

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



株式会社日昇テクノロジー

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	2011/12/01
		-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)	

• 修正履歴

※ 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

× –	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	3
2.1 \$	STM32F103ZET6	3
2.2 M	Nor Flash	3
2.3 N	Nand Flash	3
2.4 \$	SRAM	7
2.5 \$	SPI Flash	7
2.6 \$	SDカードインタフェース	3
2.7	I2C EEPROM	3
2.8 A	Audio回路	3
2.9 1	Suner	3
2.10	Ethernet)
2.11	Boot option (起動オプション))
2.12	RTCリアルタイムクロック)
2.13	CANバスインタフェース)
2.14	ブザー)
		,
2.15	LED)
2. 15 2. 16	LED))
2. 15 2. 16 2. 17	LED)))
 2. 15 2. 16 2. 17 2. 18 	LED)))
 2. 15 2. 16 2. 17 2. 18 2. 19 	LED)) [
 2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 	LED)) [[
 2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 	LED))) L L
 2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 2. 22 	LED))) [[[
2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 2. 22 2. 23	LED. 10 KEY. 10 JTAG/SWDインタフェース. 10 電源変換と電源指示LED 11 5V DC電源入力インタフェース. 11 USBインタフェース. 11 COM1. 11 RS-485 インタフェース. 12	
2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 2. 22 2. 23 2. 24	LED	
2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 2. 22 2. 23 2. 24 2. 25	LED. 10 KEY. 10 JTAG/SWDインタフェース. 10 電源変換と電源指示LED 11 5V DC電源入力インタフェース. 11 USBインタフェース. 11 COM1 11 COM2 11 RS-485 インタフェース. 12 RS-232 とRS-485 の選択ジャンパ. 12 殊機能インタフェース. 12	
 2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 2. 22 2. 23 2. 24 2. 25 2. 26 	LED. 10 KEY. 10 JTAG/SWDインタフェース. 10 電源変換と電源指示LED 11 5V DC電源入力インタフェース. 11 USBインタフェース. 11 COM1 11 COM2 11 RS-485 インタフェース. 12 RS-232 とRS-485 の選択ジャンパ. 12 殊機能インタフェース. 12 拡張インタフェース. 12 拡張インタフェース. 12)))))) [[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 2. 22 2. 23 2. 24 2. 25 2. 26 2. 27	LED. 10 KEY. 10 JTAG/SWDインタフェース. 10 電源変換と電源指示LED. 11 5V DC電源入力インタフェース. 11 USBインタフェース. 11 COM1. 11 COM2. 11 RS-485 インタフェース. 12 RS-232 とRS-485 の選択ジャンパ. 12 殊機能インタフェース. 12 拡張インタフェース. 12 2.46無線モジュールインタフェース. 15)))))) [[[[2 2 2 2 3
 2. 15 2. 16 2. 17 2. 18 2. 19 2. 20 2. 21 2. 22 2. 23 2. 24 2. 25 2. 26 2. 27 2. 28 	LED. 10 KEY. 10 JTAG/SWDインタフェース. 10 電源変換と電源指示LED. 11 5V DC電源入力インタフェース. 11 USBインタフェース. 11 COM1. 11 COM2. 11 RS-485 インタフェース. 12 RS-232 とRS-485 の選択ジャンパ 12 殊機能インタフェース. 15 拡張インタフェース. 15 315MHz無線モジュールインタフェース. 15))))))) [[[[[[[[[[[[[[[[[[[[[[[[[[[[[[



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	
4.2 サンプルプログラム紹介	
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	
5.1 シリアルポートで書き込む	
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	
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	
7.1 KEILのインストール	
7.2 既存のプロジェクトから	
7.3 新しいプロジェクトの作成	

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ー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、初心者に最適です。

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、24LCO2(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x4、RESETx1

ユーザーLEDx4

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

- $RS485 \times 1$, $RS232 \times 2$ (DB9)
- CAN BUS x 1, SN65VHD230
- ブザーx1
- ポテンショメータ入力アナログ信号x1
- 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。

	$\overline{\nabla}$		
A0 🗖 1		8 🗖 Vcc	2
A1 2		7 🗆 WP	
A2 🗆 3		6 🗖 SCI	4
Vss 🗖 4		5 🗖 SD/	A

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。

LRCK	10	16	🗖 scкi
DATA I	2	15	III MS
BCK I	3	14	
PD	4	13	D MD
AGND	— 5	12	
HGND	— 6	11	
V _{COM} I	7	10	AIN
HOUTR I	8	9	HOUTL

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)	BOOTO (J10)	
ANY	2-3	Embedded user Flash (ディフォルト)モード
(1-2、2-3 or open)		※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 からもアクセスできる。
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 からもアクセスできる。
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(CR1220 ボタン型バッテリをサポートする)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 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 GPI0 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 GPI0 の PA0、PC13、PA8、PD3 で 制御しております。キー押下すると、その GPI0 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 接繋げます。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:

SW/LDD nin name	JTAG debug port		SW debug port		Pin
Swo-DP pin name	Туре	Description	Туре	Debug assignment	ment
TMS/SWDIO	1	JTAG Test Mode Selection	١/O	Serial Wire Data Input/Output	PA13
TCK/SWCLK	1	JTAG Test Clock	1	Serial Wire Clock	PA14
TDI	1	JTAG Test Data Input	-	-	PA15
TDO/TRACESWO	0	JTAG Test Data Output	-	TRACESWO if async trace is enabled	PB3
NTRST	1	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 に変換する必要です。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 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: <u>http://www.csun.co.jp/SHOP/2009061924.html</u>

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:

FSMC_D[15:0]	
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	FSMC_D1 FSMC_D3 FSMC_D5 FSMC_D7 FSMC_D7 FSMC_D9 FSMC_D11 FSMC_D13 FSMC_D13 FSMC_D15 FSMC_A0 RS FSMC_NOE RD PA1 BLACK_LIGHT PG7 INT PG8 BUSY F_CS LCD_SD_CS SD_CS

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	DO	
PD15	FSMC_D1	D1	
PDO	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



₩₩~~₩ 株式会社日昇テクノロジー

			(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ようです:

¥STM32F10x_StdPeriph_Lib_V3.3.0

	🗅 htmresc
ī	🗀 Libraries
	🗀 Project
	🗀 Utilities
	🕘 Release_Notes
	울 stm32f1 0x_stdperiph_lib_um

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

🗉 🚞 Libraries	
🗉 🛅 CMSIS	
🗉 🚞 СМЗ	
🚞 CoreSupport	Cortex-M3カーネル関連ファイル
🖃 🚞 DeviceSupport	
🗉 🚞 ST 🛛 ST M3	32F10x起動ファイル
🗉 🛅 STM32F1	0x
🚞 Documentation	
🗉 🛅 ST M32_USB-FS-Devic	e_Driver
😑 🛅 ST M32F1 0x_StdPeriph_I	Driver
Constant inc STM32F10>	〈標準周辺機器のドライバ

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.FMT uner 🗉 🚞 16.2.4GWireless 🗉 🚞 17.Calendar 🗉 🚞 18.ENC28.J60 19.TFTLCD_2.8inch_9320and8989 🗉 🚞 19.TFTLCD_3.2inch_9320and8989 🗉 🚞 20.SD 🗉 🚞 21.MP3Player 🗉 🛅 22.uCOS+uCGUI(FSMC3.2inch_9320_8989) 🗉 🛅 ST M32 F1 Ox_StdPeriph_Examples 🗉 🛅 STM32F1 0x_StdPeriph_Template Project¥01. LED を開くイメージ:

EWARMUS	- HITOP	MDK-ARM	RIDE
TrueSTUDIO	main	■ readme	Release_Notes_for_ST M32F1 0x
	C ファイル	テキスト ドキュメント	360seURL
	5 KB	6 KB	20 KB
stm32f10x_conf	stm32f1 0x_it	stm32f10x_it	
H ファイル	C ファイル	H ファイル	
4 KB	5 KB	2 KB	

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:



🔣 Project – 🛛 Vision4	
<u>File E</u> dit <u>V</u> iew <u>P</u> roject Fl <u>a</u> sh	Debug Peripherals Tools SVCS Window Help
	金 い き き き だ だ 通 一 ・ 3 の () 0 い ク 余 回 ・ 3
: 🕸 🏦 🎬 🥔 🖳 🙀 LED	🗷 🎊 🚠 🔁
Project 👻 🖛 🗙	🖹 main.c 🛐 stm32f10x_gpio.c 💱 stm32f10x_rcc.c 👻 🗙
Image: Construction of the construc	001 /***********************************
Build Output	▲ 1 X
	Cortex-MB J-LINK

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I0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。

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です。

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
}GPIOSpeed_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

下可能への挑戦

本ボードは4つのLEDを搭載しております。それぞれGPI0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。GPI0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、標準できには5Vになると入れると、固 定周波数の音声を出します。

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。GPI0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。GPI0のPINと繋いでいる抵抗 R14 は電流過大を防ぐ為で す。STM32のI0の最大電流は25mA となっております。

1	Output current sunk by any I/O and control pin	25
10	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 を利用しております。



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 GPI0 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。





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 の信号が変化した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さる時無線モジュ
 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[~]4 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	GPI0 ピン
DS2	PF6
DS3	PF7
DS4	PF8
DS1	PF9

GPI0 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	DO	PAO
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 APB1 RSTR) 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 を利用しております。



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





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:

8	🗞 stm32 - ハイパーターミナル	
7	ファイル(E) 編集(E) 表示(V) 通信(<u>C</u>) 転送(<u>T</u>) ヘルプ(<u>H</u>)	
C		
		<u> </u>
	WWW.CSUN.CO.JP	
	COM1 test program	
	COM1 test	
	COM1 test	_
		~
り	、 刃断 ANSIW 115200 8-N-1 SCROLL CAPS NUM キャ エコーを印	

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ぎます。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です。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。

GPI0 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。







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 は発送モードになり、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 ピント繋いでいます。



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。





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

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:



株式会社日昇テクノロジー



GND

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 0x10 0x20 0x30 0x40 0x50 0x60 0x80 0x80 0x90 0x80 0x90 0x00 0x00 0xc0 0xc0 0xc0 0xc0 0xc	0X1 0X21 0X21 0X31 0X51 0X51 0X51 0X51 0X51 0X51 0X51 0X5	0x2 0x12 0x22 0x32 0x42 0x52 0x62 0x72 0x82 0x82 0x82 0x82 0x82 0x82 0x62 0x62 0x62 0x62 0x62 0x62 0x62 0x6	0x3 0x13 0x23 0x33 0x43 0x53 0x63 0x63 0x93 0x93 0x93 0x03 0xc3 0xc3 0xc3 0xc3 0xc3 0xc3 0xc	0x4 0x14 0x24 0x34 0x54 0x54 0x64 0x74 0x84 0x94 0x94 0x64 0x64 0xc4 0xc4 0xc4 0xc4 0xc4	0x15 0x25 0x35 0x45 0x65 0x65 0x95 0x95 0x95 0x05 0x05 0xc5 0xc5 0xc5 0xc5	0x6 0x16 0x26 0x36 0x56 0x66 0x76 0x86 0x86 0x86 0x86 0x86 0x86 0x66 0x6	0x7 0x17 0x27 0x37 0x47 0x57 0x67 0x77 0x87 0x97 0x87 0x97 0x67 0x67 0x67 0x67 0x67 0x67 0x67	0×8 0×18 0×28 0×38 0×58 0×58 0×58 0×88 0×88 0×88 0×88 0×58 0×5	0x9 0x19 0x29 0x39 0x69 0x69 0x69 0x89 0x89 0x89 0x89 0x89 0x89 0xc9 0xc9 0xc9 0xc9	Oxa Ox1a Ox2a Ox3a Ox5a Ox6a Ox6a Ox6a Ox9a Ox9a Ox9a Oxba Oxca Oxca Oxca Oxca	0xb 0x2b 0x2b 0x3b 0x6b 0x6b 0x6b 0x8b 0x8b 0x9b 0xab 0xcb 0xcb 0xcb 0xcb	0xc 0x1c 0x2c 0x3c 0x5c 0x6c 0x6c 0x8c 0x9c 0x8c 0x9c 0xbc 0xbc 0xcc 0xc 0xc 0xc 0xc 0xc 0xc 0xc 0xc 0	0xd 0x1d 0x2d 0x3d 0x5d 0x6d 0x6d 0x8d 0x8d 0x8d 0x8d 0x8d 0xcd 0xcd 0xcd 0xcd 0xcd 0xcd	0x2e 0x2e 0x2e 0x3e 0x5e 0x6e 0x7e 0x8e 0x9e 0x9e 0xbe 0xce 0xce 0xce 0xce 0xce	0xf 0x1ff 0x2ff 0x3ff 0x5ff 0x8ff 0x8ff 0x8ff 0x8ff 0x2ff 0x2ff
Read	data														
0x0 0x10 0x20 0x30 0x40 0x50 0x70 0x80 0x80 0x80 0x80 0x80 0x80 0x8	0x1 0x11 0x21 0x31 0x41 0x51 0x51 0x71 0x81 0x91 0x81 0x81 0x81 0x81 0x81 0x81 0x81 0x8	0x2 0x12 0x22 0x32 0x52 0x62 0x62 0x82 0x82 0x82 0x82 0x82 0x82 0x82 0x2 0x2 0x2 0x2 0x2 0x2 0x2 0x2 0x2 0x	0x3 0x13 0x23 0x33 0x43 0x63 0x73 0x83 0x83 0x83 0x83 0x83 0x83 0x83 0x8	0x4 0x14 0x24 0x34 0x44 0x54 0x64 0x74 0x84 0x84 0x84 0x84 0x84 0x84 0x84 0x8	0x5 0x15 0x25 0x45 0x65 0x75 0x85 0x85 0x85 0x85 0x85 0x85 0x85 0x8	0x6 0x16 0x26 0x46 0x66 0x76 0x86 0x86 0x86 0x86 0x86 0x86 0x86 0x8	0x7 0x17 0x27 0x37 0x47 0x57 0x67 0x77 0x87 0x87 0x87 0x87 0x87 0x87 0x8	0x8 0x18 0x28 0x38 0x48 0x58 0x58 0x78 0x88 0x88 0x88 0x88 0x88 0x88 0x8	0x9 0x19 0x29 0x49 0x59 0x69 0x79 0x89 0x89 0x89 0x89 0x89 0x89 0x89 0x8	0xA 0x1A 0x2A 0x3A 0x4A 0x5A 0x7A 0x8A 0x9A 0x8A 0x8A 0x8A 0x8A 0x8A 0x8A 0x8A 0x8	0xB 0x1B 0x2B 0x3B 0x4B 0x5B 0x6B 0x7B 0x8B 0x8B 0x8B 0x8B 0x8B 0x8B 0x8B 0x8	0xC 0x1c 0x3c 0x4c 0x5c 0x6c 0x7c 0x8c 0x9c 0xAc 0xAc 0xAc 0xAc 0xAc 0xAc	0xD 0x1D 0x2D 0x3D 0x4D 0x5D 0x6D 0x7D 0x8D 0x8D 0x8D 0x8D 0x8D 0x8D 0x8D 0x8	0xE 0x1e 0x2e 0x3e 0x4e 0x5e 0x7e 0x8e 0x8e 0x8e 0x8e 0x8e 0x8e 0x8e 0x8	0xF 0x1F 0x3F 0x5F 0x5F 0x6F 0x6F 0x8F 0x8F 0x8F 0x8F 0x8F 0x8F 0x8F 0x8

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。





GPI0 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
S0	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。





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 ビットのカウントダウンタイマーで、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 E	1	44 🗖 A17
A1 C	2	43 A16
A2 [3	42 A15
A3 C	4	41 OE
A4 C	5	40 UB
CE C	6	39 🗖 LB
VOO E	7	38 VO15
VO1 C	8	37 🗖 1014
VO2 C	9	36 VO13
V03 C	10	35 VO12
Vcc E	11	34 🗖 GND
SND E	12	33 🗖 Vcc
1/04 E	13	32 VO11
105 C	14	31 🗖 VO10
V06 C	15	30 🗖 1/09
V07 C	16	29 🗖 VOB
WE C	17	28 NC
A5 0	18	27 🗖 A14
A6 [19	26 🗖 A13
A7 [20	25 A12
AB C	21	24 🗖 A11
A9	22	23 A10


A0-A17	Address Inputs	LB	Lower-byte Control (I/O0-I/O7)
I/O0-I/O15	Data Inputs/Outputs	UB	Upper-byte Control (I/O8-I/O15)
CE	Chip Enable Input	NC	No Connection
OE	Output Enable Input	Vcc	Power
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	X
ファイル(E) 編集(E) 表示(V) 通信(<u>C</u>) 転送(<u>T</u>) ヘルプ(<u>H</u>)	
SRAM test DS1 flash: program normal DS2 on: sram access OK DS3 on: sram access NG 	< III
接続 0:00:24 自動検出 115200 8-N-1 SCROLL CAPS 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と繋ぎます。



株式会社日昇テクノロジー



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 ₁₉ -A ₀	Address Inputs	To provide memory addresses. During Sector-Erase A ₁₉ -A ₁₁ address lines will select the sector. During Block-Erase, A ₁₉ -A ₁₅ address line will select the block.	
DQ ₁₅ -DQ ₀	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	
VDD	Power Supply	To provide power supply voltage:	3.0-3.6V for SST39LF160 2.7-3.6V for SST39VF160
Vss	Ground		
NC	No Connection	Unconnected pins	

ソフトウェア設計

下可能への挑戦

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。



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と繋ぎます。



HY27UF081G2AはHynix社製の1Gbit Nand Flash。 HY27UF081G2Aのピン配置:







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。





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ア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 GPI0 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。



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:





GPI0 ピンと 2.4G 無線モジュール nRF24L01 ピン対応関係:

nRF24L01 ビン	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 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(E) 編集(E) 表示(V) 通信(C) 転送(I) ヘルプ(H)	
2.46 wireless test Please set NRF24L01 work mode USER1: set NRF24L01 to receiver USER2: set NRF24L01 to transmiter TAMEPR: quit NRF24L01 receiver setting OK Waiting for data Received data: Received data: Received data: "#\$%&'()*+,/0123456789:;<=>?@ Received data: "#\$%&'()*+,/0123456789:;<=>?@A Received data: #\$%&'()*+,/0123456789:;<=>?@A Received data: #\$%&'()*+,/0123456789:;<=>?@AB Received data: \$%&'()*+,/0123456789:;<=>?@ABCD Received data: \$%&'()*+,/0123456789:;<=>?@ABCD Received data: {'*+,/0123456789:;<=>?@ABCDE Received data: {'*+,/0123456789:;<=>?@ABCDE Received data: {'*+,/0123456789:;<=>?@ABCDE Received data: {*+,/0123456789:;<=>?@ABCDEFF Received data: {\	
<u>ニーーーーーーーーーーーーーーーーーーーーーーーーーーーーーーーーーーーー</u>	*

🏶 stm32-B - ハイパーターミナル	
ファイル(E) 編集(E) 表示(V) 通信(C) 転送(I) ヘルプ(H) D 🕝 🍘 🕼 💾 🖆	
2.4G wireless test Please set NRF24L01 work mode USER1: set NRF24L01 to receiver USER2: set NRF24L01 to transmiter TAMEPR: quit NRF24L01 transmiter setting OK Please check if the receiver is normal Sending data: !"##%&'()*+,/0123456789:;<=>?@ Sending data: "##%&'()*+,/0123456789:;<=>?@A Sending data: ##%&'()*+,/0123456789:;<=>?@AB Sending data: #%&'()*+,/0123456789:;<=>?@AB Sending data: \$%&'()*+,/0123456789:;<=>?@ABCD Sending data: \$% ()*+,/0123456789:;<=>?@ABCD Sending data: \$()*+,/0123456789:;<=>?@ABCDEF Sending data: ()*+,/0123456789:;<=>?@ABCDEFG Sending data: ()*+,/0123456789:;<=>?@ABCDEFG Sending data: ()*+,/0123456789:;<=>?@ABCDEFG Sending data: *+,/0123456789:;<=>?@ABCDEFGH Sending data: *+,/0123456789:;<=>?@ABCDEFGH Sending data: *+,/0123456789:;<=>?@ABCDEFGH Sending data: +,/0123456789:;<=>?@ABCDEFGHI Sending data: +,/0123456789:;<=>?@ABCDEFGHI Sending data: +,/0123456789:;<=>?@ABCDEFGHI Sending data: +,/0123456789:;<=>?@ABCDEFGHI Sending data: +,/0123456789:;<=>?@ABCDEFGHI Sending data: -, ./0123456789:;<=>?@ABCDEFGHIJK Sending data:/0123456789:;<=>?@ABCDEFGHIJKL	
切断 自動検出 115200 8-N-1 SCROLL CAPS NUM キャ エコーを印	.::

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	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

ENC28J60 ピンとボードの GPI0 ピン対応:

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、<u>http://192.168.1.15/CSUN</u>を入力してHTTPアクセスができます。

🛎 http://192.168.1.15/CSUN – Windows Internet Explorer			
ファイル(E) 編集(E) 表示(V) お気に入り(A) ツール(I) ヘルプ(H) 🦧	7		
🕝 戻る 🔹 📀 🔹 🛃 🌮 検索 🌟 お気に入り 🤣 😒 🌭 🖬 🔹 🔜 🥸 🤷 🥸			
アドレス(D) 🕘 http://192.168.1.15/CSUN ←输入关键词 探望 🔽 移動	勆		
DS1 LED status: ON <u>[refresh]</u>	^		
<u>Close DS1 LED</u>			
STM32 board WEB test	~		

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(E) 編集(E) 表示(<u>V</u>) 通信(<u>C</u>) 転送(<u>T</u>) ヘルプ(<u>H</u>)	
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]	
<pre>[192.168.1.15] send ARP response receive ICMP package from[1.101.192.168] [192.168.1.15] send ICMP response</pre>	
[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 エコーを印	

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:



株式会社日昇テクノロジー



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_DO	メディアカード/SD/SDI0 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.SD¥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:



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 (H)
 ■ itest-video
 ※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。





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	12S3_WS	左右チャネルクロック、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:



	PW PA (TOP	CKAGE VIEW)	
	10	16	
	2	15	II MS
BCK	3	14	
PD I	4	13	
AGND	5	12	
HGND	6	11	
VCOM I	7	10	
HOUTR I	8	9	HOUTL

TERMINAL			DECODINE ION		
NAME	NO.	1/0	DESCRIPTION		
AGND	5	-	Analog ground. This is a return for V _{CC} .		
AIN	10	1	Monaural analog signal mixer input. The signal can be mixed with the outputs of the L- and R-channel DACs.		
BCK	3	1/0	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	1	Serial audio data input		
HGND	6	-	Analog ground. This is a return for V _{HP} .		
HOUTL	9	0	L-channel analog signal output of the headphone amplifiers		
HOUTR	8	0	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	1	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	1	Mode control port select. The control port is active when this terminal is low.		
PD	4	1	Reset input. When low, the PCM1770 device is powered down, and all mode control registers are reset to default settings.		
SCKI	16	1	System clock input		
Vcc	12	-	Power supply for all analog circuits except the headphone amplifier.		
V _{COM}	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 _{HP} nominal.		
VHP	11	-	Analog power supply for the headphone amplifier circuits. The voltage level must be the same as V _{CC} .		

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:



🗞 stm32 - ハイパーターミナル	
ファイル(E) 編集(E) 表示(V) 通信(C) 転送(I) ヘルプ(H)	
	<u>^</u>
Audio play test	
Serial port initial finish PCM1770 SPT interface initial finish	
AudioFile_Init	
WaveFileStatus Valid_WAVE_File	
123_00111g	
CODEC_Config Dependence SandDummuDete	
GetVar_SendDummyData=0	
MUSICBUTTERLURRENTPOS = 01	
MusicBufferCurrentPos = 0!	
	*
	>
切断 ANSIW 115200 8-N-1 SOROLL CAPS NUM キャ エコーを印	



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。



InstallShield Wizard		X
License Agreement Please read the following license agree carefully.	ement	
Press the PAGE DOWN key to see the res	t of the agreement.	
SOFTWARE LICENSE AGREEMENT By using this Licensed Software, You a terms and conditions of this License A Licensed Software until You have read terms and conditions. The use of the automatically the acceptance of the fo Please indicate your acceptance or NON ACCEPT) or (I DO NOT ACCEPT) as indica	re agreeing to be bound by the greement. Do not use the and agreed to the following icensed Software implies blowing terms and conditions. -acceptance by selecting (I ted below in the media.	
Do you accept all the terms of the pred choose No, the setup will close. To V1.3, you must accept this agreement. InstalShield	ceding License Agreement? If you install Flash Loader Demonstrator	
	< Back Yes No	

ユーザー名と会社名を入力して、「Next」ボタンを押す。

InstallShield Wizard
Customer Information Please enter your information.
User Name:
csun
Company Name:
csun
Install this application for:
Anyone who uses this computer (all users)
C Only for me (dwtech)
InstallShield
< Back Next > Cancel





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。

InstallShield Wizard			
	InstallShield Wizard Complete Flash Loader Demonstrator V2.1.0 Setup is almost complete. Choose the options you want below. ▼ I would like to view the Release Notes. ▼ I would like to launch Flash Loader Demonstratc Click Finish to complete Flash Loader Demonstrator V2.1.0 Setup.		
K Back Finish Cancel			

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:(<u>http://www.csun.co.jp/SHOP/2010040601.html</u>)

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。

🤣 Flash loader demonstrator (v	1.3)				
	STMicroel Select the co	ectronics 利用している ルポートを指す	シリア ings. ますろ	then click next to	open connection.
		COM1 57600	Pa Sto	rity Even op Bits 1 neout(s) 5	× *
FLASHIT					
		Back	<u>N</u> ext	Qancel	<u>E</u> inish

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。)

🧼 Flash loader demonstrator (v1.3)
	STMicroelectronics
	Target is readable. Please click "Next" to proceed.
FLASRIT	Flash Size 512 Kb
	Back Next Cancel Einish

「Next」ボタンを押す。

🧼 Flash loader demonstrator (v1.3)						
	STMicr	oelectronics				57/	
	Please, selec	t your device in	the target list				
	Target	STM32F10xxE	xx			-	
913	PID (h)	0414	_				
	Version	2.2	_				
	Flash mappin	g					
and free and an interest	Name	Start addre	End address	Size	R W.		
	🔦 Page0	0x 8000000	0× 80007FF	0x800 (2K)	66		
Contraction of the Association o	💊 Page1	0× 8000800	0× 8000FFF	0×800 (2K)	88		
Street Wilson or Constant Constant of Constant of Constant	💰 Page2	0× 8001000	0× 80017FF	0×800 (2K)	88		
FLACHT	🛸 Page3	0× 8001800	0× 8001 FFF	0×800 (2K)	88		
TEROIIII	🔦 Page4	0x 8002000	0× 80027FF	0×800 (2K)	66		
Concession of Concession, Stationer,	🦠 Page5	0× 8002800	0× 8002FFF	0×800 (2K)	66		
A REAL PROPERTY AND A REAL	🦠 Page6	0× 8003000	0× 80037FF	0×800 (2K)	66		
Barris and a state of the second	🦠 Page7	0× 8003800	0× 8003FFF	0×800 (2K)	66		
	🦠 Page8	0× 8004000	0× 80047FF	0×800 (2K)	66		
	🦠 Page9	0× 8004800	0× 8004FFF	0×800 (2K)	66		
1 10	🦠 Page10	0× 8005000	0× 80057FF	0×800 (2K)	66		
11111	🛝 Paga11	0~ 2005200	0~ 8005EEE	0∼800 (2K)	A A		
all still have	Legend :	🞦 Prote	ected	🕤 UnProtected			
		<u>B</u> ack	<u>N</u> ext	<u>C</u> ar	icel	<u>F</u> inisł	1

そのまま「Next」ボタンを押す。



Flash loader demonstrator (v1.3)					
	STMicroelectronics				
	C Erase				
	All C Selection				
	Download from file D:¥tmp¥stm32¥uCOSDemo.bin				
	Erase necessary pages C Erase all pages (faster)				
	Optimize (Remove some FFs) Verify after download				
FLASRit	C Upload from device				
	C Enable/Disable Flash protection				
114	C Edit option bytes				
	Back Next Qancel Finish				

書き込む*.Binファイルを選択して、「Next」ボタンを押す。

🔎 Flash loader demonstrator (v1.3) 📃 🗆 🔀				
	STMicro	oelectronics		
	Target Map file Operation File name File size	STM32F10xxExx STM32F10xxExxSTmap DOWNLOAD D¥tmp¥stm32¥uCOSDemo.bin 234.18 KB (239804 bytes)		
PLASE	Status Time	7.03 KB (7194 bytes) of 234.18 KB (239804 bytes) 00:00:05 Downloading data 3%		
		Back Next Cancel Einish		



Flash loader demonstrator (v1	.3)	
	STMicro	pelectronics
5	Target	STM32F10xxExx
	Map file	STM32FTUxxExxSTmap
	Operation File name	DOWNLOAD D:¥tmp¥stm32¥uCOSDemo.bin
FLASHIT	File size Status Time	234.18 KB (239804 bytes) 234.18 KB (239804 bytes) of 234.18 KB (239804 bytes) 00:02:44
		Download operation finished successfully
		Back Next Cancel Finish

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: <u>http://www.csun.co.jp/SHOP/2009121901.html</u>)。

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 SEGGER 社様のソフトウエ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、ディフォルトのままで行ってください。



▼■■~の■■ 株式会社日昇テクノロジー

🔏 Choose Destinat	ion Location	×
	Setup will install J-Link ARM V3.20h in the following folder. To install into a different folder, click Browse, and select another folder. You can choose not to install J-Link ARM V3.20h by clicking Cancel to exit Setup. Destination Folder C:\\SEGGER\JLinkARM_V320h <u>Browse</u>	
	< <u>B</u> ack <u>Next></u> Cancel	

2	Please read the following license agreement. Use the scroll to view the rest of this agreement.	bar
	Important - Read carefully: This license is a legal agreement between YOU (either an individual or a single entity) and SEGGER Microcontroller Systeme GmbH. By downloading and/or using J-Link ARM software you agree to be bound by the terms of this agreement. 1. LICENSE AGREEMENT In this agreement "Licensor" shall mean SEGGER Microcontroller Systeme GmbH (called SEGGER) except under the following circumstances:	<
	Do you accept all the terms of the preceding license agreen If so, click on the Yes push button. If you select No, Setup close.	nent? will







🔏 Installation Co	aplete	×
	J-Link ARM V3.20h has been successfully installed. Press the Finish button to exit this installation.	
	< <u>B</u> ack Einish Cancel	

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...) :



SEGGER J-	Flash ARM ¥4.02c	[D:\SEGGER\JLinkART_V402c\Default.jflash *]	
<u>F</u> ile <u>E</u> dit <u>V</u> ie	w <u>T</u> arget <u>Options W</u> i	dow <u>M</u> elp	
J Project	- De Project s	ttings Alt-F7	
Name	Value	tings	
Connection	USB [Device 0]		
Target interface	JTAG		
Init ITAG speed	51/17		
JTAG speed	Auto recognition		
TAP number	<not used=""></not>		
IR len	<not used=""></not>		
CPU	Auto		
Endian	Little		
Check core Id	No		
Core Id	0x0		
Use target RAM	No		
RAM address	0x0		
RAM size	4 KB		
Flash memory	Auto detection		
Base address	0x0		
Urganization	16 bits x 1 chip	SEGGER	
LOG			🛛
Application log - J-Flash ARM - JLinkARM dll Reading flash d - List of flas Reading MCU dev - List of MCU Creating new pr - New project	; started V4.02c (Jr-Flash compile) levice list [D:\SEGGEN, h devices read success rice list [D:\SEGGEN,L devices read successful oject file [D:\SEGGEN, created successfully	d Feb 27 2009 20:37:54) eb 27 2009 20:37:19) Lind&M Y402-CHTC\Flash.esv] ully (525 Devices) do&M Y402-CHTC\Flash\MCU.esv] ly (311 Devices) Lind&MM_Y402-ChDefault.jflash]	×.
<			8
how project set	ttings dialog		

STM32 ボードの CPU 型番を選択する。

oject settir	ngs			? ×
General T	arget Interface CPU	Flash Producti	on	
☞ Device ○ Core	ST STM32F103ZE	۲ ۲ ۲	Check core ID ID 38A00477 Use target RAM (faster) Addr 20000000 64 KB	
Use follow # Actio 0 Rese	ing init sequence: n Value0 t 0	Value1 Oms	Comment Reset and halt target	
Add	Insert D	elete Edit	Up Down	
			OK +	ッンセル 適用(A)



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。

SEGGER J-Flash ARM V4.10f - [G:¥01_work	¥embeded¥JLinkARM_V41	0f¥Samples¥JFlash¥ProjectFiles¥s	tm32f103vc.jflash]
File Edit View Target Options Window	Help		
🔜 Open data file		×	
Good and a set of the set of	2) • Obj • •	Objの検索 の	
整理 ▼ 新しいフォルダー		=	
3 00_bak ^	名前	更新日時	
01_work	uCOSDemo.hex	2010/05/07 23	
02_tools			
ehidenweb2			
🖟 KuGou			
🔒 MSOCache 🗏			
ppsvodcache			
lemp			
📕 Vmware 👻	∢ [+	
ファイル名(N): uCOSDe	mo.hex 🔹	Intel Hex files (*.hex) 🔹	
		開く(0) キャンセル	
		h.	
 JTAG speed: 5 KHz (Fixed) Initializing CPU core (Init sequence) Initialized curpersofully 			
 JTAG speed: 1000 kHz (Auto) J-Link found 2 JTAG devices. Core ID: 0x3BA0047 	7 (Cortex-M3)		
- Connected successfully			-
<)			
Ready		Connected	Core Id: 0x3BA00477 Speed: 1000 kH

「開く(0)」をクリックする。



SEGGER J-Flash ARM V4.10f - [G:¥	01_work¥embeded¥JLinkARM_V410f¥Samples¥JFlash¥ProjectFiles¥stm32f103vc.jflash *]			
File Edit View Target Options	Window Help			
Project - st 🗖 🗉 🔀	G:¥temp¥stm32¥redbull¥uCOS_UCGUI(3.2)¥Obj¥uCOSDemo.hex	- • ×		
Name Value 🔺	Address: 0x8000000 x1 x2 x4			
Connection USB [Device 0] Target interface JTAG	Address 0 1 2 3 4 5 6 7 ASCII	_		
Init JTAG speed 5 kHz JTAG speed Auto recognition	8000000 30 44 00 20 01 01 00 08 0D 8000008 05 01 00 08 07 01 00 08	_		
IR len <not used=""></not>	8000010 09 01 00 08 0B 01 00 08			
MCU ST STM32F103ZE	8000020 00 00 00 00 00 00 00 00 8000028 00 00 00 00 11 01 00 08			
Core Id Yes Core Id 0x3BA00477	8000030 OF 01 00 08 00 00 00			
RAM address 0x20000000 RAM size 64 KB	8000040 FF 03 00 08 01 04 00 08			
Flash memory STM32F10xxE inten Manufacturer ST Size 512 KB	8000048 03 04 00 08 05 04 00 08 8000050 5D 04 00 08 5F 04 00 08]			
Flash Id 0x0	8000060 61 04 00 08 63 04 00 08 ac 8000060 65 04 00 08 67 04 00 08 eg	-		
J_Link firmware: V120 (J-Link ARM V8 compiled Jan 29 2010 19:34:13) J-Link firmware: V120 (J-Link ARM V8 compiled Jan 29 2010 19:34:13) J-MG speed: 5 kHz (Fixed) J-Initalized successfully				
- J-Link found 2 JTAG devices. Core ID: 0x3BA00477 (Cortex-M3) - Connected successfully Opening data file [Q¥tempVstm32WedbullWuCOS_UCQUIX32WOb¥UcOSDemo.hex] _ - Data file opened successfully (289804 bytes, 1 range, CRC = 0x5ED070A7)				
Ready	Connected Core Id: 0x3BA00477	Speed: 1000 kH		

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:



SEGGER J-Fla	ash ARM V4.10f - [G:¥	01_work¥embeded¥JLinkARM_V410f¥Samples¥JFlash¥ProjectFiles¥stm32f103vc.jflash *] 🛛 📼 🛛			
File Edit Viev	w Target Options	Window Help			
Project - st	🗆 🖂 📖	G:¥temp¥stm32¥redbull¥uCOS_UCGUI(3.2)¥Obj¥uCOSDemo.hex	3		
Name	Value 🔺	Address: 0x8000000 x1 x2 x4			
Connection	USB [Device 0]		-		
Target interface	JTAG	Address 0 1 2 3 4 5 6 7 ASCI			
Init.ITAG sneed	5 kHz	8000000 30 44 00 20 01 01 00 08 0D			
JTAG speed	Auto recognition	8000008 05 01 00 08 07 01 00 08			
TAP number	<not used=""></not>	8000010 09 01 00 08 0B 01 00 08			
IR len	<not used=""></not>				
	-				
MCU	ST STM32F103ZE =				
Check core Id	Ver	8000028 00 00 00 00 11 01 00 08			
Core Id	0x3BA00477	L Flach ARM VA 10f			
Use target RAM	Yes				
RAM address	0x20000000				
RAM size	64 KB				
	071400510 511	Target erased, programmed and verified successfully -			
Flash memory	STM32FTUXXE interi-	Completed after 30.166 sec			
Size	512 KB				
Flash Id	0x0				
OLIVI, GLUE IA	NI		-1		
		ок	-		
LOG			Ś		
- Chip unsecu	red successfully				
 Erasing affects Erasing sect 	ed sectors tor 0 1 2 3 4 5 6 7 8 9	10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44			
- Erase opera	tion completed successfu				
- Programming t	arget (239804 bytes, 1 rar	ige)			
- Verifying CBC	- Varifet programmed successfully				
- CRC of affe	- CFC of affected sectors verified successfully (CRC = 0x783C80EB)				
- Target erased,	- Target erased, programmed and verified successfully - Completed after 30.166 sec				
Ready		Connected Core Id: 0x3BA00477 Speed: 1000 kH	_		
			11,		

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

H-JTAGの特性:

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

ディ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。

H-JTAG	Server
1	Vnable to find target. Please make sure that the hardware is properly connected and powered up. 通道正
″確定″ボ	タンをクリックすると、初の画面が出て来る。

H H-JTAG Server		
File Operations Flas	her Script Settings	Options Help
	UNKNOWN 0xfffffff	
Ready		

H-JTAG $\mathcal{O} \not\prec = \exists \neg$: Setting \rightarrow LPT Jtag Setting



Jtag Selection	User Defined Pin Assignment
Wiggler (Predefined)	TMS Pin4 D2 💌
 Sdt Jtag (Predefined) User Defined 	TCK Pin2 D0 -
Reset Signal Output	TDI Pin3 D1 -
nTRST output inverted nSRST output inverted	TDO Pin11 Busy 💌
	nTRST NO TAP RST
TCK Speed : MAX / 1 -	nSRST NO SYS RST

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。



H-JTAG Server File Operations Flasher Script Settings Options Help 🐞 USB/LPT Selection ¥ 🔍 X F 👍 USB Jtag Setting 🤚 LPT Jtag Setting 🜏 LPT Port Setting 🍗 Target Setting 🚯 Target Manager 📴 TAP Configuration CORTEX-INIS 0x3BA00477 TCK=Max/1 LPT TAP configuration

下記画面の通りに設定する。

TAP Configuration
TDI >> BYPASS >> ARM >> BYPASS >> TDO
TAP Num: 1 1 0 1
IReg Len: 5 + Auto + 0 +
Default OK Cancel

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



H-JTAG Serve	er 🔳 🗖 🔀
File Operations	Flasher Script Settings Options Help
* 0 V	F Start H-Flasher
	Auto Download
CORTEX-M3 0x3BA00477	
Enable auto flash dov	vnload TCK=Max/1 LPT

メニュー「Flasher」→「Start H-Flasher」を選択する。



STM32F103ZE を選択する。


	H-Flasher			
Ne	ew Load Save Save/	As Options	Exit About	
	Program Wizard 👘	>> Progra	mming - STM32F103XB	
	1 Flash Selection	Flash:	Unchecked	Check
	Configuration	larget:	Unchecked	
	Init Scripts	Туре:	Intel Hex Format	Program
	🦞 Perm Options	Dst Addr:		Verify
	5 Programming	Cur Film		
	H-Flasher Help	Src File:	C:≠cygwin+asp≠CSUN_STM32FTU3+asp.ne ▼	
		From	Entire Chip	Erase
		То:	Entire Chip	Blank
		Address:	Size:] Read

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。



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?を入力してエント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(<u>http://www.keil.com/</u>)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。

Velcome to Keil IÉVision3 Release 11/2007	An AR	M [®] Company
This SETUP program installs:		
RealView Microcontroller Development Kit V3.15b		
This SETUP program may be used to update a previous product installation. However, you should make a backup copy before proceeding.		
It is recommended that you exit all Windows programs before continuing with SETUP.		
Follow the instructions to complete the product installation.		
Keil µVision3 Setup		
KK Back	Next>>	Cancel

「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。

icense Agreement Please read the following license agreement carefully.		An A	ARM [®] Compa
To continue with SETUP, you must accept the terms of the Lice check box below.	nse Agreement. To acce	ept the agreement, c	click the
End-User License Agreement for AR Tools	M Kell Softwa	re Developr	nent 🔳
THIS END USER LICENCE AGREEMENT ("LICENCE")	IS A LEGAL AGREEN	MENT BETWEEN	/OU
(EITHER A SINGLE INDIVIDUAL, OR SINGLE LEGAL I THE USE OF THE SOFTWARE ACCOMPANYING THIS LICENSE THE SOFTWARE TO YOU ON CONDITION THIS THIS LICENCE. BY CLICKING "I AGREE" OR BY INST	ENTITY) AND ARM L LICENCE. ARM IS O HAT YOU ACCEPT A FALLING OR OTHER)	NLY WILLING TO LL OF THE TERMS WISE USING OR	
(EITHER A SINGLE INDIVIDUAL, OR SINGLE LEGAL I THE USE OF THE SOFTWARE ACCOMPANYING THIS LICENSE THE SOFTWARE TO YOU ON CONDITION TH THIS LICENCE. BY CLICKING "I AGREE" OR BY INST I agree to all the terms of the preceding License Agreement	ENTITY) AND ARM L LICENCE. ARM IS O HAT YOU ACCEPT A TALLING OR OTHER	NLY WILLING TO LL OF THE TERM: WISE USING OR	S IN



EIL ^M 1 [®] Company
vse
Cancel

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。

Customer Information Please enter your in	n formation.	An ARM® Company
Please enter your na	me, the name of the company for whom you work and	l your E-mail address.
First Name:	csun	
Last Name:	Jup	
Company Name:	Josun	
Company Name: E-mail: Keil ptVision3 Setup	info@csun.co.jp	

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。



Setup RealView Microcontroller Development Kit V3	.15b		
Setup Status		An	
µVision Setup is performing the requested operations.			
Install Files			
Installing STR71x.s.			
— Keil µVision3 Setup	Kack (Next >>	Cancel

インストール中の画面です。

ested operations successfully			
tly used project list.			
1	ntly used project list.	ntly used project list.	ntly used project list.

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。





ADC - µVision4	
Eile Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
- 1012日間にので「今日のない」のは「「日日」ので「日日」] 🔍 🎮 🔍 🖢 🔿 🖓 (
🔄 🖉 🎬 🥔 📇 🙀 ADC 🛛 🖃 🛣 🔛	
Project 🔹 🔍 🗸	
B-ADC	
Open File	
Open List File	
🕀 🧰 🔛 Rebuild all target files	
Euild target F7	
Tr <u>a</u> nslate File	
표귓 Stop b <u>u</u> ild	
A <u>d</u> d Group	
Add <u>Files to Group</u>	
Remoye Item	
Build Outr	→ û ×
Show Include File Dependencies	*
	· · ·
Build Output	
Configure target options	Cortex-M3 J-LINK

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。

MADC - µVision4	
<u>File Edit View Project Flash Debug Peripherals Tools SVCS Window H</u> elp	
Project Device Target Output Listing User C/C++ Asm Linker Debug Utilities	
ADC Select Folder for Objects Name of Executable: ADC	
Create Executable: ¥obj¥ADC	
Debug Information	Create Batch File
Create HEX File	
Do Browse Information	
C Create Library: .¥obj¥ADC.LIB	
🖻 Project 🧹	
Build Output	→ # ×
	<u>^</u>
OK Cancel Defaults	Help
	•
Build Output 🛃 Find in Files	
	Cortex-M3 J-LINK

Create HEX Fileの所に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。



MADC - µVision4		
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>P</u> roject Fl <u>a</u>	sh <u>D</u> ebug Pe <u>r</u> ipherals <u>T</u> ools <u>S</u> VCS <u>W</u> indov	v <u>H</u> elp
i 🗋 💕 🖬 🎒 i X 🖻 🖪 i	りや ←⇒ や発発表 律律/// /	k 🖄 🚽 🔍 🔗 🖉
🗄 🔗 🍱 🎬 🧼 🖳 🙀 ADO	💌 🔊 📥 🔁	
Project	→ # ×	
E 🔁 ADC		
E GTM22510		
STM32F10X.s		
i i i i i i i i i i i i i i i i i i i		
🖻 🚖 libary		
E STM32F10XR	Ogtions for File 'STM32F10xR.LIB' Alt+F7	
readme.txt	Open File	
	Open <u>L</u> ist File	
	Open <u>M</u> ap File	
	<u>R</u> ebuild all target files	
🖻 Project 🎯 Books 🚯 Fu 🔛	Build target F7	
Build Output	Tr <u>a</u> nslate File	→ ↓ ×
	Stop b <u>u</u> ild	A
	A <u>d</u> d Group	
	Add <u>F</u> iles to Group	_
4	Remove File 'STM32F10xR.LIB'	
🔄 Build Output 🗔 Find in 📥	Manage <u>C</u> omponents	
Configure target options	Show Include File Dependencies	Cortex-M3 J-LINK

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。

Options for File 'STM32F10>	(R.LIB	×
Properties		
Path: G.¥01_wo File Type: Library file	ork¥embeded¥MDK4.01¥ARM¥RV31¥LIB¥	ST¥STM32F10xR.LIB
Size: 1759306 last change: Thu May	Bytes 07 14:47:22 2009	🔽 Always Build
Stop on Exit Code: Not speci	fied	_
Custom Arguments:		
Memory Assignment:		Select Modules
Code / Const:	<default></default>	cortexm3_macro.o
Zero Initialized Data:	<pre><default></default></pre>	stm32f10x_adc.o
Uther Data:		□ stm32f10x_bkp.o □ stm32f10x_can.o □ stm32f10x_can.o □ stm32f10x_carc.o
	OK Cancel	Defaults Help





ツールバーの「Rebuild all target files」を押すと、ビルドが開始する。

ADC - μVision4				
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>P</u> roject Fl <u>a</u> sh <u>D</u> ebug Per	ipherals <u>T</u> ools <u>S</u> VCS	<u>W</u> indow <u>H</u> elp	p	
i 🗋 😭 🖟 🕷 🖉 🕹 🕼 👘 🖓 i k	作 作 作 作 律	筆 //= //泉 🖄		💽 🗟 🥐 🔍 🔍 🔿 (
: 🔗 🍱 🎬 🧼 🔜 🙀 ADC	- 🔊 🛔 🗟			
Project 👻 🕂 🗙				
E ADC				
🖻 📇 Startup Code				
STM32F10x.s				
i ⊡ i main.c				
ibary				
STM32F10xR.LIB				
🖻 🖳 Document				
I readme.txt				
E Project Books () Function (), Templat				
Build Output				
compiling atm22f10y it a				* * ^
linking				^
Program Size: Code=3872 RO-data=28	8 RW-data=24 ZI	-data=576		
".\obj\ADC.axf" - 0 Error(s), 0 Wa	arning(s).			-
				4
🖅 Build Output 🛛 🙀 Find in Files				
				Cortex-M3 J-LINK

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」 \rightarrow 「New uVision Project…」を選択する。

M AD	с - µ	Vision4										
<u>F</u> ile	<u>E</u> dit	View	<u>P</u> roject	Fl <u>a</u> sh	<u>D</u> ebug	Pe <u>r</u> ipherals	<u>T</u> ools	<u>s</u> vcs	<u>W</u> indow	<u>H</u> elp		
1 🗋 🛛	<u>i</u>	9	Nev	v µ <u>V</u> isioı	n Project					2		- 🔊 🛝 🔊
1 1	🔬 🏦		Nev	v Multi-F	Project <u>W</u> o	orkspace						
Project	Project Open Project								▼ X			
	ADC		<u>C</u> lo:	se Projec	t					_		=
	. 📄 C	MSIS	<u>E</u> xp	ort						•		<u></u>
-	🚖 A	PP	Mar	nage) on ((م نہ	
	÷ [🖞 mair	Sele	ect Devic	e for Taro	et 'ADC'					010/ Ta:101	
	÷) stm	Ren	ove Iter	n					aPIU_	initotruct	ure;
	II •	ח ה ה	Kell	io <u>v</u> e nei						h10.63	≒ 7∔	

プロジェクトネームを入力して、保存する。

Create New Project	
	 ✓ ✓
整理 ▼ 新しいフォルダー	8≡ ▾ 🔞
(論 ライブラリ 4前)	* 更新日時 種類 検索条件に一致する項目はありません。
■ コンピューター	
	III
ファイル名(N): stm32_demo	•
ファイルの種類(T): Project Files (*.uvproj)	•
 フォルダーの非表示 	保存(S) キャンセル .::

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



ſ	🛛 stm	32_de	mo - μVision4	23
	<u>F</u> ile	<u>E</u> dit	<u>View Project Flash Debug Peripherals Tools SVCS Window H</u> elp	
	1 🗋 🛛	39		0
	0	1 🔛	Select Device for Target 'Target 1'	
			CPU	
	E Pro	ject 🔇	Vendor: STM:croelectronics Device: STM:32F1032E Toolsei: ARM Data base Description: Image: STM:32F103RE ARM 32bit Cortex-M3 Microcontroller, 72MHz, 512kB Riash, 64kB SRAM, * Image: STM:32F103RE ARM 32bit Cortex-M3 Microcontroller, 72MHz, 512kB Riash, 64kB SRAM, * Image: STM:32F103RE ARM 32bit Cortex-M3 Microcontroller, 72MHz, 512kB Riash, 64kB SRAM, * Image: STM:32F103RE ARM 32bit Cortex-M3 Microcontroller, 72MHz, 512kB Riash, 64kB SRAM, * Image: STM:32F103RE ARM 32bit Cortex-M3 Microcontroller, 72MHz, 512kB Riash, 64kB SRAM, * Image: STM:32F103RE ARM 32bit Cortex-M3 Microcontroller, 72MHz, 512kB Riash, 64kB SRAM, * Image: STM:32F103RE STM:32F103RE Image: STM:32F103VB STM:32F103VB Image: STM:32F103VC STM:32F103VC Image: STM:32F103VC STM:32F103VC <td< th=""><th></th></td<>	
	Build C	output		н ×
			OK Cancel Help	
	≮ ≣≓ Bui	ld Outr	ut 🗖 Eind in Eilar	•
	C Bui	iu Outp	Contrar M2 LINK	

「OK」ボタンをクリックすると下記画面が表示される。

🕅 stm32_demo - µVision4	
<u>File Edit View Project Flash Debug Peripherals Tools SVCS Window H</u> elp	
□■■■■	l 🕫 🔯 🕒
😟 🎱 🕮 🥔 👯 🛛 Target 1 💿 🔊 📩 📥 🗟	
Project 👻 🕂 🗙	
μVision X	
Copy STM32 Startup Code to Project Folder and Add File to	
V Project ?	
Project Rooks {	
Build Output はい(Y) いいえ(N)	▼ ₽ ×
	-
٠	*
	ULINK Cort

「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。



2 • stm32_demo • • • • · · · · · · · · · · · · · · ·								
しいフォルダー								
■ 名前	更新日時	種類	サイズ					
退 Startup	2010/04/15 20:32	ファイル フォル						
stm32_demo.plg	2010/04/15 20:25	PLG ファイル	1 KB					
🖻 stm32_demo.uvproj	2010/04/15 20:17	∜Vision4 Project	0 KB					
stm32_demo_Target 1.dep	2010/04/15 20:25	DEP ファイル	1 KB					

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。

🔣 stm32_demo - µVision	4	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>P</u> roject	Fl <u>a</u> sh <u>D</u> ebug Pe <u>r</u> ipherals <u>T</u> ools <u>S</u> VCS <u>W</u> indow <u>H</u> elp	
i 🗋 💕 🖟 🎒 i X 🖬 🕯	8 ウや ←→ を発発数 導達症版 墜	- 🔒 🏘 🎯
i 🔗 🕮 🎬 🧼 🗮 l 🙀 l	Target 1 💽 💉 📥 🚍	
Project	→ ‡ X	
🖃 🛅 Target 1		
Source Group 1	Options for Group 'Source Group 1' Alt+E7	
	Open File	
	Open List File	
	<u>R</u> ebuild all target files	
	<u>B</u> uild target F7	
Project Books {	Tr <u>a</u> nslate File	
	stop b <u>u</u> ild	— T ×
	Add Group	· · · · ·
	Add Files to Group 'Source Group 1'	
	Remove Group 'Source Group 1' and its Files	
L 🕹	Manage <u>C</u> omponents	
~	Show Include File Dependencies	
		+
Add Files to surrent Drain t Co) Cimulatian
Add Flies to current Project Gro	oup	Simulation



🔞 Add Files to Group 'Source Group 1'		X
ファイルの場所(D: 🌇 Startup	- ← 🗈 💣 💷 -	
☑ 名前	更新日時	種類
🔽 🗊 cortexm3_macro.s	2008/09/21 13:21	Asse
🔽 🗑 stm32f10x_startup.s	2009/04/05 13:29	Asse
۲ III		Þ
ファイル名(N): 『cortexm3_macro.s" "stm32f10x_startu	ip.s″ Add	
ファイルの種類(T): Asm Source file (*.s*; *.src; *.a*)	Close	,,,

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。

💟 sti	m32_dem	10 -	µVision4	
<u>F</u> ile	<u>E</u> dit <u>V</u> i	ew	Project Flash Debug Peripherals Tools SVCS Window Help	
	💕 🖟 🕯		メ 国 臨 ウ や ← → 色 魯 魯 魯 律 律 /// // // ● 📃 🔹 🕞	l 🖉 🎯 🖉
: 🕸	🔛 🛗 🤘) 	🖞 📫 Target 1 🔹 💉 📥 🚍	
Projec	ct		<u>▼ </u>	
	Target	Ň	Ogtions for Target 'Target 1' Alt+F7	
	🟦 (Open File	
	····· 🔡 9	5	Open List File	
			Open <u>M</u> ap File	
		**	Rebuild all target files	
			Build target F7	
			Tr <u>a</u> nslate File	
			Stop b <u>u</u> ild	
Pr	roject 🧠		A <u>d</u> d Group	
Build	Output		Add Files to Group	
			Remo <u>v</u> e Item	<u>^</u>
		*	Manage <u>C</u> omponents	
		~	Show Include File Dependencies	
4				
Confi	gure targe	opti	ions	Simulation

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。



Options for Target 'Target 1'	×
Device Target Output Listing User C/C++ Asm Linker Debug Utilities	
Select Folder for Objects Name of Executable: stm32_demo	
 ⑦ Create Executable: .¥stm32_demo ☑ Debug Information 	Create Batch File
 ✓ Create HEX File ✓ Browse Information 	
C Create Library: .¥stm32_demo.LIB	
OK Cancel Defaults	Help

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。

Options for	r Target 'Target 1'	A.	23
Device	🛪 🚽 Output Luce Luce Loca.	.l. lei lei liere l	
	Folder: 🔒 stm32_demo	▼ 🗧 🐴 🖬 ▼	
•	名前	更新日時	種類
	퉬 Startup	2010/04/15 20:32	ファ・
1	🔽 퉬 新しいフォルダー	2010/04/15 21:09	ファ・
	stm32_demo.plg	2010/04/15 20:25	PLG
n.	🗹 stm32_demo.uvproj	2010/04/15 20:17	∦Visi
0	stm32_demo_Target 1.de	p 2010/04/15 20:25	DEP

「obj」フォルダを作成して指定する。

🔛 Browse for F	older				x
Folder:	길 obj			∲ ⊞▼	
名前	*		更新日時		種類
8	検索条件に一	·致する項目はありま	5世ん₀		
•					F.
Path:	G:¥temp¥stm32¥s	tm32_demo¥obj¥			
				OK	

「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 する場合はディフォルトのUse Simulatorのままで良い。

Options for Target 'Target 1'	×			
Device Target Output Listing User C/C++ Asm Linker Debug Utilities				
C Use Simulator Settings	Cortex-M3 J-LINK			
Limit Speed to Real-Time				
✓ Load Application at Startup ✓ Run to main()	✓ Load Application at Startup			
Initialization File:				
Restore Debug Session Settings	Restore Debug Session Settings			
I Breakpoints I Toolbox	I Breakpoints I Toolbox			
Watchpoints & PA	Watchpoints			
I✓ Memory Display	Memory Display			
	Déve Dille - Deventer			
SARMCM3 DU	SARMCM3 DLL			
or a mono.bee				
Dialog DLL: Parameter:	Dialog DLL: Parameter:			
DARMSTM.DLL PSTM32F103VE	TARMSTM.DLL PSTM32F103VE			
OK Cancel Defaults Help				

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。

Options for Target 'Target 1'					
Device Target Output Listing User C/C++ Asm Linker Debug Utilities					
Configure Flash Menu Command					
Use Target Driver for Flash Programming					
Cortex-M3 J-LINK Settings Vpdate Target before Debugging					
Init File: Edit					
C Use External Tool for Flash Programming					
Command:					
Arguments:					
🗖 Run Independent					
OK Cancel Defaults Help					

「Setting」ボタンを押すと、次の画面が表示される。



jLink/jTrace Cortex-M Target Dri	iver Setup			×
Debug Trace Flash Download				
Download Function C Erase Full Chip C Erase Sectors C Do not Erase	 ✓ Program ✓ Verify ✓ Reset and Run 	RAM for Al	gorithm 20000000 Size: 0x0800	
Description	Device Type	Device Size	Address Range	_
		Start:	Size:	
	Add	Remove]	
	ОК	Cancel		Help

「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。

Link/jTrace Cortex-M Debug Trace Flast	1 Target Driver Setup	RAM for Algorith	m		25
	ase Full Chip 🔽 Program Add Flash Programming Algo	prithm		X	
Programming Algo	Description LPC17xx IAP 32kB Rash LPC17xx IAP 64kB Rash LPC17xx IAP 64kB Rash RC28F640J3x Dual Rash S29JL032H_TOP Rash S1032H_TOP Rash S1032H_TOP Rash S1032H_TOP Rash S1032H_TOP Rash S1032F10x Low-density Rash S1032F10x Low-density Rash S1032F10x Low-density Rash S1032F10x Rash Options TMPM330F1x 25kB Rash TMPM330F1x 512kB Rash TMPM330F1x 2048kB Rash TMPM360Fx 2048kB Rash Add	Device Type On-chip Flash On-chip Flash Ext. Flash 32-bit Ext. Flash 16-bit Ext. Flash 16-bit On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash	Device Size 32k 512k 64k 16M 4M 16M 4M 16k 512k 256k 8M 16 128k 256k 512k 26k 512k 2M		
	OK	Cancel		He	elp

「Add」ボタン押すと、次の画面になる。



jLink/jTrace Cortex-M Target Driver Setup					
Debug Trace Flash Download					
Download Function C Erase Full Chip Image: Program RAM for Algorithm Image: C Erase Sectors Image: Verify Start: Dx20000000 Size: Dx0800 Image: C Erase Sectors Start: Dx20000000 Size: Dx0800					
Programming Algorithm					
STM32F10x High-density Rash On-chip Flash 512k 08000000H - 0807FFFFH					
Start: [0x08000000 Size: [0x00080000					
Add Remove					
OK Cancel	lelp				

「Reset and Run」の所にチェックを入れて「OK」ボタンを押す。

.ink/jTrace Cortex-M Target Driver Setup				X
Debug Trace Flash Download				
Download Function Case Full Chip Erase Sectors C Do not Erase	Program Verify Reset and Run	Start:	lgonthm x20000000 Size: 0x0800	
Description	Device Type	Device Size	Address Range	
STM32F10x High-density Flash	On-chip Flash	512k	08000000H - 0807FFFFH	
		Start:	Size:	
	Add	Remove		
	OK	Cance		Help

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。



👿 stm32_demo - µVision4	- 0 - X	
Eile Edit View Project Flash Debug Peripherals Tools SVCS Window Help		
] @ 見 @ 太 印 逸 ウ や や ゆ ぬ 肉 (幸 幸 川 版 @	a 🔊 🍳 🛙	0
🖸 🖄 🎬 🥔 🚉 🛄 Target 1 🔹 🌋 🛔 🚍		
Project 🔍 🗸 🕸 X		
Control Contro Control Control Control Control Control Control Control Control Co		
Under List File		
Open <u>M</u> ap File		
Rebuild all target files		
🔛 Build target F7		
Tr <u>a</u> nslate File		
Stop build		
Add Group		
Add Files to Group	• #	
Remo <u>v</u> e Item		1
Anage Components		
Show Include File Dependencies		
4		*
Configure file extensions, books and environment	Cortex-M3	

必要に応じてグループフォルダを追加する。LIB、APP など。

(Components, Environment and Books					
	Project Components Folders/Extensions Books					
	Project Targets: 🖄 🗙 🛧 🗲 Groups:					
	Source Group New (Insert)					
	Set as Crimant Tarriet					
	OK Cancel Help	ī				

Components, Environment and Books				
Project Components Folders/Extensions Books				
Project Targets: Mr 🖈 🗲	Groups: ★★ ★ File: Source Group 1	s: 🗙 🗲 🗲		
Set as Current Target	1	Add Files		
	OK Cancel	Help		



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