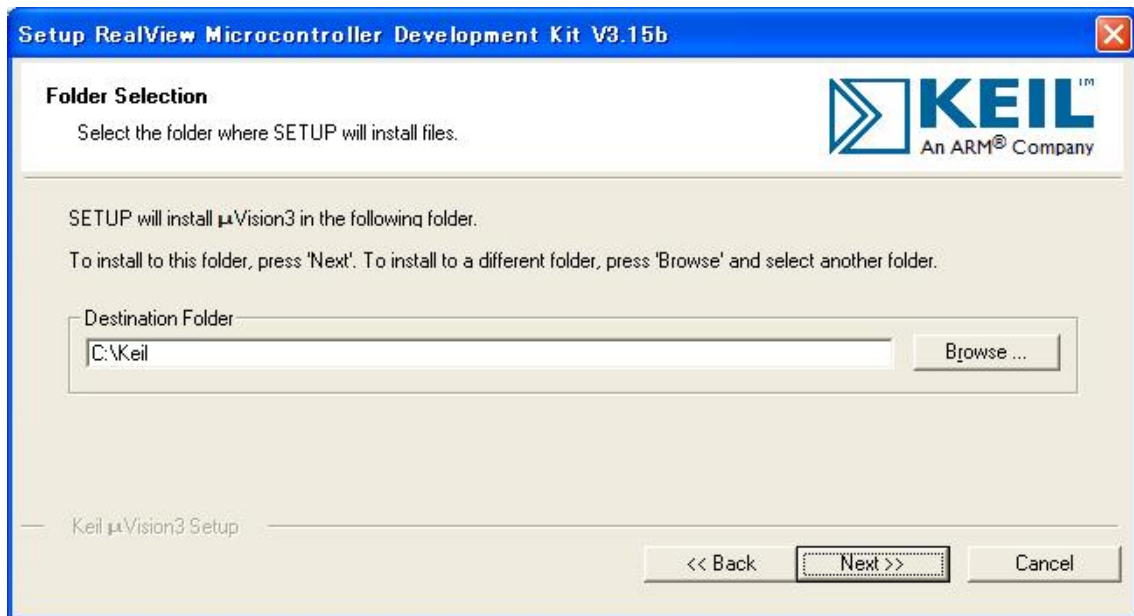
### 7.1 KEILのインストール

MDK315B.exe を実行して、KEIL3.15 をインストールする。

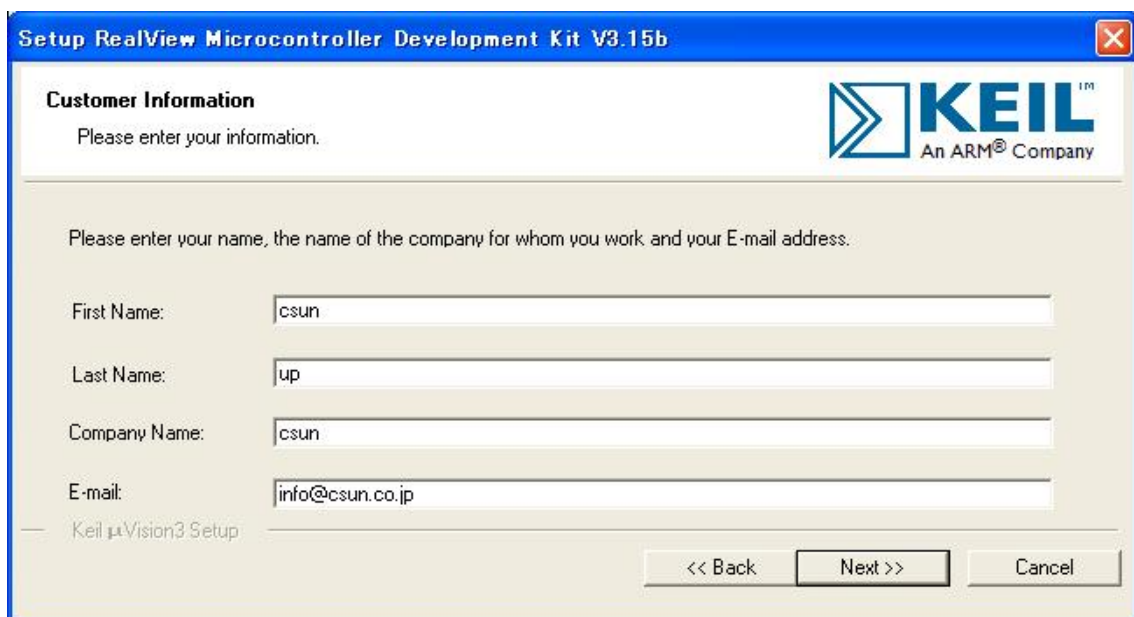


「Next」ボタンを押すと、英文のライセンス契約画面が表示される。同意できる場合は、「I accept the terms of the license agreement」を選択して、「Next」ボタンを押す。

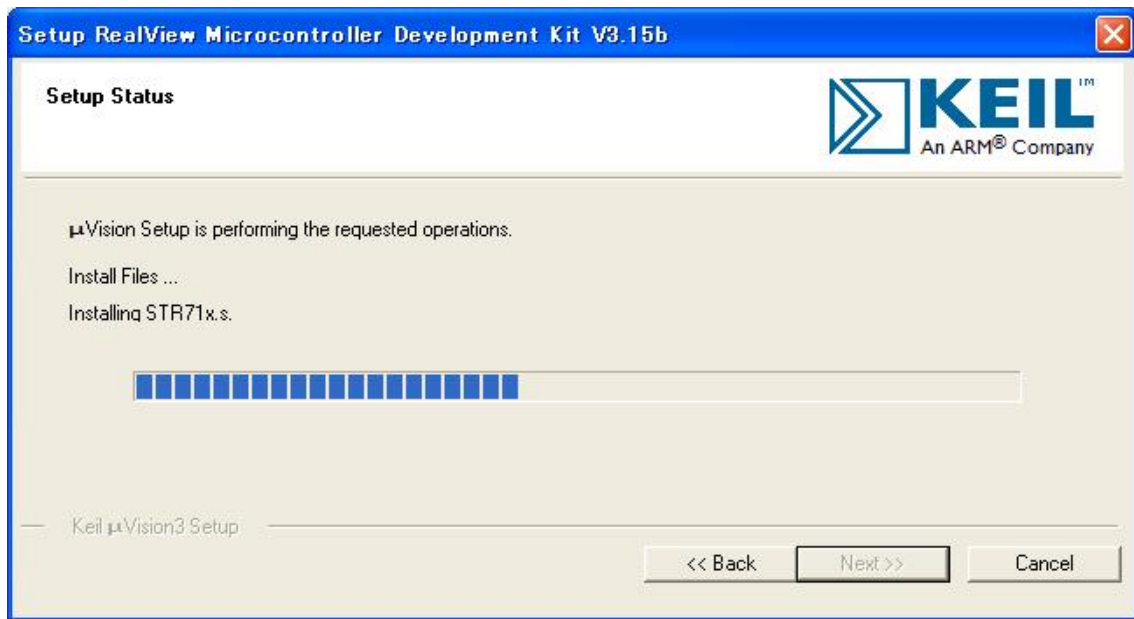




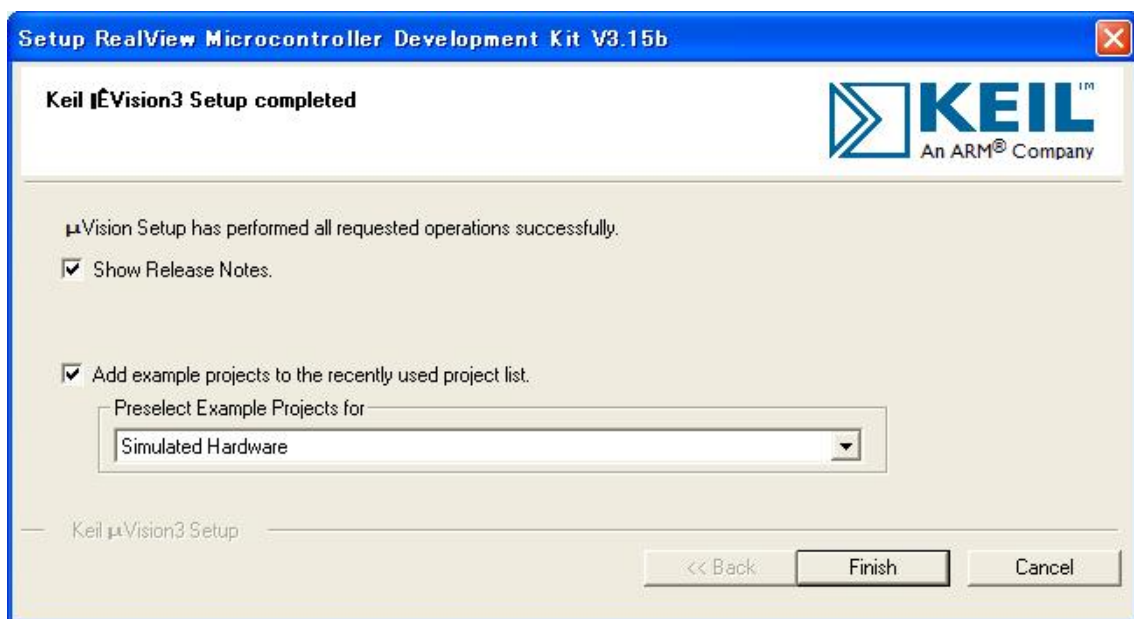
インストール先フォルダを変更せず、そのまま進んでください。



使用者の名前と所属会社名を入力するダイアログが表示される。名前は半角のアルファベットで入力してください。



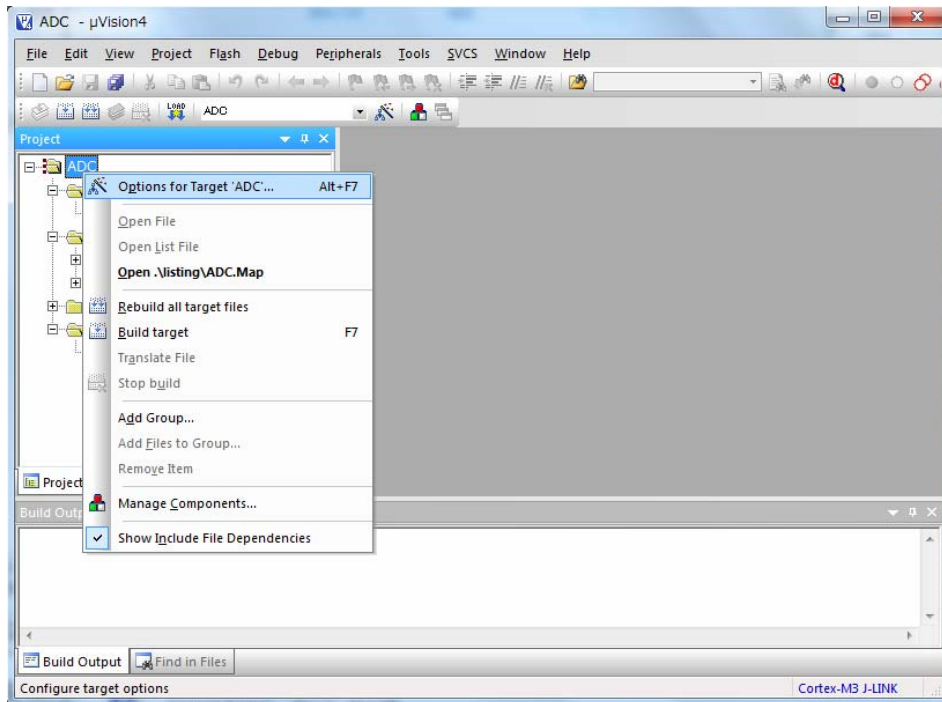
インストール中の画面です。



最後に「Finish」をクリックすると、ウィザードが閉じられてインストール終了。  
デモ版ではライセンスがないので、プログラムのサイズ制限があります。ライセンスを取得するにはKeil社の日本代理店と連絡する事。

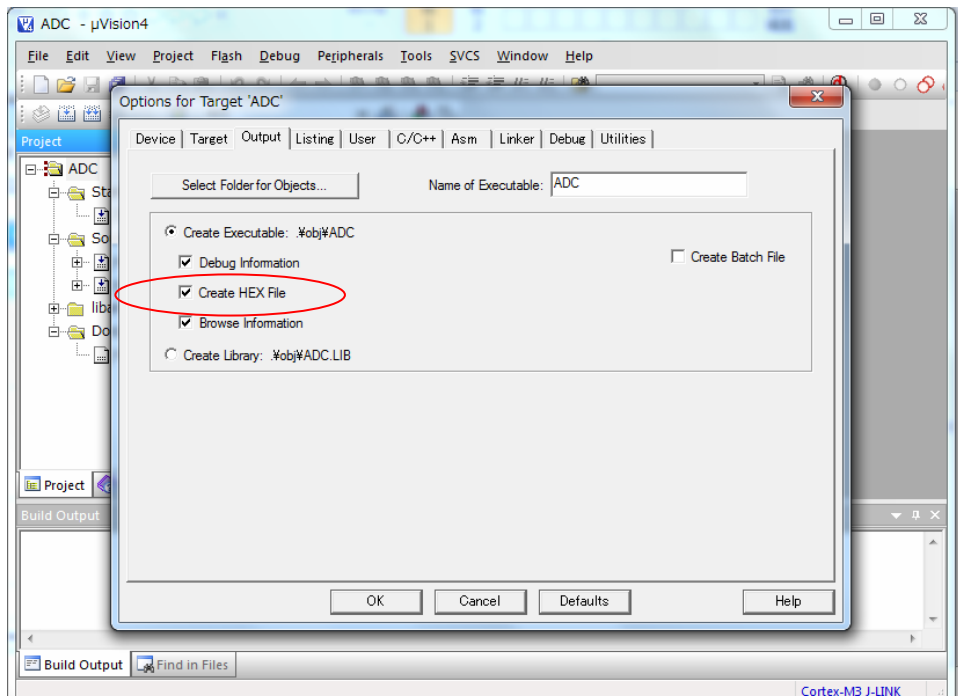
## 7.2 既存のプロジェクトから

プロジェクトファイルExample/ADC\_test/ADC.Uv2をダブルクリックする。或いはKEILのメニューでProject→Open Project…でADC.Uv2を選択する。

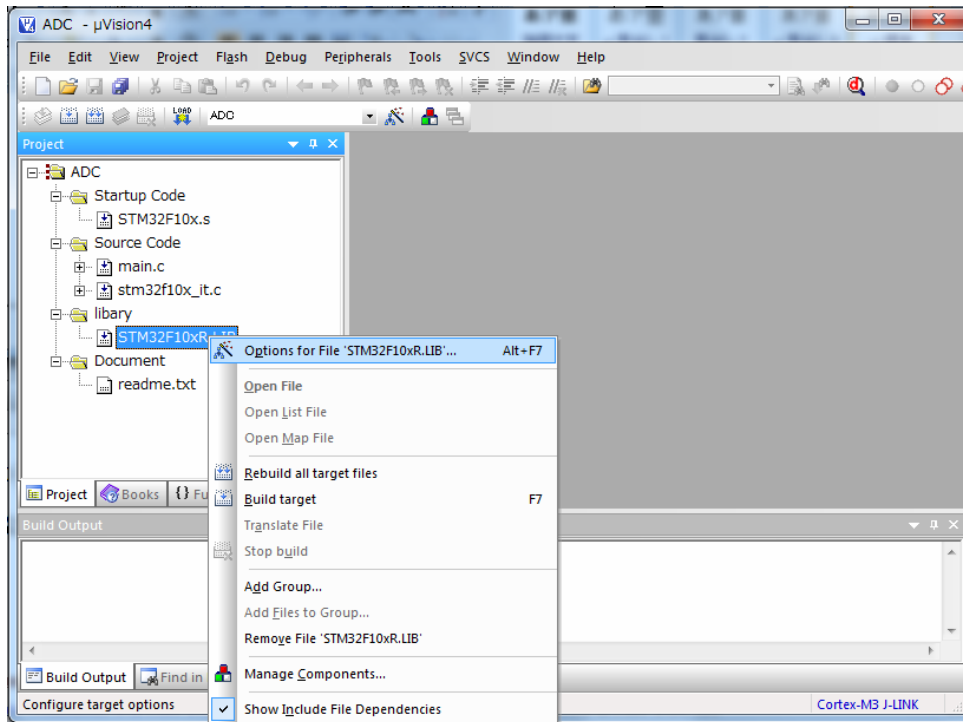


プロジェクト ADC を右クリックして「Options for Target ‘Target 1’ …」をクリックする。

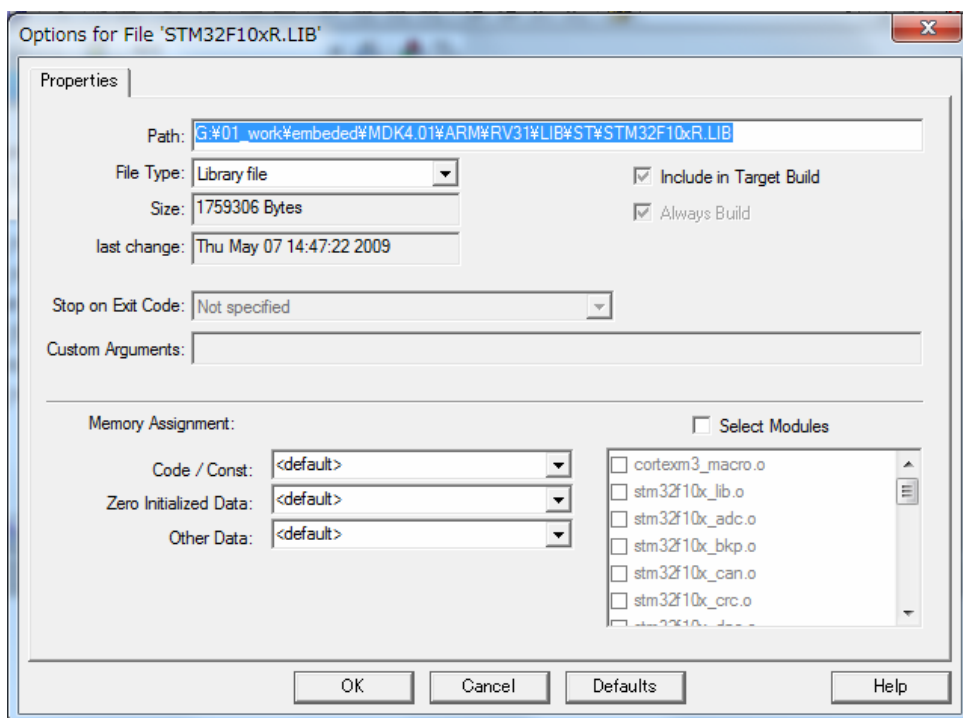
Options for Target ‘Target 1’ の画面が出て来る。「output」タブを選択する。

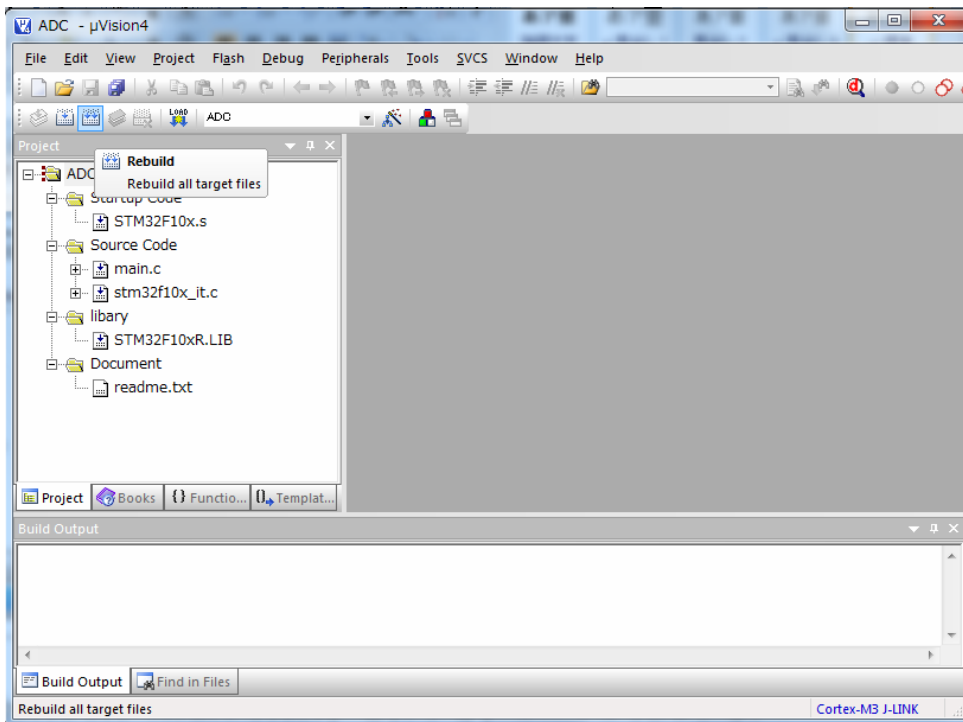


Create HEX Fileの所のをチェックを入れて「OK」ボタンをクリックする。  
STM32F10xR.LIBが見つけない場合は、右クリックして「Options for File ‘STM32F10xR.LIB’ …」をクリックする。

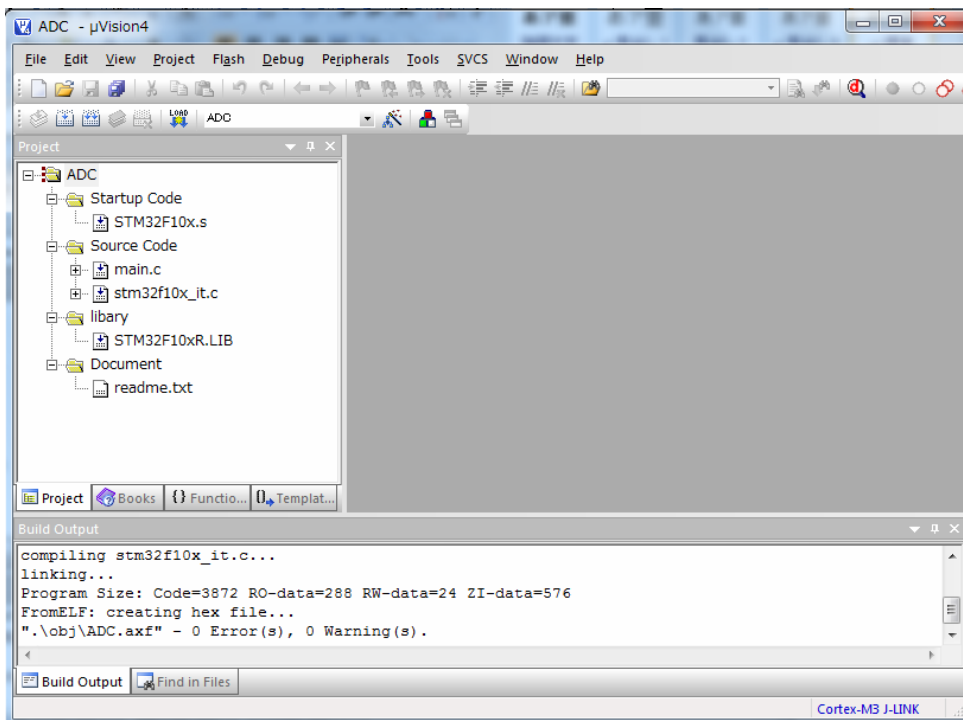


Pathの所に“Keilのインストールフォルダ¥ARM¥RV31¥LIB¥ST”を入力してOKを押す。





ツールバーの「Rebuild all target files」を押すと、ビルドが始まる。



ビルドが成功したら、プロジェクトのoutputフォルダにADC.hex ファイルを生成される。  
このHEXファイルをSTM32F103 ボードに書き込む。

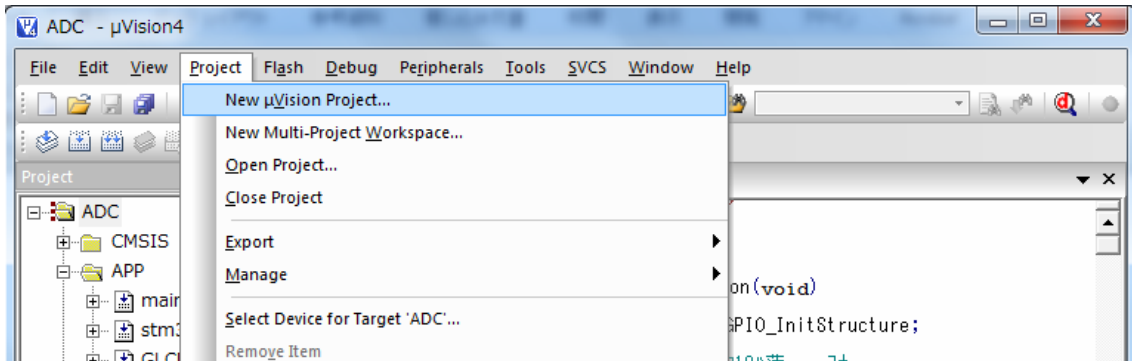
実行する前にPC側のハイパーターミナル（115200(B)、8(D)、なし(P)、1(S)、なし(F))を  
起動する。

※VR1のボリュームを調整するとハイパーターミナルの画面で数値が変化する。

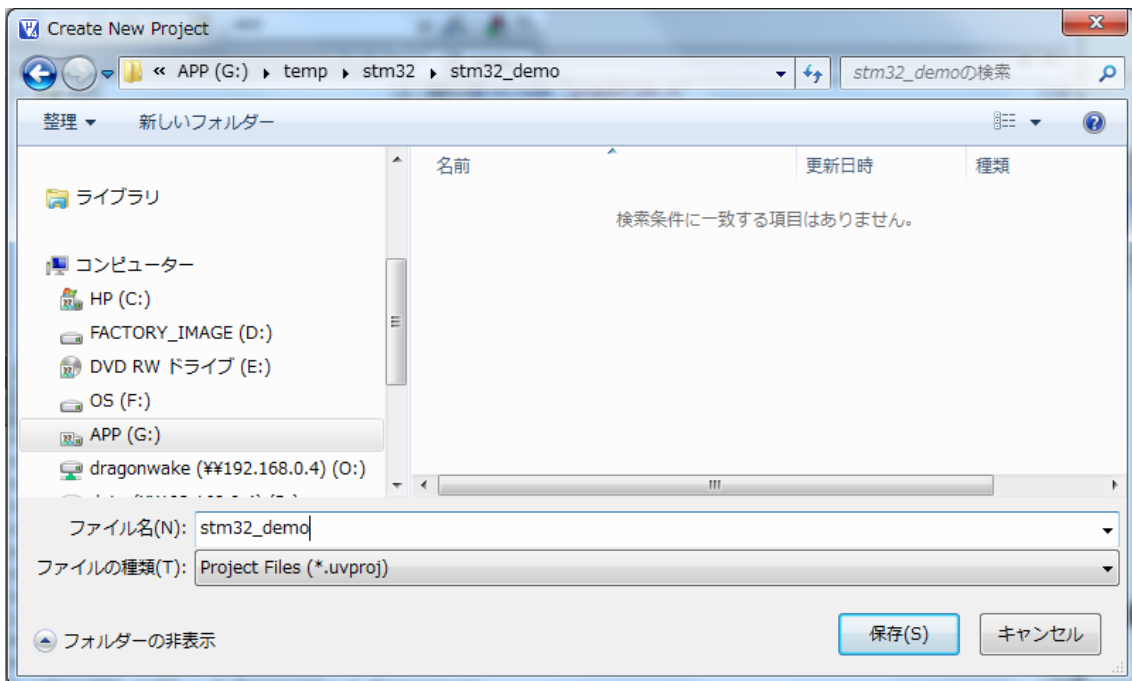


### 7.3 新しいプロジェクトの作成

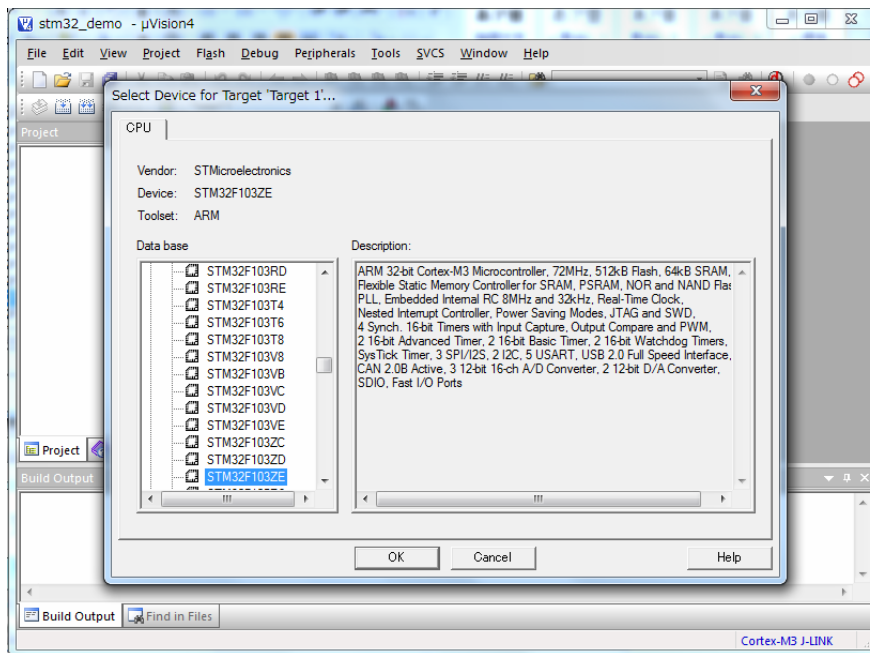
Keil のメニュー「Project」→「New uVision Project…」を選択する。



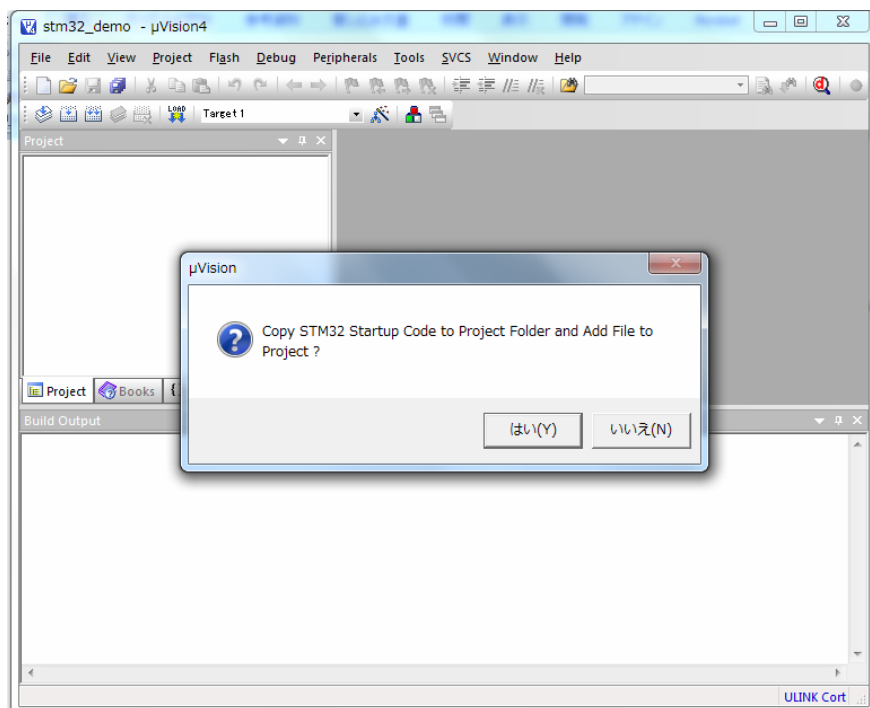
プロジェクト名前を入力して、保存する。



CPU 選択画面が出て来る。選択肢 STMicroelectronics を開いて STM32F103ZE を選択する。

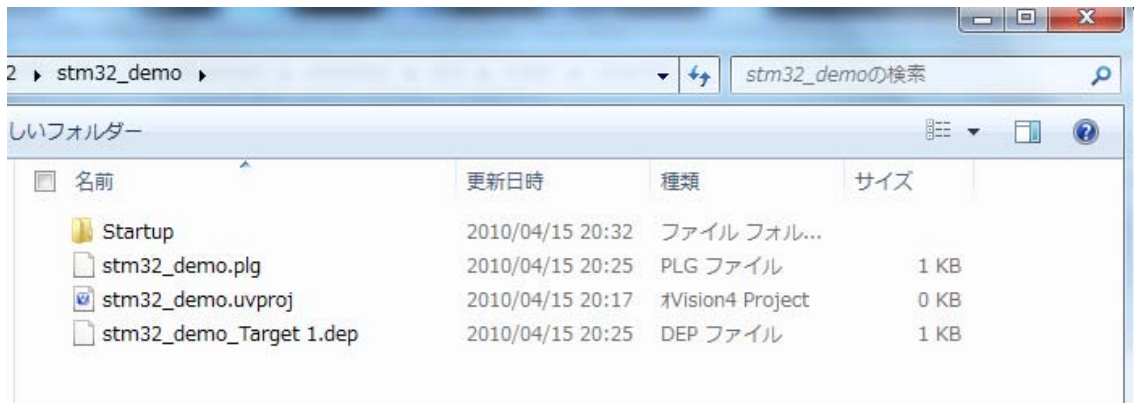


「OK」ボタンをクリックすると下記画面が表示される。

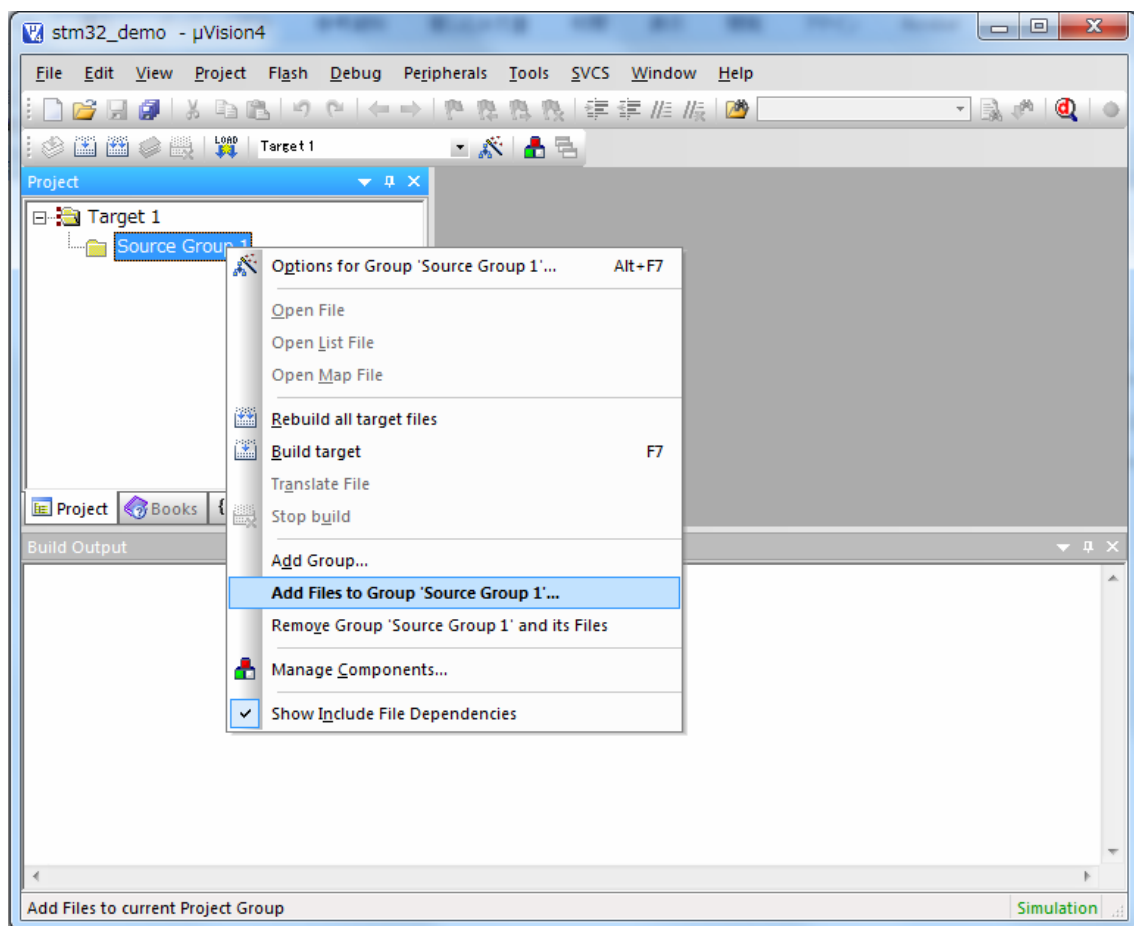


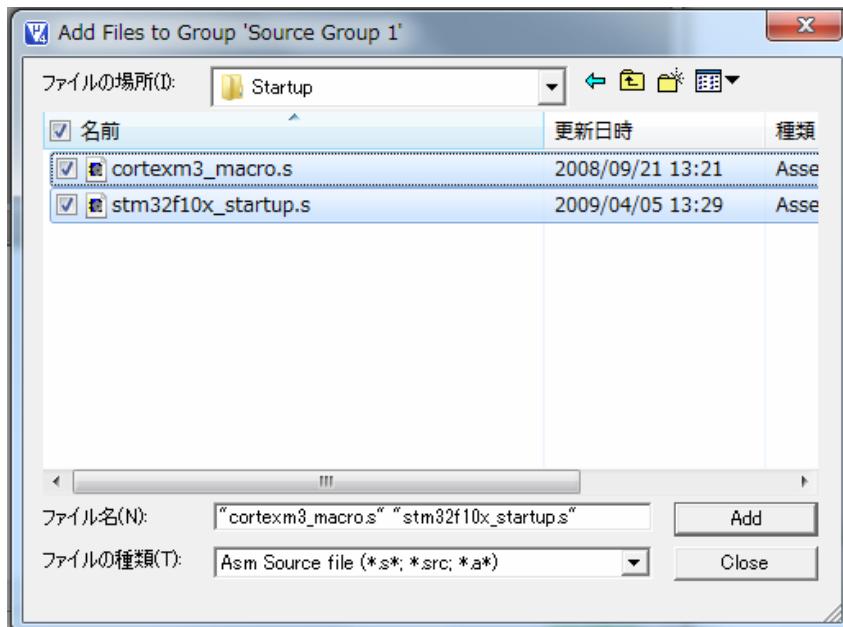
「いいえ」ボタンを押してください。

弊社 HP で提供している tools.rar にある Startup フォルダをプロジェクトにコピーする。

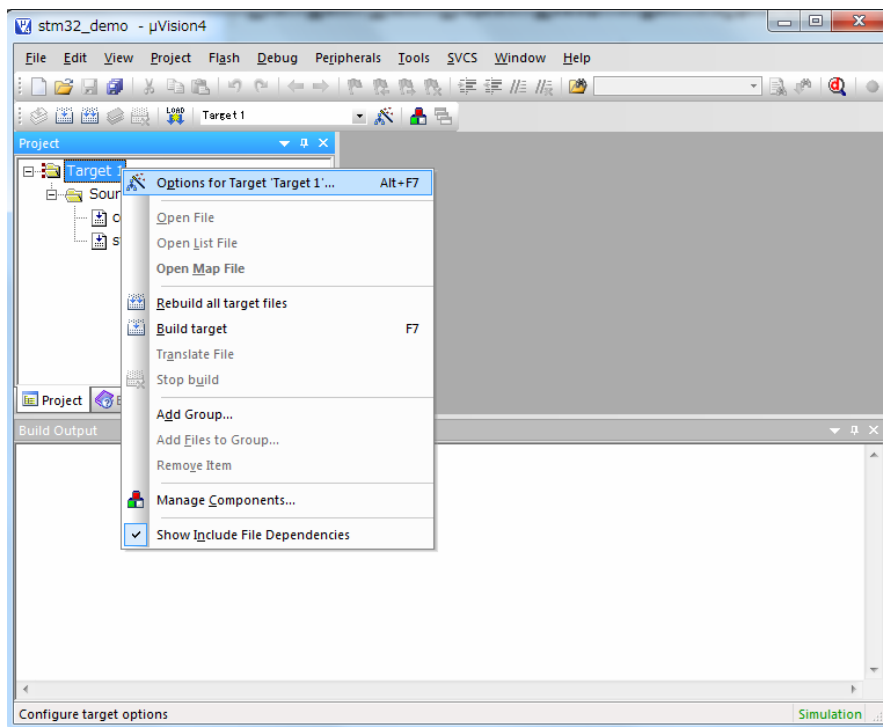


プロジェクトの「Source Group 1」でマウスを右クリックしてメニューから「Add Files To Group 'Source Group 1' ...」をクリックしてファイルを添加する。

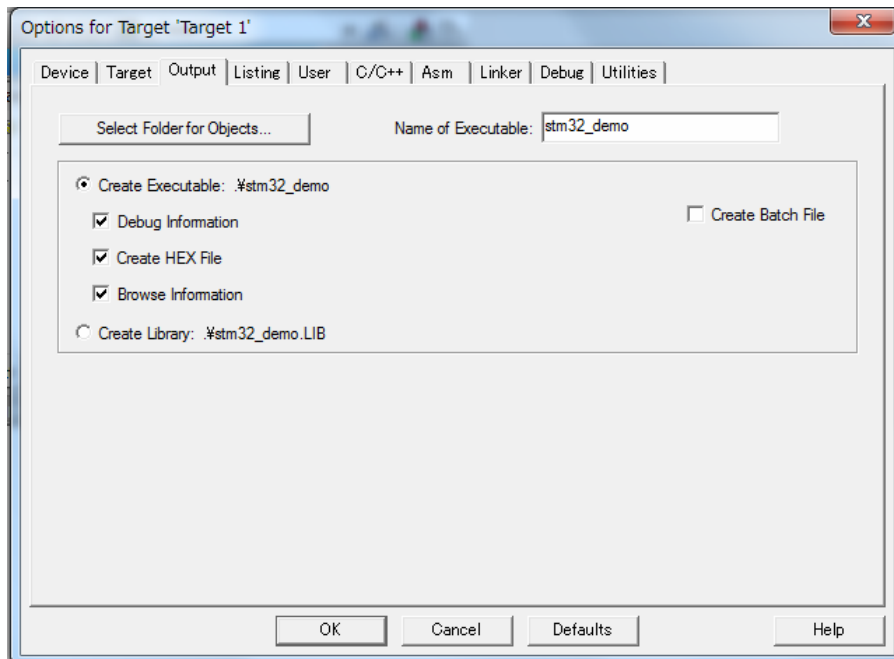




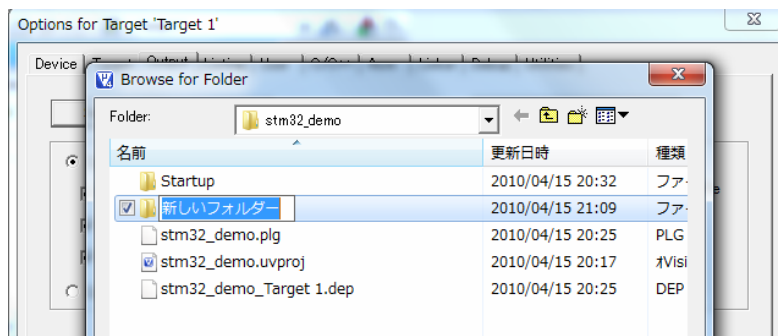
スタートアップファイルを添加される。  
プロジェクトのオプションを設定する。



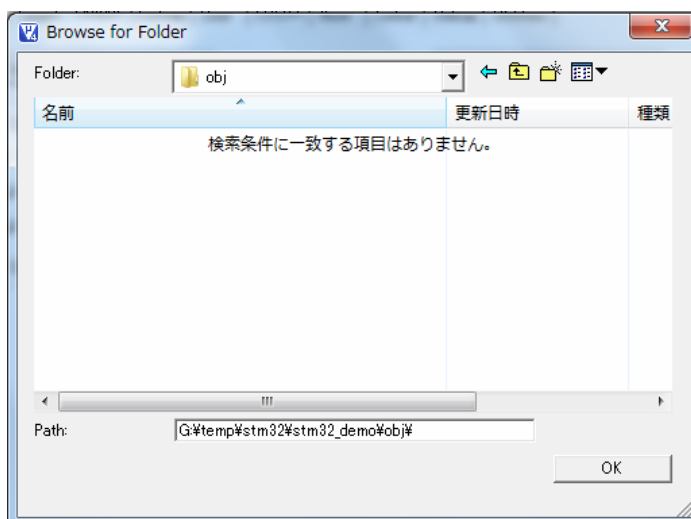
オプション設定画面で「output」タブを選択して、Hex ファイルを作成する選択肢にチェックを入れる。



上記画面で「Select Folder For Objects」ボタンを押して、出力フォルダを指定する。

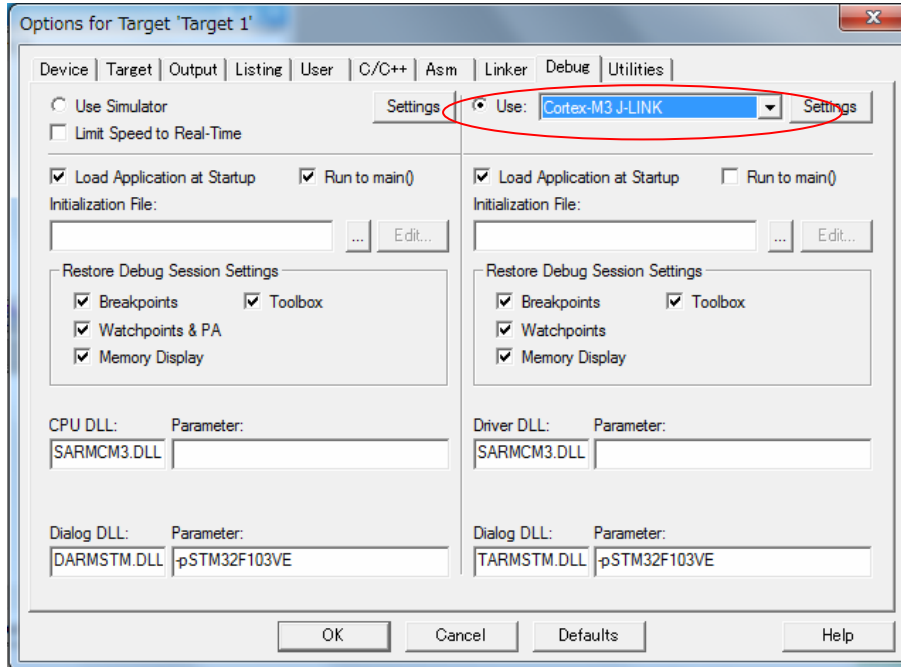


「obj」フォルダを作成して指定する。

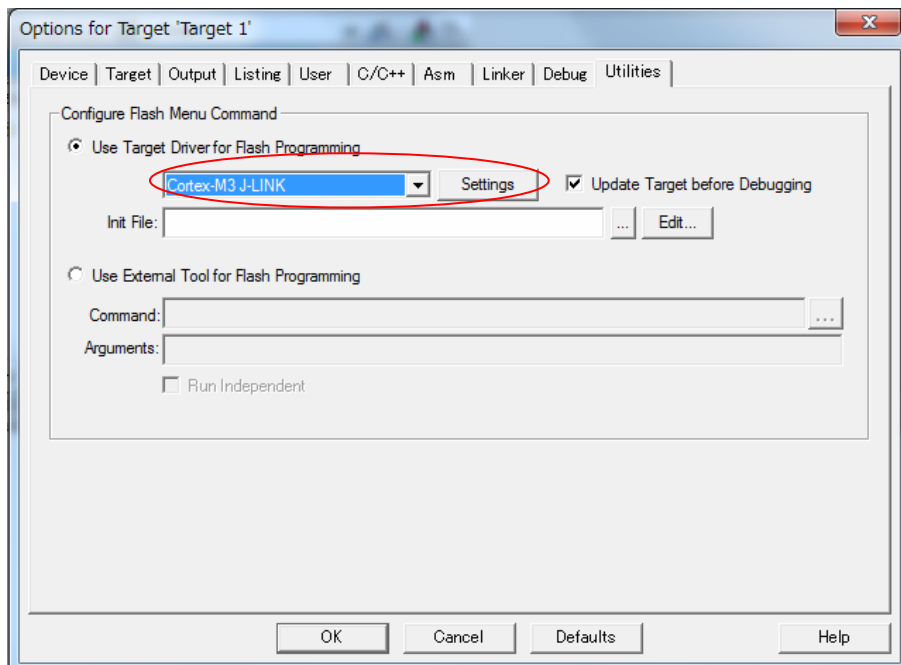


「OK」押してオプション設定画面に戻る。

「Listing」タブを選択して、上記と同じ手順で list フォルダを作成する。  
次は「Debug」タブを選択して、利用している JTAG を選択する。シミュレータでデバッグする場合はデフォルトの Use Simulator のままで良い。



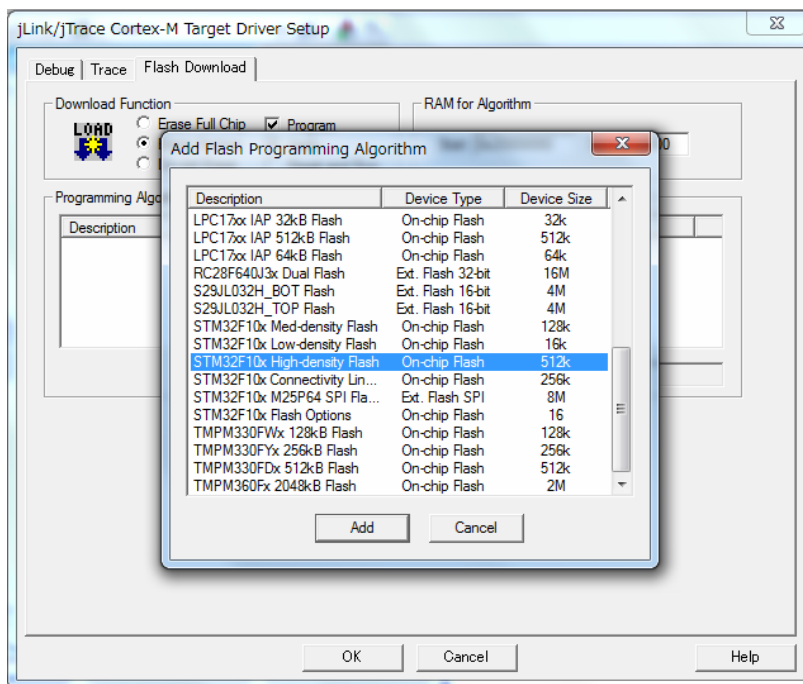
最後は「Utilities」タブを選択して、「Use Target Diver for Flash Programming」を選択する。ここは Debug タブで選択した JTAG と合わせて設定する。



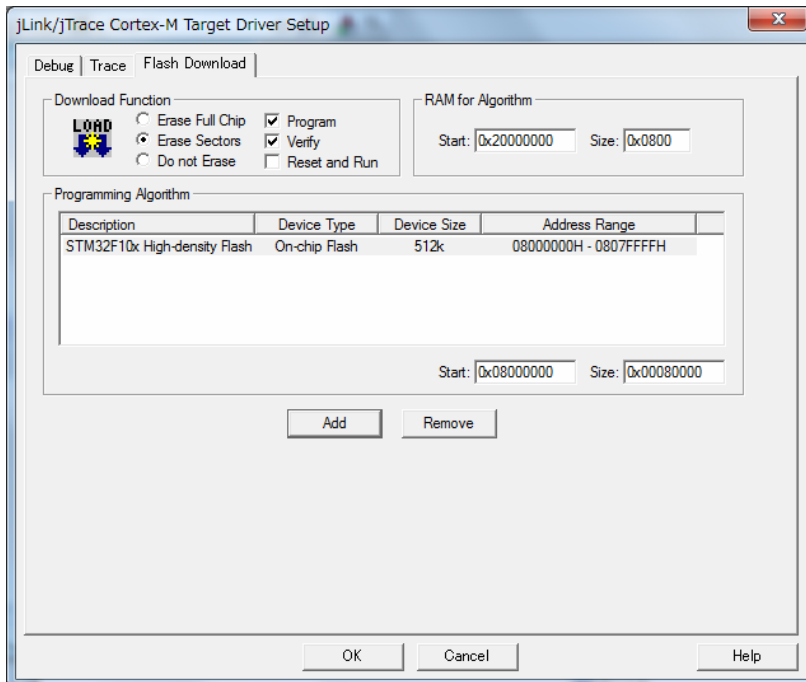
「Setting」ボタンを押すと、次の画面が表示される。



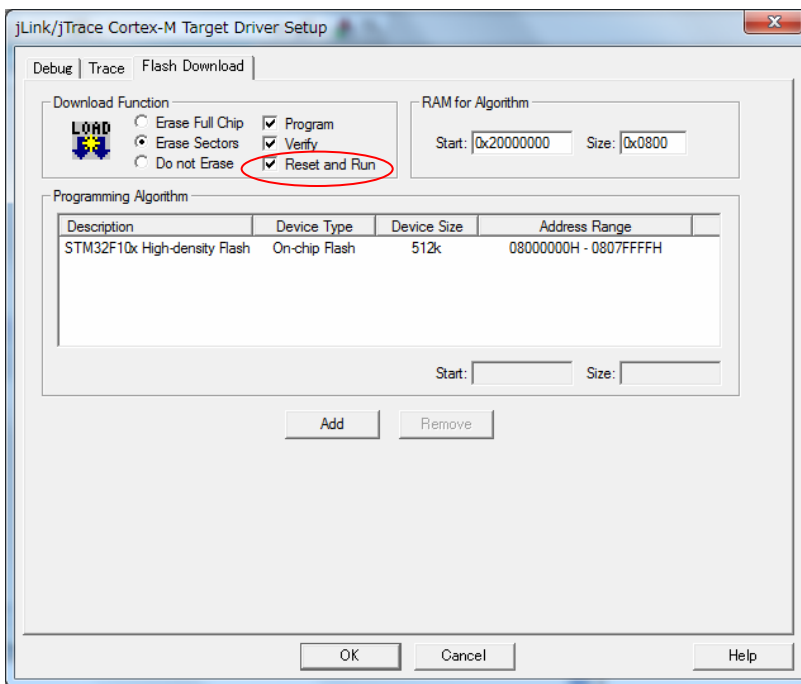
「Add」ボタンを押して、プログラムの書き込みアルゴリズムを設定する。



「Add」ボタン押すと、次の画面になる。



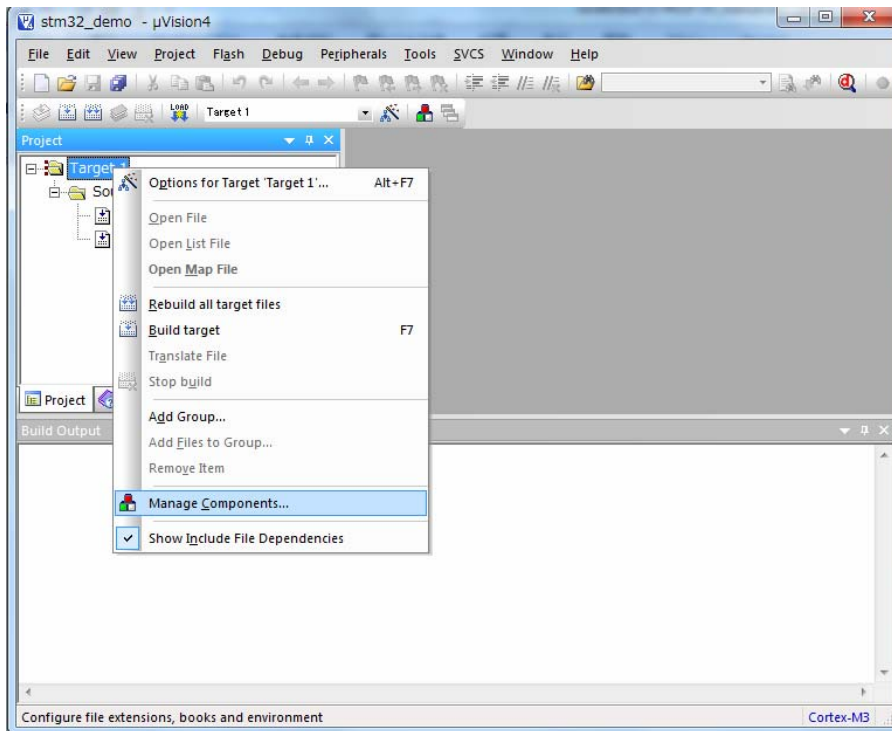
「Reset and Run」の所にチェックを入れて「OK」ボタンを押す。



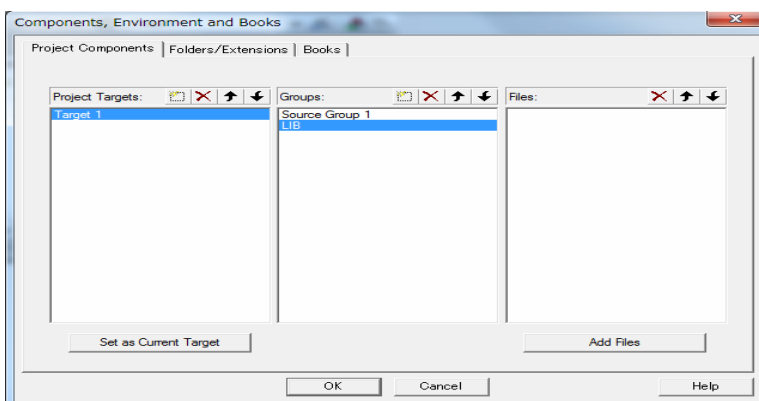
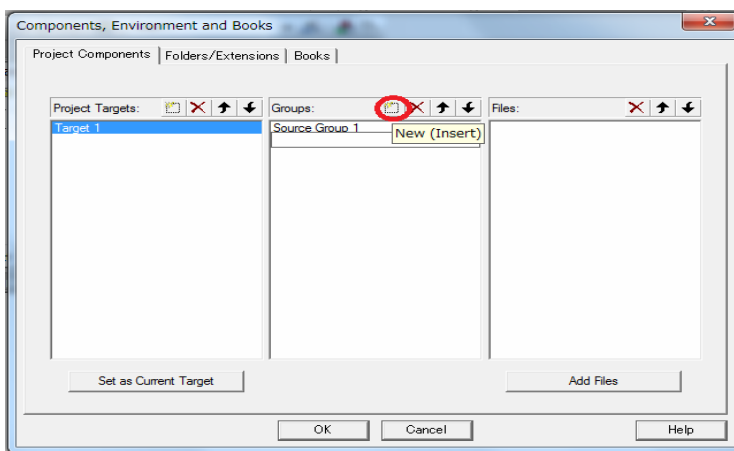
オプション設定画面に戻して「OK」ボタンを押す。

Target1 でマウスを右クリックして” Manage Components” を選択する。

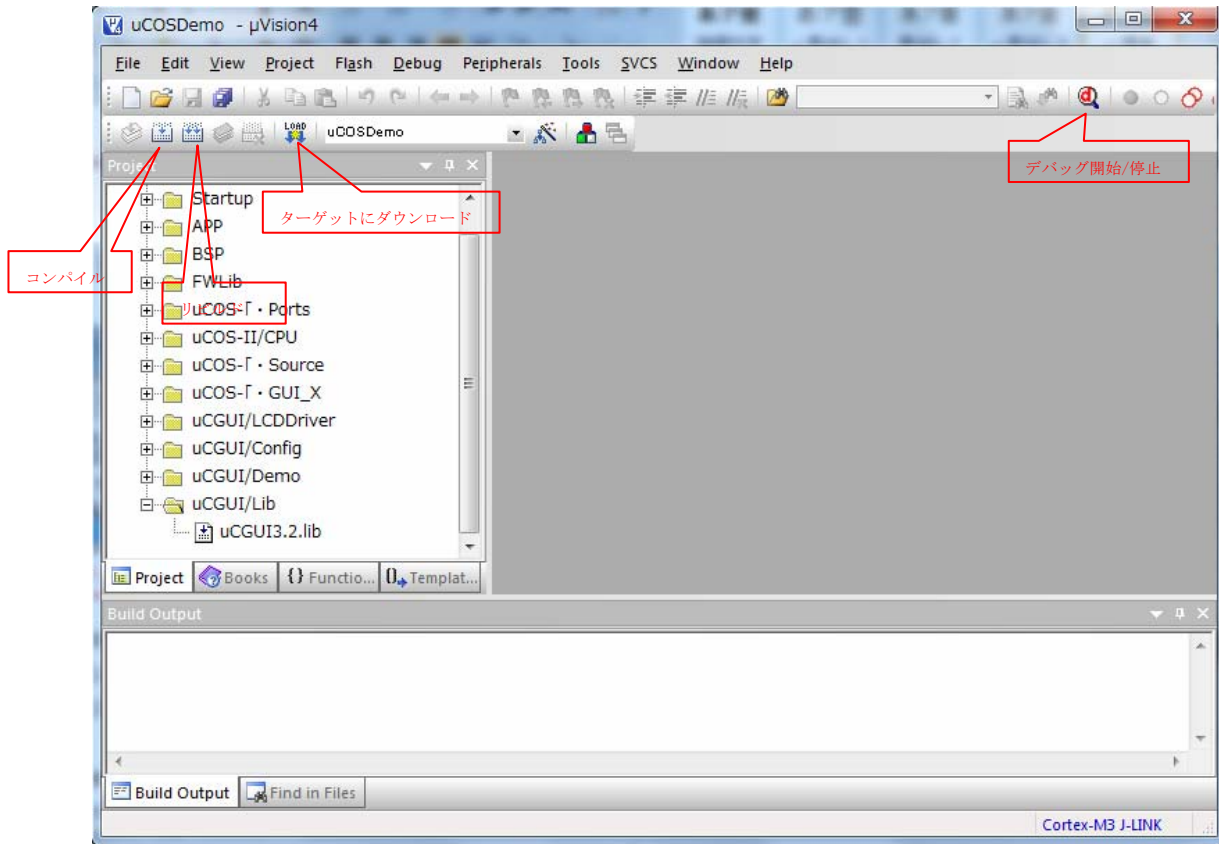




必要に応じてグループフォルダを追加する。LIB、APP など。



コンパイル、ビルド、ダウンロード、デバッグなどの操作。



以上