V850E2/MN4
I2C バス制御応用編

R01AN1499JJ0100
Rev.1.00
2013.3.14

要旨

本アプリケーションノートでは、V850E2/MN4 の I2C バス制御のための設定方法、およびサンプルコードの動作概要や使用方法について説明します。

このサンプルコードでは、V850E2/MN4 のデュアル・コア製品を用いて、片方のコアを I2C マスタの機能に、もう一方を I2C スレーブの機能に割り当てます。

また、内蔵 RAM の一部 (以降、転送対象メモリとします。) を EEPROM に見立て、スレーブが受信するデータをこのメモリにライトし、このメモリからリードしたデータをスレーブがマスタに送信します。

本サンプルコードの主な仕様を以下に示します。

- I2C バスをシングル転送モードに設定します。
- デュアル・コア製品を使用、コア 1 をマスタ、コア 2 をスレーブとして通信を行います。
- スレーブに、転送対象メモリへのライト、リードの関数を用意します。
- EEPROM に見立てた転送対象メモリに対するアドレス指定は、16 ビットとします。
- マスタ側でリードしたデータが、ライトしたものと一致するかテストし、異常があれば LED に表示します。
- 異常の有無に関わらず、通信が終了後、別の LED を点灯します。
-

対象デバイス

V850E2/MN4 (デュアル・コア製品：μPD70F3515)

開発環境

評価ボード： TB ボード(QB-V850E2MN4DUAL-TB)

エミュレータ：E1 エミュレータ

開発環境：CubeSuite+

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

目次

1. 仕様

デュアル・コア製品の一方のプロセッサ・エレメント(PE) を I2C のマスタとします。
もう一方の PE を I2C のスレーブとし、EEPROM に見立てた転送対象メモリへのリード／ライトの処理を行います。

マスタが転送対象メモリの特定のアドレスに 6Byte のデータをライトし、さらに同一のアドレスから 6Byte のデータをリードして、両者が一致するか確認します。

1.1 I2C マスタ

プロセッサ・エレメント 1 (PE1)をマスタの処理に使用します。

- 関数を初期化やデータ送受信など基本的な I2C の通信を受け持つ層と、転送対象メモリにアクセスする層にファイル単位で分離します。
- 転送対象メモリへのアクセスとしては、リード、ライトの 2 関数を用意します。
  - i2c_memory_write(uint32_t d_addr, uint32_t addr, uint8_t *data_w, int32_t len)
    スレーブ・アドレス、転送対象メモリのアドレス、ライトするデータ(配列)へのポインタ、データ長(Byte 単位)を指定し、転送対象メモリにデータをライトします。
  - i2c_memory_read(uint32_t d_addr, uint32_t addr, uint8_t *data_w, int32_t len)
    スレーブ・アドレス、転送対象メモリのアドレス、リードするデータを受け取る内蔵 RAM 中のユーザ定義のバッファ(以降、ユーザ定義バッファ)へのポインタ、データ長(Byte 単位)を指定し、転送対象メモリからデータをリードしてユーザ定義バッファに格納します。

1.2 I2C スレーブ

プロセッサ・エレメント 2(PE2)をスレーブの処理に使用します。

- 初期化やデータ送受信など基本的な I2C の通信を受け持つ層と、転送対象メモリの処理を行う層に分離しています。
- 転送対象メモリの仕様は次の通りです。
  - 容量は 64Byte とします。
  - 次の順序でデータを通信します。
    1. 最初にスレーブ・デバイスを示す 7 ビットのスレーブ・アドレスを受信します。これで自分あての通信か識別します。
    2. 続いて 1 ビットの転送方向指定ビットを受信します。本アプリケーションノートでは EEPROM を模しているため、この時点では 0 を受信します。
    3. 次の 2 バイトに転送対象メモリのアドレスを 1 バイトずつに分けてデータとして受信します。これでリード／ライトするアドレスを指定します。
    4. ライトの場合は続けてデータが受信されるので、転送対象メモリの、最初に受信したアドレスにライトします。
    5. リードの場合は、データ受信がなく、代わりに 7 ビットのスレーブ・アドレスと、転送方向指定ビット“1”を受信することで、スレーブ送信であることが指定されます。
    6. このフラグを確認したら、ストップ・コンディションを受信するまで、マスタにデータを送信し続けます。(マスタからリードするデータのサイズは送信されません。)

1.3 通信仕様

本サンプルコードでの I2C の通信の仕様は次の通りです。

- シングル転送モード
- ストップ・コンディション検出時ステータス割り込み要求発生
- 9 クロック目の立ち下がりで割り込み要求発生
- アクノリッジ許可
- 高速転送モード(400kHz)
- デジタルフィルタ無し
- 初期状態でのスタート・コンディション許可
- 通信予約禁止

1.4 使用する周辺機能

表1.1 使用する周辺機能と用途

周辺機能	用途
I2C チャネル 0	I2C のマスタとして動作
I2C チャネル 3	I2C のスレーブとして動作、
LED1	P13_7 に接続し、ライト後、リードしたデータと一致しなければ点灯します。
LED2	P13_6 に接続し、通信が終了した時点で点灯します。 (何らかの異常により通信が終了しない場合があります。)

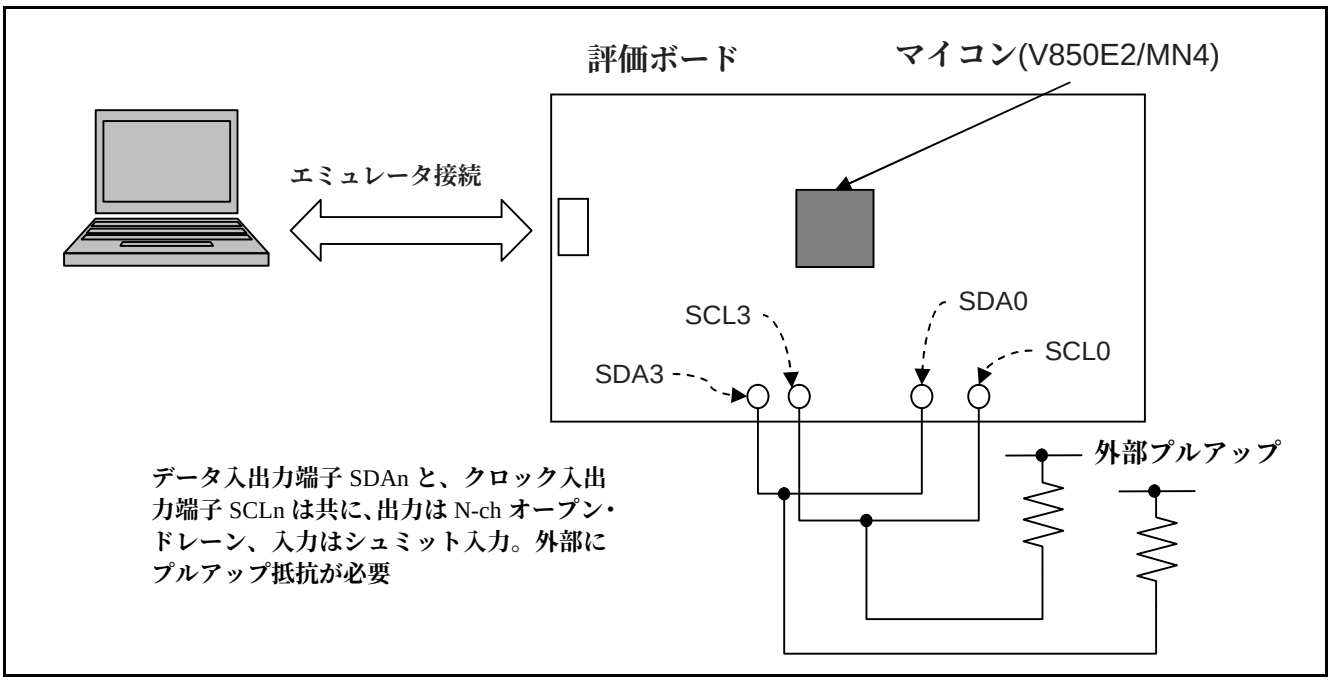

図1.1 使用例

2. 動作確認条件

本アプリケーションノートのサンプルコードは、下記の条件で動作を確認しています。

表2.1 動作確認条件

項目	内容
使用マイコン	V850E2/MN4 DualCore 版(μPD70F3515)
動作周波数	200MHz(発振 10MHz×PLL 20 逓倍)
動作電圧	3.3V
統合開発環境	CubeSuite+ V1.00
C コンパイラ	CX V1.20(CubeSuite+)、最適化：デフォルト
動作モード	通常動作モード
サンプルコードのバージョン	V1.00
CPU ボード	TB ボード(QB-V850E2MN4DUAL-TB)
エミュレータ	E1
使用ツール	なし

3. ハードウェア説明

3.1 ハードウェア構成例

データ入出力端子 SDAn と、クロック入出力端子 SCLn は共に、出力は N-ch オープン・ドレーン、入力はシュミット入力、外部にプルアップ抵抗が必要です。

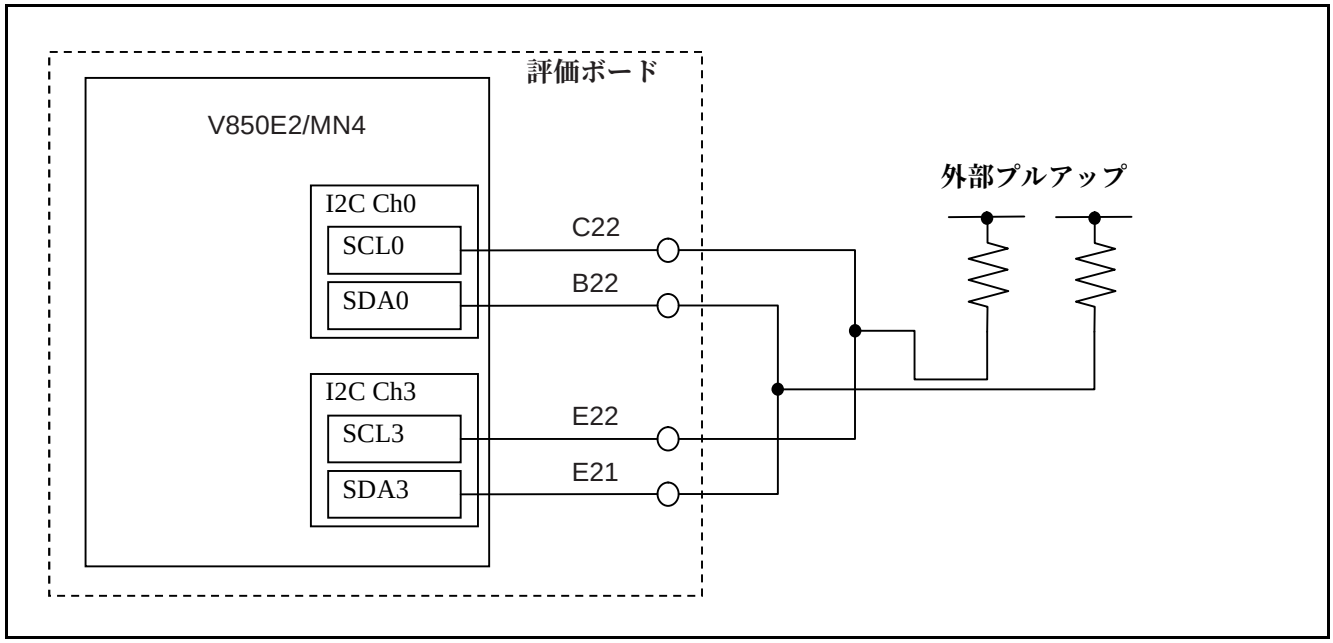

図3.1 ハードウェア構成

3.2 使用端子一覧

表 3.1に使用端子と機能(I2C) を示します。

表3.1 使用端子と機能(I2C)

端子名	入出力	内容
SCL0	出力	I2C チャネル 0 のクロック入出力端子
SDA0	入出力	I2C チャネル 0 のデータ入出力端子
SCL3	入力	I2C チャネル 3 のクロック入出力端子
SDA3	入出力	I2C チャネル 3 のデータ入出力端子

4. ソフトウェア説明

4.1 動作概要

本サンプルコードの動作概要を図 4.1に示します。

PE1 では、I2C のチャネル 0 をマスタ動作させるための初期化を行い、スレーブとして動作させる I2C のチャネル 3 に対しデータを送信し、転送対象メモリにライトします。
さらに転送対象メモリからリードし、両データが一致するか確認します。

PE2 では、I2C のチャネル 3 をスレーブ動作させるための初期化を行い、転送対象メモリとしての初期化も行います。
その後、マスタ側からの通信を待ち、受信した転送方向指定ビットに基づき、データライト、リードを行います。

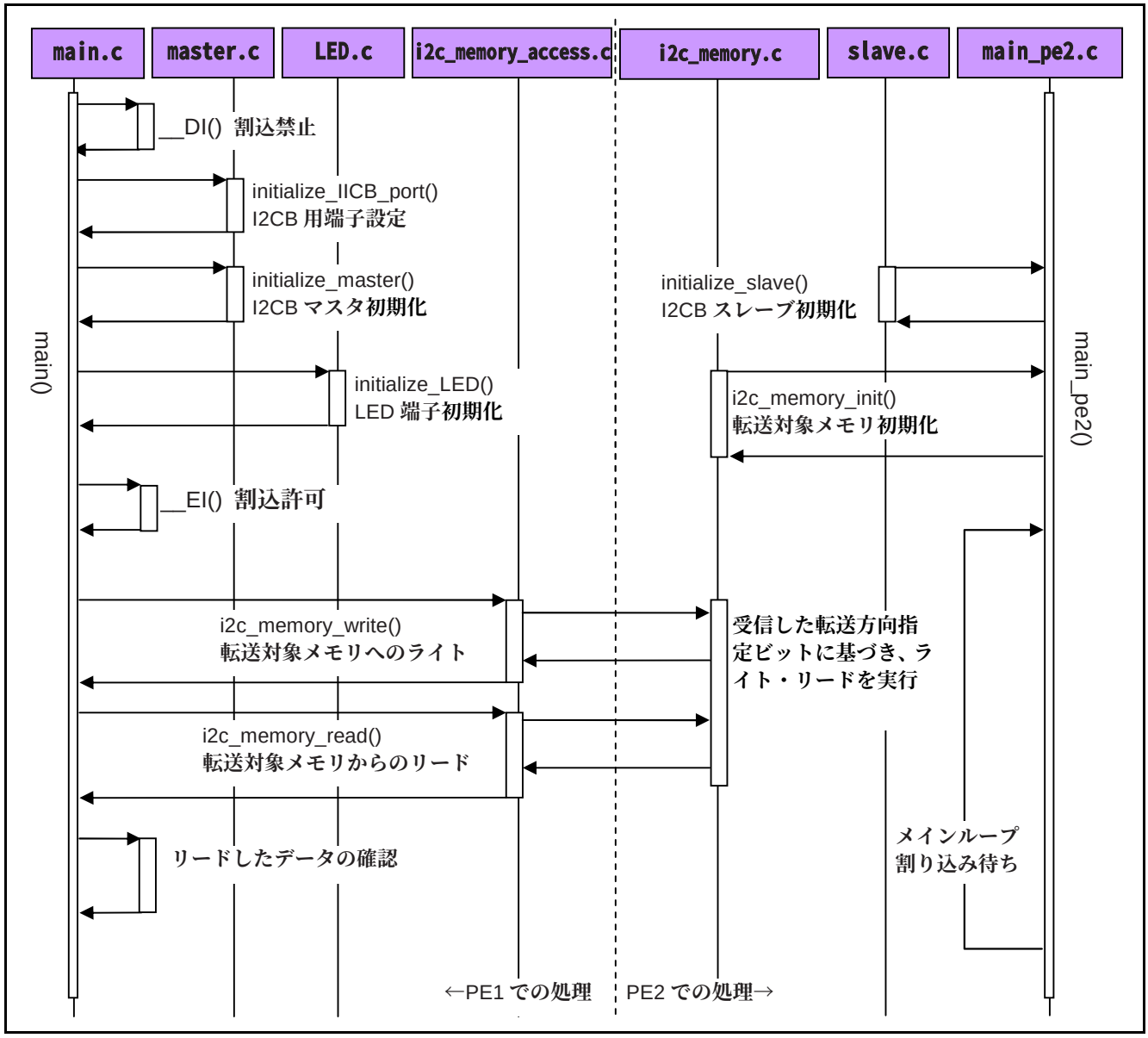

図4.1 サンプルプログラムの動作フロー

4.1.1 転送対象メモリへのライト処理

マスタ側より、スレーブのアドレス、転送対象メモリのアドレスの上位 8bit、下位 8bit を送信後、データ本体を送信します。
スレーブ側は、スタート・コンディションより、受信した順番でスレーブのアドレス、転送対象メモリのアドレスの上位 8bit、下位 8bit、データ本体の区別をしてそれぞれの処理を実行します。

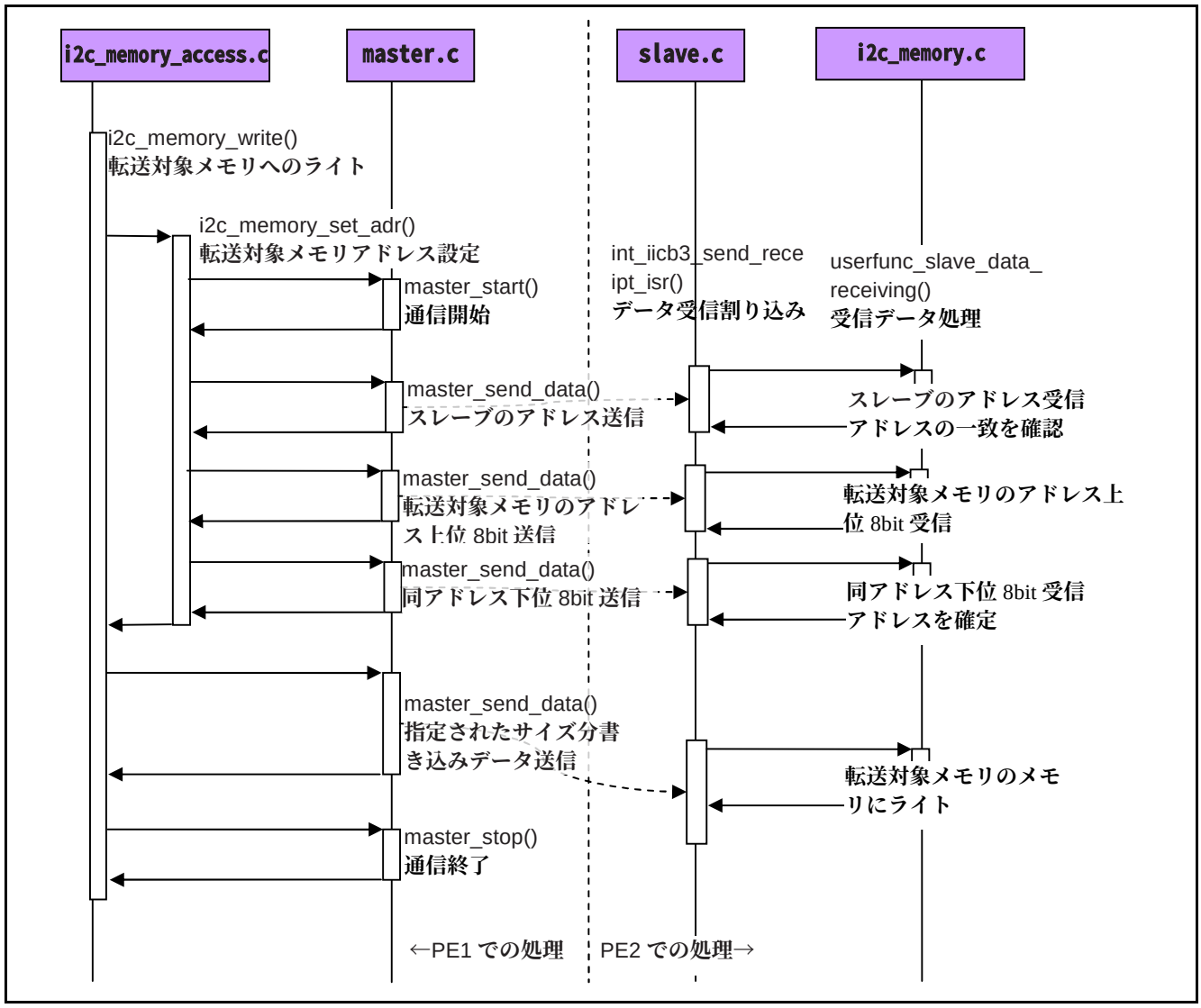

図4.2 転送対象メモリへのライトフロー

4.1.2 転送対象メモリからのリード処理

マスタ側は、スレーブ・アドレス、転送対象メモリのアドレスの上位 8bit、下位 8bit を送信します。次にスレーブ・アドレスに続き、転送方向指定ビットに“1”を送信し、スレーブ側からのデータ送信の割り込みを待ちます。この割り込みで、リードしたデータを受信します。受信したデータの個数が、i2c_memory_read()の引数で指定した数に達した時、ストップ・コンディションを発生して通信を終了します。

スレーブ側は、前半のデータ通信で転送対象メモリのアドレスを設定します。後半にスレーブ・アドレスの後に、転送方向指定ビットに“1”を受信すると、リードとみなし、転送対象メモリ中のデータを、マスタ側が発生したストップ・コンディションを受信するまで送信し続けます。

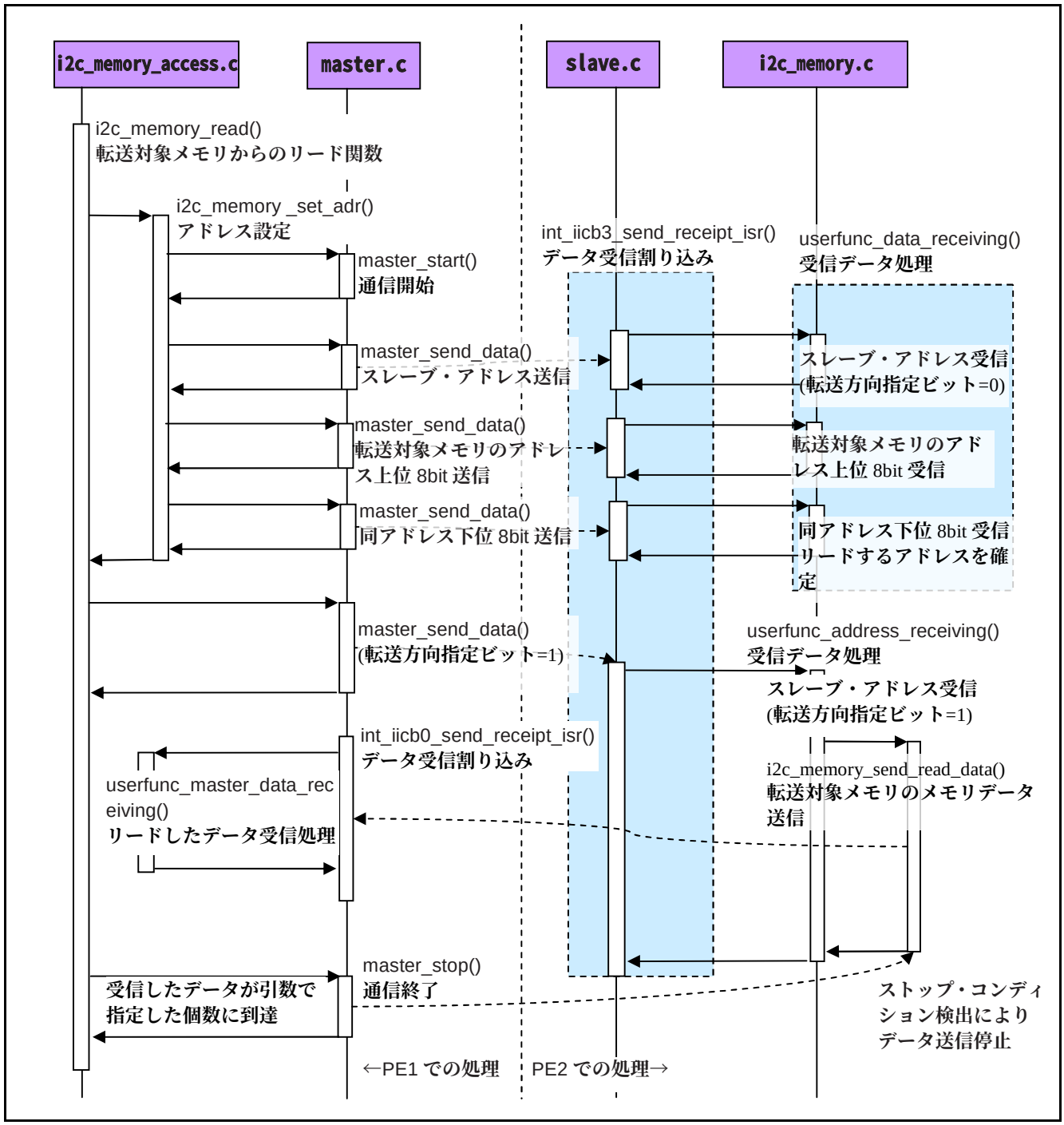

図4.3 転送対象メモリからのリードのフロー

4.2 必要メモリサイズ

表 4.1に必要メモリサイズを示します。(CubeSuite+、最適化オプション＝デフォルトで測定)

表4.1 **必要メモリサイズ**

使用メモリ	サイズ	備考
ROM	7041	CubeSuite+の生成する map ファイルに出力された ROM 領域で使用するサイズ
RAM	692	CubeSuite+の生成する map ファイルに出力された RAM 領域で使用するサイズ
最大使用ユーザスタック	44	CubeSuite+のスタック見積もりツールで算出
最大使用割り込みスタック	212	同上

【注】 必要メモリサイズは C コンパイラのバージョンやコンパイルオプションにより異なります。

4.3 ファイル構成

表 4.2にサンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成されるファイルは除きます。

表4.2 サンプルコードで使用するファイル

ファイル名	概要	備考
cstartM.asm	ハードウェア初期化処理	PE1、PE2 共通部
stackPE1.asm	スタックサイズ設定	PE1 用
stackPE2.asm		同 PE2 用
V850E2MN4 Dual.dir	リンク・ディレクティブ・ファイル	デュアル・コア製品専用
iic.h	マクロ・変数・関数宣言	
r_typedefs.h	標準型再定義	
i2c_memory.h	転送対象メモリ用マクロ・変数・関数宣言	
main.c	メイン処理	PE1 用
main_pe2.c	メイン処理	PE2 用
master.c	I2C マスタ側の各種関数	PE1 のみ利用する関数
slave.c	I2C スレーブ側の各種関数	PE2 のみ利用する関数
i2c_memory_access.c	転送対象メモリアクセス関数	データのライト、リードアクセスする転送対象メモリのアドレス設定、データ受信時の処理
i2c_memory.c	転送対象メモリサービス関数	転送対象メモリの機能を受け持ち、ライト・リードの要求に応じた動作
LED.c	LED の処理、初期化、点灯などの制御	

4.4 オプション設定メモリ

本サンプルコードでは、オプション・バイトの設定は行っていません。必要に応じて設定してください。

4.5 定数一覧

表 4.3にサンプルコードで使用する定数を示します。

表4.3 サンプルコードで使用する定数

定数名	設定値	内容
const uint8_t g_write_test_data[]	{2, 3, 5, 7, 11, 13}	転送対象メモリにライトするテストデータ
const uint8_t g_write_test_data_length	sizeof(g_write_test_data) / sizeof(uint8_t)	上記テストデータ中のデータ数

4.6 列挙子一覧

図 4.4にサンプルコードで使用する列挙子を示します。

列挙子名		内容	使用関数
enum i2c_memory_slave_mode		I2C スレーブが現在受信中のデータの種類を示す。	userfunc_slave_data_receiving()
	I2C_MEMORY_DEVICE_ADDR =0,	スレーブ・アドレス	
	I2C_MEMORY_UPPER_ADDR,	転送対象メモリのアドレス上位 8 ビット	
	I2C_MEMORY_LOWER_ADDR,	転送対象メモリのアドレス下位 8 ビット	
	I2C_MEMORY_DATA,	ライト用データ	

図4.4 サンプルコードで使用する列挙子

4.7 変数一覧

表 4.4にグローバル変数を示します。

表4.4 グローバル変数

型	変数名	内容	使用関数
uint8_t	g_read_test_data[I2C_MEMORY_MAX_ADDR]	転送対象メモリからリードしたデータを受けるユーザ定義バッファ	main()
int32_t	g_iicb0_comm_flag	I2CB0 (マスタ)でデータ受信割り込みが発生したことを示すフラグ	master_wait_interrupt() master_clear_flag() int_iicb0_send_receipt_isr()
int32_t	g_iicb0_comm_status_flag	I2CB0 でステータス割り込みが発生したことを示すフラグ	master_clear_flag() int_iicb0_status_isr()
int32_t	g_master_reception_flag	I2CB0 がデータを受け取ったことを示すフラグ	clear_master_reception_flag() wait_master_reception() int_iicb0_send_receipt_isr()
int32_t	g_iicb3_comm_flag	I2CB3 (スレーブ)でデータ受信割り込みが発生したことを示すフラグ	slave_clear_flag() int_iicb3_send_receipt_isr()
int32_t	g_iicb3_comm_status_flag	I2CB3 でステータス割り込みが発生したことを示すフラグ	slave_clear_flag() int_iicb3_status_isr()
uint32_t	g_received_data_counter ;	クライアント(転送対象メモリを利用する側、I2C マスタ)がリードの際、転送対象メモリから受け取ったデータの数	i2c_memory_read() userfunc_master_data_receiving()
uint8_t *	g_p_read_buffer	転送対象メモリからのリードの際、I2C マスタがデータを格納するためのユーザ定義バッファへのポインタ	i2c_memory_read() userfunc_master_data_receiving()
uint8_t	g_i2c_memory_buffer[I2C_MEMORY_MAX_ADDR]	転送対象メモリのデータを保持	i2c_memory_init() userfunc_slave_data_receiving() i2c_memory_send_read_data()
(enum 型)	g_i2c_memory_read_write	転送対象メモリへのライト命令かリード命令かを示すフラグ	i2c_memory_clear_status() userfunc_slave_data_receiving() userfunc_address_receiving()
(enum 型)	g_i2c_memory_slave_mode	転送対象メモリの内部状態、受け取ったデータが転送対象メモリのアドレスか、ライトするデータかを示すフラグ	i2c_memory_clear_status() userfunc_slave_data_receiving()
uint16_t	g_i2c_memory_address	クライアントが指定した、転送対象メモリのアドレスを保持する変数	i2c_memory_clear_status() userfunc_slave_data_receiving() i2c_memory_send_read_data()

表4.5 const 型変数

型	変数名	内容	使用関数
uint8_t	g_write_test_data[]	転送対象メモリにライトするテストデータ	void main(void)
uint8_t	g_write_test_data_length	上記テストデータ中のデータ数	void main(void)

4.8 関数一覧

表 4.6、表 4.7 に関数を示します。

表4.6 関数(1/2)

関数名	概要
void main(void)	PE1 のメインルーチン、各初期化処理関数を呼び出したあと、永久ループに入ります。
void wait10ms(void)	転送対象メモリへのライト後などで、ウェイトをかけます。
void main_pe2(void)	PE2 のメインルーチン、転送対象メモリ初期化処理関数を呼び出したあと、永久ループに入り、I2C の割り込みを待ちます。
void initialize_IICB_port(void)	I2C の端子設定を行います。
void initialize_master(void)	I2C マスタの設定を行います。
void master_start(void)	I2C 通信を開始します。
void master_send_data(uint8_t data)	I2C マスタからスレーブに 8 ビットデータを送信します。
void master_wait_interrupt(void)	I2C マスタのデータ送信後、I2C 送受信割り込みを待ちます。
void master_wait_acknowledge(void)	I2C マスタのデータ送信後、アクノリッジ信号を待ちます。
void master_stop_waiting(void)	I2C マスタのウェイト状態を解除します。
void master_clear_flag(void)	I2C マスタの割り込み、アクノリッジ信号待ちのフラグをクリアします。
void master_stop(void)	ストップ・コンディション発行、通信を終了します。
void clear_master_reception_flag(void)	転送対象メモリからのデータリードなどで用いる、I2C マスタ側データ受信フラグをクリアします。
void wait_master_reception(void)	I2C マスタでデータを受信するのを待ちます。
void int_iicb0_send_receipt_isr(void)	I2C マスタデータ送受信割り込み関数。データ受信時に、転送対象メモリから受信したデータを処理する関数を呼びます。
void int_iicb0_status_isr(void)	I2C マスタエラー受信割り込み関数、エラーフラグ iicb0_communication_status_flag をセットします。
void initialize_slave(void)	I2C スレーブの設定を行います。
void slave_send_data(uint8_t data)	I2C スレーブから I2C マスタにデータを送信します。
void slave_stop_waiting(void)	I2C スレーブのウェイト状態を解除します。
void slave_clear_flag(void)	I2C スレーブの割り込みフラグをセットします。
uint32_t slave_stop_condition(void)	I2C スレーブがストップ・コンディション(通信終了を示す信号)を受信したかを示します。
void int_iicb3_send_receipt_isr(void)	I2C スレーブ送受信割り込み関数。データ受信時に、転送対象メモリとしての処理を行う関数を呼びます。
void int_iicb3_status_isr(void)	I2C スレーブエラー受信割り込み関数、エラーフラグ iicb3_communication_status_flag をセットします。

表4.7 関数(2/2)

void i2c_memory_set_adr(uint32_t d_addr, uint32_t addr)	転送対象メモリにアクセスするアドレスを第 2 引数で設定します。 ※第一引数はスレーブ・アドレス。
void i2c_memory_write(uint32_t d_addr, uint32_t addr, uint8_t *data_w, int32_t len)	転送対象メモリにデータをライトします。
void i2c_memory_read(uint32_t d_addr, uint32_t addr, uint8_t *data_w, int32_t len)	転送対象メモリからデータをリードします。
void userfunc_master_data_receiving(uint8_t data)	I2C マスタがデータを受け取った時の処理、I2C マスタデータ送受信割り込み関数から呼ばれ、受け取ったデータを i2c_memory_read() で指定したユーザ定義バッファに蓄積します。
void i2c_memory_clear_status(void)	転送対象メモリに関する各種ステータス変数を初期化します。
void i2c_memory_init(void)	転送対象メモリを初期化します。
void userfunc_slave_data_receiving(uint8_t data)	I2C スレーブがデータを受信した時の処理、受信時のモードにより、転送対象メモリのアドレス設定、データライトなどの処理を行います。
void userfunc_slave_status(void)	I2C スレーブがエラーを受信したときの処理。
uint32_t i2c_memory_send_read_data(void)	I2C マスタに対し、リード要求に応じてデータを送信します。
void userfunc_address_receiving(uint8_t data)	I2C スレーブ・アドレス受信時の処理。スレーブ・アドレスとともに送信される Read/Write ビットにより、それぞれの処理を行います。
void initialize_port_LED(void)	LED 端子設定関数
void initialize_LED(void)	LED 関連の初期化関数
void LED_off(void)	全 LED 消灯
void disp_LED1(void)	TB ボード上 LED1 を点灯、I2C 転送エラーを示します。
void disp_LED2(void)	TB ボード上 LED2 を点灯、通信終了を示します。

4.9 関数仕様

サンプルコードの関数仕様を示します。

main()		
概　要	PE1 用メイン関数	
ヘッダ	ー	
宣　言	void main(void)	
説　明	各初期化処理関数を呼び出し、I2C のマスタとスレーブを設定します。 その後、クライアントとして転送対象メモリにライト後、リードを転送対象メモリの同一のアドレスに対して行います。 両データが一致するかテストし、結果に応じて LED を点灯します。	
引　数	ー	ー
リターン値	ー	

main_pe2()		
概　要	PE2 用メイン関数	
ヘッダ	ー	
宣　言	void main_pe2(void)	
説　明	転送対象メモリの設定を行い、クライアントからの要求による割り込みを待ちます。	
引　数	ー	ー
リターン値	ー	

wait10ms()		
概　要	10msec 待つ	
ヘッダ	iic.h	
宣　言	void wait10ms(void)	
説　明	転送対象メモリにライト後、処理待ちのため 10msec 待ちます。	
引　数	ー	ー
リターン値	ー	

initialize_IICB_port()		
概　要	IICB 用の端子設定	
ヘッダ	ー	
宣　言	void initialize_IICB_port(void)	
説　明	IICB が使用する端子機能の設定を行います。 (P4_13:SDA3, P4_11:SCL3, P4_6:SDA3, P4_7:SCL3)	
引　数	ー	ー
リターン値	ー	

initialize_master()		
概　要	I2C マスタの設定	
ヘッダ	iic.h	
宣　言	void initialize_master(void)	
説　明	シングル転送モードで、ストップ・コンディション検出時ステータス割り込み信号発生し、アクノリッジ信号も発生するよう、I2C マスタ(IICB0)を設定します。	
引　数	ー	ー
リターン値	ー	

master_start()		
概　要	スタート・コンディション発生	
ヘッダ	iic.h	
宣　言	void master_start(void)	
説　明	IICB0 からスタート・コンディション発生し、通信を開始します。 以降、IICB0 はマスタとして機能します。	
引　数	ー	ー
リターン値	ー	

master_send_data()		
概　要	マスタからデータ送信	
ヘッダ	iic.h	
宣　言	void master_send_data(uint8_t data)	
説　明	IICB0 からスレーブ側に 8 ビットデータを送信します。 送信後、送受信割り込みとアクノリッジ信号を待ちます。	
引　数	uint8_t data	送信するデータ
リターン値	ー	

master_wait_interrupt()		
概　要	送受信割り込み待ち	
ヘッダ	iic.h	
宣　言	void master_wait_interrupt(void)	
説　明	IICB0 が送受信割り込み(IICBTIAn)を受け取るのを待ちます。	
引　数	ー	ー
リターン値	ー	

master_wait_acknowledge()		
概　要	アクノリッジ信号待ち	
ヘッダ	iic.h	
宣　言	void master_wait_acknowledge(void)	
説　明	IICB0 がアクノリッジ信号を受け取るのを待ちます。	
引　数	ー	ー
リターン値	ー	

master_stop_waiting()		
概　要	ウェイト状態解除	
ヘッダ	iic.h	
宣　言	void master_stop_waiting(void)	
説　明	データ送受信後、I2C は自動的にウェイト状態になります。そのウェイト状態を解除します。	
引　数	ー	ー
リターン値	ー	

master_clear_flag()	
概　要	割り込みフラグ解除
ヘッダ	iic.h
宣　言	void master_clear_flag(void)
説　明	データ送受信割り込み、アクノリッジ受信の有無を示すフラグをクリアします。
引　数	ー　　ー
リターン値	ー

master_stop()	
概　要	ストップ・コンディション発行
ヘッダ	iic.h
宣　言	void master_stop(void)
説　明	ストップ・コンディションを発行し、通信を終了します。
引　数	ー　　ー
リターン値	ー

clear_master_reception_flag()	
概　要	データ受信フラグ解除
ヘッダ	iic.h
宣　言	void clear_master_reception_flag(void)
説　明	転送対象メモリからのデータリードなどで用いる、I2C マスタ側データ受信フラグをクリアします。
引　数	ー　　ー
リターン値	ー

wait_master_reception()	
概　要	マスタのデータ受信待ち
ヘッダ	iic.h
宣　言	void wait_master_reception (void)
説　明	スレーブからデータ送信後、マスタが受信するのを待ちます。
引　数	ー　　ー
リターン値	ー

int_iicb0_send_receipt_isr()	
概　要	送受信割り込み処理
ヘッダ	ー
宣　言	__interrupt void int_iicb0_send_receipt_isr(void)
説　明	I2C マスタデータ送受信割り込み関数。データ受信時に、転送対象メモリから受信したデータを処理する関数を呼びます。
引　数	ー　　ー
リターン値	ー

int_iicb0_status_isr()		
概　要	**送受信割り込み処理**	
ヘッダ	**－**	
宣　言	__interrupt void int_iicb0_status_isr(void)	
説　明	I2C **マスタエラー受信割り込み関数、エラーフラグ** iicb0_communication_status_flag **をセットします。**	
引　数	**－**	**－**
リターン値	**－**	

initialize_slave()		
概　要	I2C **スレーブ初期化**	
ヘッダ	iic.h	
宣　言	void initialize_slave(void)	
説　明	I2C **スレーブの設定を行います。**	
引　数	**－**	**－**
リターン値	**－**	

slave_send_data()		
概　要	I2C **スレーブからのデータ送信**	
ヘッダ	iic.h	
宣　言	void slave_send_data(uint8_t data)	
説　明	I2C **スレーブから** I2C **マスタにデータを送信します。**	
引　数	uint8_t data	**送信するデータ**
リターン値	**－**	

slave_stop_waiting()		
概　要	**ウェイト状態解除。**	
ヘッダ	iic.h	
宣　言	void slave_stop_waiting(void)	
説　明	**データ送信後、自動的にウェイト状態になりますが、その状態を解除します。**	
引　数	**－**	**－**
リターン値	**－**	

slave_clear_flag()		
概　要	**スレーブフラグクリア**	
ヘッダ	iic.h	
宣　言	void slave_clear_flag(void)	
説　明	I2C **スレーブの割り込みフラグをクリアします。**	
引　数	**－**	**－**
リターン値	**－**	

slave_stop_condition()

概　要	**ストップ・コンディション受信の有無**	
ヘッダ	iic.h	
宣　言	uint32_t slave_stop_condition(void)	
説　明	I2C **スレーブがストップ・コンディション(通信終了を示す信号)を受信したかを示します。**	
引　数	**－**	**－**
リターン値	0：**受信していない、非** 0：**受信した**	

int_iicb3_send_receipt_isr()

概　要	**データ送受信割り込み**	
ヘッダ	iic.h	
宣　言	void int_iicb3_send_receipt_isr(void)	
説　明	I2C **スレーブ送受信割り込み関数。データ受信時に、転送対象メモリとしての処理を行う関数を呼びます。**	
引　数	**－**	**－**
リターン値	**－**	

int_iicb3_status_isr()

概　要	**エラー受信割り込み**	
ヘッダ	iic.h	
宣　言	void int_iicb0_status_isr(void)	
説　明	I2C **スレーブエラー受信割り込み関数、エラーフラグ** iicb3_communication_status_flag **をセットします。**	
引　数	**－**	**－**
リターン値	**－**	

i2c_memory_set_adr()

概　要	**転送対象メモリのアドレス指定**	
ヘッダ	i2c_memory.h	
宣　言	void i2c_memory_set_adr(uint32_t d_addr, uint32_t addr)	
説　明	**転送対象メモリにアクセスするアドレスを第** 2 **引数で設定します。**	
引　数	uint32_t d_addr	**スレーブ・アドレス。本サンプルコードでは、**iic.h **のマクロ** SLAVE_DEVICE_ADDR (0xaa) **のみ指定します。**
	uint32_t addr	**転送対象メモリのアドレス**
リターン値	**－**	

i2c_memory_write()		
概　要	**転送対象メモリにデータライト**	
ヘッダ	i2c_memory.h	
宣　言	void i2c_memory_write(uint32_t d_addr, uint32_t addr, uint8_t *data_w, int32_t len)	
説　明	**転送対象メモリにデータをライトします。**	
引　数	uint32_t d_addr	**スレーブ・アドレス**
	uint32_t addr	**転送対象メモリのアドレス**
	uint8_t *data_w	**ライトするデータの配列へのポインタ**
	int32_t len	**ライトするデータの長さ**(Byte **単位**)
リターン値	**－**	

i2c_memory_read()		
概　要	**転送対象メモリからデータリード**	
ヘッダ	i2c_memory.h	
宣　言	void i2c_memory_read(uint32_t d_addr, uint32_t addr, uint8_t *data_w, int32_t len)	
説　明	**転送対象メモリからデータをリードします。**	
引　数	uint32_t d_addr	**スレーブ・アドレス**
	uint32_t addr	**転送対象メモリのアドレス**
	uint8_t *data_w	**リードしたデータを受ける配列へのポインタ**
	int32_t len	**リードするデータの長さ**(Byte **単位**)
リターン値	**－**	

userfunc_master_data_receiving()		
概　要	**転送対象メモリからデータリード**	
ヘッダ	i2c_memory.h	
宣　言	void userfunc_master_data_receiving(uint8_t data)	
説　明	I2C **マスタがデータを受け取った時の処理。**I2C **マスタのデータ送受信割り込み関数から呼ばれ、受け取ったデータを** i2c_memory_read()**で指定した配列に蓄積します。**	
引　数	uint32_t data	**受信したデータ**
リターン値	**－**	

i2c_memory_clear_status()		
概　要	**転送対象メモリステータス初期化**	
ヘッダ	**－**	
宣　言	void i2c_memory_init(void)	
説　明	**転送対象メモリのステータスを初期化します。フラグ、アドレスのクリアを行います。**	
引　数	**－**	**－**
リターン値	**－**	

i2c_memory_init()		
概　要	**転送対象メモリ初期化**	
ヘッダ	**－**	
宣　言	void i2c_memory_init(void)	
説　明	**転送対象メモリを初期化します。フラグのクリア、内部データの初期化を行います。**	
引　数	**－**	**－**
リターン値	**－**	

userfunc_slave_data_receiving()		
概　要	データ受信時の処理	
ヘッダ	ー	
宣　言	void userfunc_slave_data_receiving(uint8_t data)	
説　明	I2C スレーブがデータを受信した時の処理、受信時のモードにより、転送対象メモリのアドレス設定、データライトなどの処理を行います。	
引　数	uint32_t data	受信したデータ
リターン値	ー	

userfunc_slave_status()		
概　要	ステータス割り込みの処理	
ヘッダ	ー	
宣　言	void userfunc_slave_status(void)	
説　明	ステータス割り込み発生時に転送対象メモリに関する処理を行います。	
引　数	ー	ー
リターン値	ー	

i2c_memory_send_read_data()		
概　要	マスタ側にデータ送信	
ヘッダ	ー	
宣　言	uint32_t i2c_memory_send_read_data(void)	
説　明	転送対象メモリからリードしたデータをマスタ側に送信します。	
引　数	ー	ー
リターン値	送信したデータ数(Byte 単位)	

userfunc_address_receiving()		
概　要	アドレス受信時の処理	
ヘッダ	ー	
宣　言	void userfunc_address_receiving(uint8_t data)	
説　明	受信したスレーブ・アドレスに転送方向指定ビットがセットされていたら、転送対象メモリのリード処理を行います。	
引　数	uint8_t data	受信したスレーブ・アドレス(7 ビット)+転送方向指定ビット(1 ビット)
リターン値	ー	

initialize_port_LED()		
概　要	LED の端子設定関数	
ヘッダ	ー	
宣　言	void initialize_port_LED(void)	
説　明	PortConfigulation[7:12]() を呼び出して、LED で使用する端子機能を選択します。	
引　数	ー	ー
リターン値	ー	

initialize_LED()

概　要	LED 関連の初期化関数
ヘッダ	―
宣　言	void initialize_hbus(void)
説　明	前出の初期化関数を呼び、LED を利用可能にします。
引　数	― ―
リターン値	―

LED_off()

概　要	全 LED 消灯
ヘッダ	―
宣　言	void LED_off(void)
説　明	すべての LED を消灯します。
引　数	― ―
リターン値	―

disp_LED1()

概　要	LED1 点灯
ヘッダ	―
宣　言	void disp_LED1(void)
説　明	TB ボード上 LED1 を点灯、I2C 転送エラーを示します。
引　数	― ―
リターン値	―

disp_LED2()

概　要	LED2 点灯
ヘッダ	―
宣　言	void disp_LED2(void)
説　明	TB ボード上 LED2 を点灯、通信終了を示します。
引　数	― ―
リターン値	―

4.10 フローチャート

4.10.1 PE1 のメイン処理

図 4.5にPE1 のメイン処理のフローチャートを示します。各種設定を行った後、転送対象メモリにライトを行い、同一の転送対象メモリのアドレスからリードを行い、ライトしたデータと一致するか検証します。検証の結果に応じて LED を点灯します。

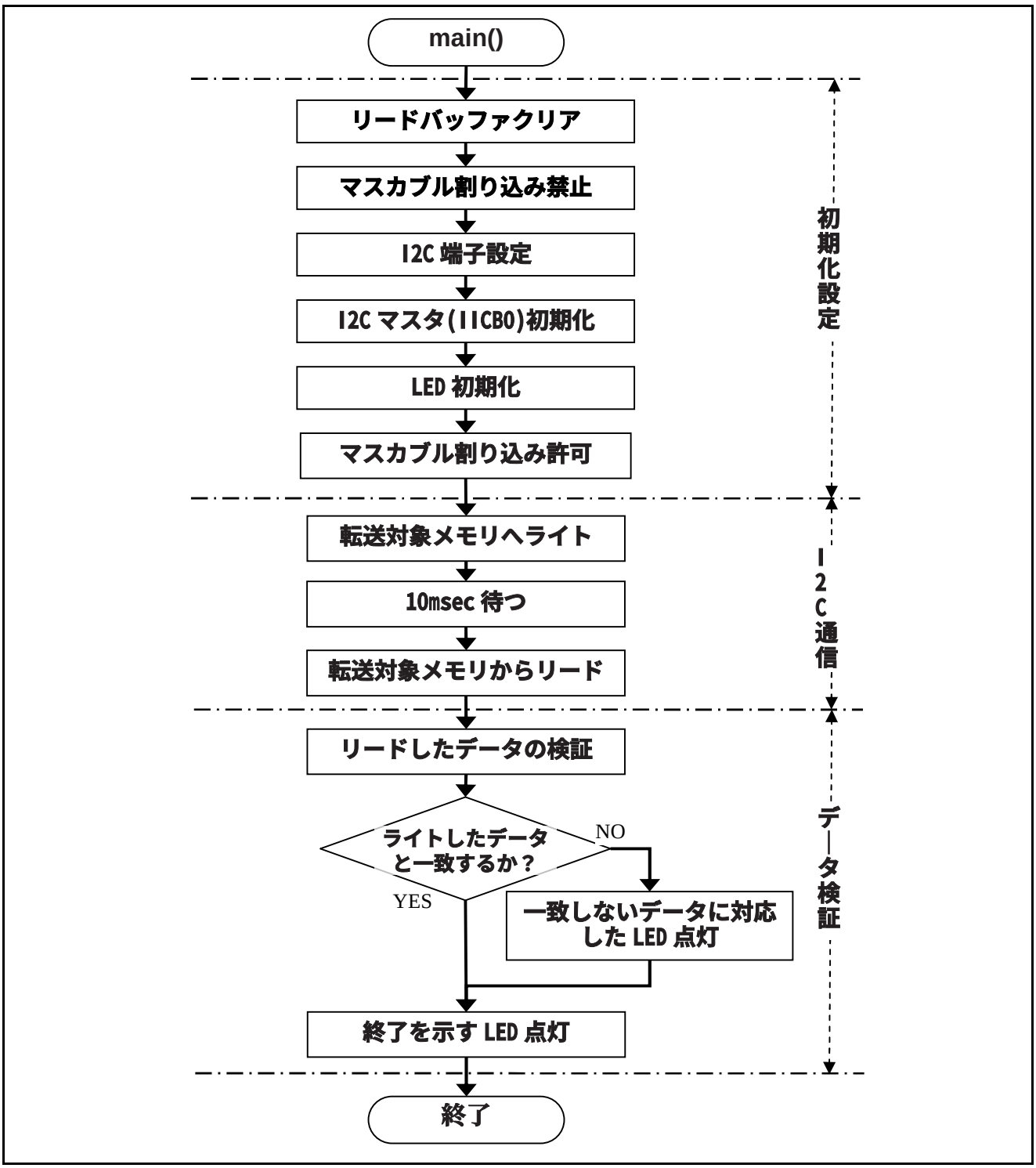

図4.5 PE1 のメイン処理

4.10.2 PE2 のメイン処理

図 4.6にPE2 のメイン処理のフローチャートを示します。
I2C や転送対象メモリの初期化を行った後、割り込みウェイト状態に入ります。転送対象メモリへのリード／ライトすべて割り込みで処理されます。

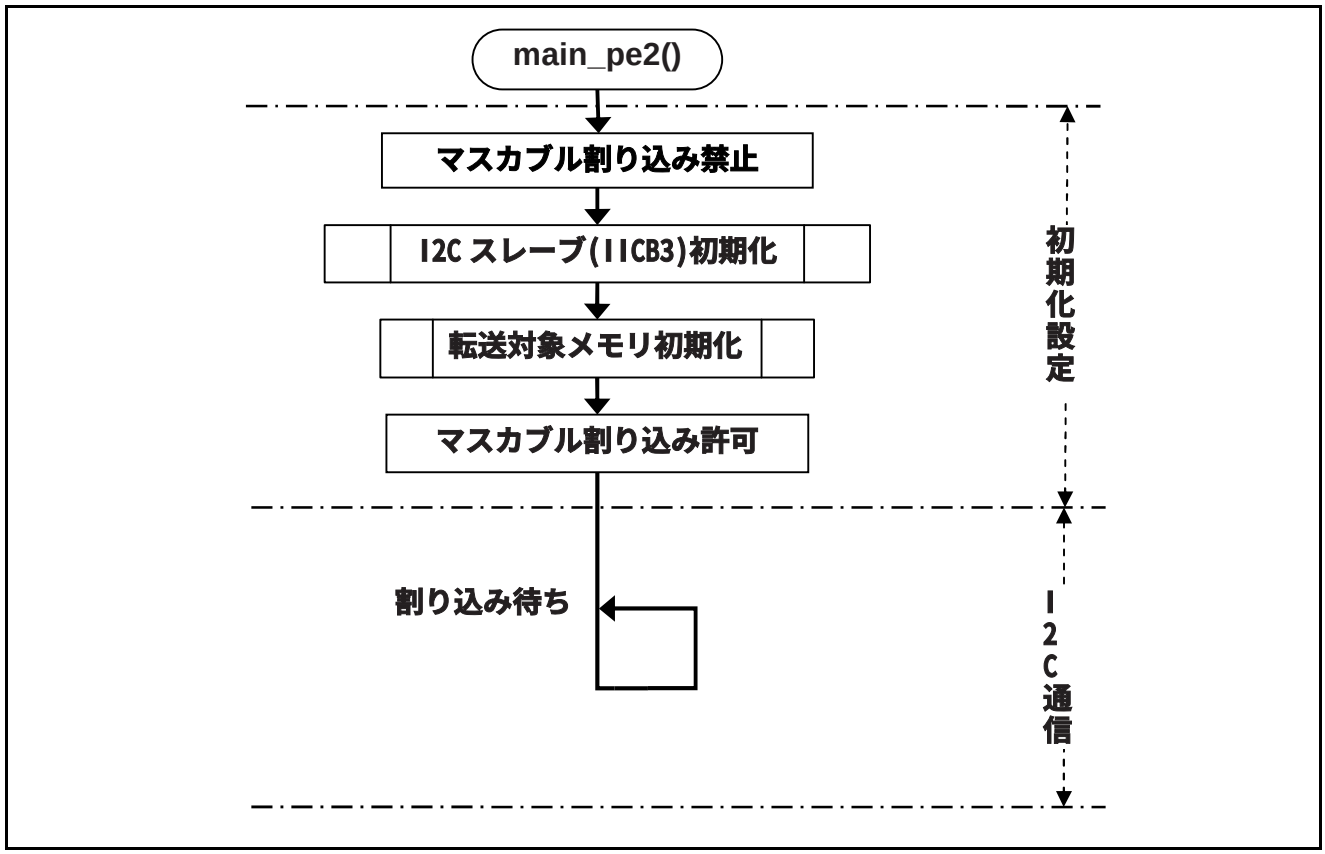

図4.6 PE2 のメイン処理

5. サンプルコード

サンプルコードは、ルネサス エレクトロニクスホームページから入手してください。

6. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

V850E2/MN4 ユーザーズマニュアル ハードウェア編（R01UH0262JJ）
（最新版をルネサス エレクトロニクスホームページから入手してください。）

テクニカルアップデート／テクニカルニュース
（最新の情報をルネサス エレクトロニクスホームページから入手してください。）

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com

お問合せ先
http://japan.renesas.com/inquiry

改訂記録	V850E2/ML4 アプリケーションノート I2C 制御編

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2013.03.14	—	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレスのアクセス禁止
   【注意】リザーブアドレスのアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレスがあります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、事前に問題ないことをご確認下さい。
   同じグループのマイコンでも型名が違うと、内部メモリ、レイアウトパターンの相違などにより、特性が異なる場合があります。型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、
   各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、
   家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通用信号機器、
   防災・防犯装置、各種安全装置等
   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 本資料に記載されている当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。
注2. 本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/