TJBN007

TWI コントローラ

TOPPERS/JSP for Blackfin プロジェクト
最終更新: 2012/12/2 Rev 1.0

1. はじめに

TOPPERS/JSP for Blackfin は、Blackfin の I2c(TWI)ペリフェラルを制御するマスター・コントローラを提供しています。このコントローラは、制御 API、割り込みハンドラ、イニシャライザ、コンフィギュレータからなります。API はスレッドセーフになっており、複数のタスクが同時にアクセスしても安全に I2C デバイスにアクセスできます。

1.1. ツールバージョンなど

開発に使用したソフトウェアやハードウェアは以下のとおりです

- ホスト OS : Ubuntu 12.04 LTS 32bit (Windows 7 64bit 上の VMware Workstation 8 で検証)
- GNU Toolchain : Blackfin 2012R1 RC4
- TOPPERS/JSP カーネル 1.4.3、Blackfin 依存部 3.3.1
- JTAG ICE : gnICE+
- ACB-BF592
- PCF8570

PCF8570 は、プロトコルの試験に使用した I2C RAM です。

1.2. 対応プロセッサ

2012 年 9 月の時点で、動作を確認しているプロセッサは ADSP-BF592 だけです。他のプロセッサへの対応については、後述します。

2. 使い方

TWI コントローラは、TOPPERS/JSP と共に使うことを前提に開発されています。システムへの組み込みは簡単です。

2.1. アプリケーションに組み込む

アプリケーションへの組み込みは、コンフィグレータで行います。コントローラのコンフィグレータ・ファイル i2c0_m.cfg をアプリケーションのコンフィグレータ・ファイルからインクルードします。これでイニシャライザと割り込みハンドラの登録が完了し、アプリケーションから使えるようなります。

ソースファイル i2c_subsystem.c と、インクルードファイル i2c_subsystem.h をどこに置くかは任意です。システム依存部においても構いませんが、jsp/systask に置くほうがきれいでしょう。

ビルドを正しく行うためには、システム依存部の Makefile.config の KERNEL_OBJ 変数に、i2c_subsystem.o を追加してください。

```
  KERNEL_COBJS := $(KERNEL_COBJS) chip_config.o uart.o
chip_debugboot.o chip_dump.o i2c_subsystem.o
```

2.2. API 呼び出し

API は3つしかありません。いずれも関数です。関数呼び出し前に必要な準備はすべてイニシャライザが行なっているため、アプリケーションからなにかする必要はありません。

API 関数は以下の3つです。

```
  int i2c_master_write( int peripheral, int slave, unsigned char
write_data[], int write_count );


  int i2c_master_read( int peripheral, int slave, unsigned char
read_data[], int read_count);


  int i2c_master_write_read( int peripheral, int slave, unsigned char
write_data[], int write_count, unsigned char read_data[], int
read_count );
```

write および read 関数は write_read 関数のサブセットです。

peripheral 引数は、プロセッサ内蔵の TWI ペリフェラルの番号を表します。0 から始まる整数です。ほとんどの Blackfin プロセッサは TWI をひとつしか持っていません。この場合、peripheral 引数は常に 0 です。ADSP-BF548 のように TWI を 2 つ持つデバイスでは、TWI0 を使うには 0 を、TWI1 を使うには 1 を引数として与えてください。

slave 引数は I2C スレーブ・デバイスの I2C アドレスです。このアドレスは 7bit 整数として渡してください。

write_data[]および read_data[]は、NULL ではないポインタです。write_data[]に送信するデータの配列を格納して渡します。read_data[]には転送終了後の受信データが格納されます。

write_count および read_count はそれぞれ送信データバイト数と受信データバイト数を示します。

返り値はエラーステータスです。0 ならばエラーではありません。ERROR ステータスは、I2C_ERROR_XXXX ステータスと、TWI ペリフェラルの割り込みステータスのビット論理輪になっています。

2.3. 移植のヒント

以下ではこのコントローラを ADSP-BF592 以外の Blackfin デバイス用に移植するための情報を提供します。

2.3.1. インクルード・ファイル

i2c_subsystem.c は、ADSP-BF592 用のインクルードファイルを読み込んでいます。この読み込みをプロセッサにあわせて変えてください。具体的には、以下の部分を変更します。

```
#ifdef _COMMON_BF592
#include <cdefBF592-A.h>
#else
#error "This processor is not supported"
#endif
```

ifdef 節の後ろに、elif defined()節を追加して他のプロセッサに対応します。なお、参照するマクロは、TOPPERS/JSP for Blackfin のチップ依存部で宣言されている __COMMON_BFXXX を使います。例えば、BF537 であれば、_COMMON_BF537 です。

2.3.2. レジスタの設定方法

対応するプロセッサの TWI ペリフェラルが BF592 互換であれば、ほぼそのままビルドできるはずです。ビルドエラーが起きるときには、イニシャライザ関数の中だと思われますので、twi_control[]配列の初期化部のレジスタ名を対応プロセッサ用に変更してください。

イニシャライザ以外はレジスタ名に依存しない実装になっています。

2.3.3. 2つ以上の TWI がある場合

2つ以上の TWI がある場合、I2CNUM マクロの数を TWI ペリフェラルの数にあわせてください。デフォルトでは 1 で、これにより twi_control[]配列の長さが1に制限されています。

イニシャライザの中では、TWI のペリフェラルの数分だけ twi_control[]に対して初期化を行います。2つ以上の TWI がある場合にはここを書き換えてください。

3. リファレンス

3.1. i2c_master_write()

この関数は peripheral で指定する TWI ポートから、slave で指定するアドレスを持つ I2C デバイスに対して書き込みを行います。

```
int i2c_master_write(
                int peripheral,
                int slave,
                unsigned char write_data[],
                int write_count
        );
```

peripheral は、TWI ポートの番号です。TWI0 なら 0 を、TWI1 なら 1 を指定します。TWI しかない場合には 0 を指定します。

slave は、7bit の i2C スレーブアドレスです。

write_data は書き込むデータの配列で、write_count は書き込むバイト数です。書きこみバイト数は 1 より大きな値にしてください。

実行はブロッキング状態で行われます。すなわち、書き込みが終わるまでこの API は呼び出し側に制御を戻しません。問題なく終了すれば返り値は 0 です。問題が発生した場合には、TWI ポートの割り込みステータスレジスタの値と、API 固有のエラー値のビット論理和が返されます。

また、実行は排他的に行われ、複数のタスクが同時に同じ TWI ポートをアクセスしようとしても、順番に実行されます。

3.2. i2c_master_read()

この関数は peripheral で指定する TWI ポートから、slave で指定するアドレスを持つ I2C デバイスより読み出しを行います。

```
int i2c_master_read(
                int peripheral,
                int slave,
                unsigned char read_data[],
                int read_count
        );
```

peripheral は、TWI ポートの番号です。TWI0 なら 0 を、TWI1 なら 1 を指定します。TWI しかない場合には 0 を指定します。

slave は、7bit の i2C スレーブアドレスです。

read_data は読みだしたデータを格納する配列で、read_count は読み出しバイト数です。読み出しバイト数は 1 より大きな値にしてください。

実行はブロッキング状態で行われます。すなわち、読み出しが終わるまでこの API は呼び出し側に制御を戻しません。問題なく終了すれば返り値は 0 です。問題が発生した場合には、TWI ポートの割り込みステータスレジスタの値と、API 固有のエラー値のビット論理和が返されます。

また、実行は排他的に行われ、複数のタスクが同時に同じ TWI ポートをアクセスしようとしても、順番に実行されます。

3.3. i2c_master_write_read()

この関数は peripheral で指定する TWI ポートから、slave で指定するアドレスを持つ I2C デバイスに対して書き込みを行います。書き込みが終了すると、直ちに I2C の Repeated Start 機能を使用して、同じアドレスから読み出しを行います。

```
int i2c_master_write_read(
              int peripheral,
              int slave,
              unsigned char write_data[],
              int write_count,
              unsigned char read_data[],
              int read_count
        );
```

peripheral は、TWI ポートの番号です。TWI0 なら 0 を、TWI1 なら 1 を指定します。TWI しかない場合には 0 を指定します。

slave は、7bit の i2C スレーブアドレスです。

write_data は書き込むデータの配列で、write_count は書き込むバイト数です。書きこみバイト数は 1 より大きな値にしてください。

read_data は読みだしたデータを格納する配列で、read_count は読み出しバイト数です。読み出しバイト数は 1 より大きな値にしてください。

実行はブロッキング状態で行われます。すなわち、書き込みが終わるまでこの API は呼び出し側に制御を戻しません。問題なく終了すれば返り値は 0 です。問題が発生した場合には、TWI ポートの割り込みステータスレジスタの値と、API 固有のエラー値のビット論理和が返されます。

また、実行は排他的に行われ、複数のタスクが同時に同じ TWI ポートをアクセスしようとしても、順番に実行されます。

3.4. エラーコード

以下のようなエラーコードが定義されています。

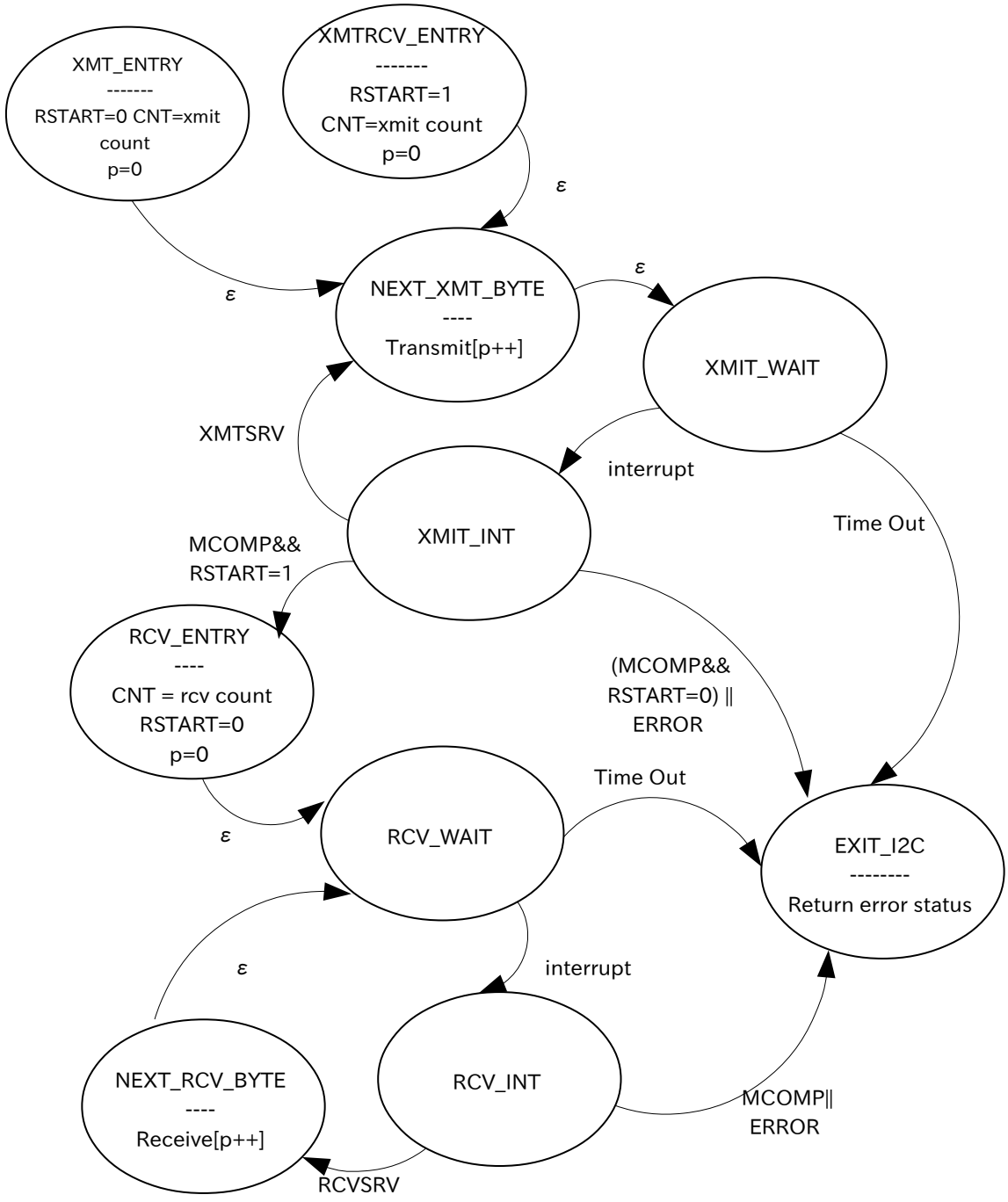
```
#define I2C_ERR_WRONGPARAM      0x4000
#define I2C_ERR_TOOLONGBUFFER   0x2000
#define I2C_ERR_TIMEOUT         0x1000
```

I2C_ERR_WRONGPARAM は、API の引数がおかしい時に返されます。例えば、間違ったポート番号を与えた時などにあたります。

I2C_ERR_WRONGPARAM は、write_count あるいは read_count 引数が大きいすぎるときに返されるエラーです。現在の実装では、254 バイトまでのデータしか転送できません。

I2C_ERR_TIMEOUT スレーブが応答しないなどの理由でタイムアウトした場合に返されるエラーです。スレーブアドレスが間違っている場合などに発生します。

4. 状態遷移図


XMT_ENTRY

RSTART=0 CNT=xmit count
p=0
XMTRCV_ENTRY

RSTART=1
CNT=xmit count
p=0
ε
ε
NEXT_XMT_BYTE

Transmit[p++]
ε
XMIT_WAIT
XMTSRV
interrupt
XMIT_INT
Time Out
MCOMP&&
RSTART=1
RCV_ENTRY

CNT = rcv count
RSTART=0
p=0
(MCOMP&&
RSTART=0) ||
ERROR
Time Out
ε
RCV_WAIT
EXIT_I2C

Return error status
ε
interrupt
NEXT_RCV_BYTE

Receive[p++]
RCV_INT
MCOMP||
ERROR
RCVSRV

5. 文献･履歴など

5.1. 参考文献

- ADSP-BF592 Hardware Reference Manual
- I2C バス仕様書

5.2. 履歴

- 2012/12/2 : Rev 1.0