お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 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。
5. 輸出に際しては、「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。本資料に記載されている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の目的で使用しないでください。また、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に使用することができません。
6. 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。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されてい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生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   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

   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、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社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。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、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、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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。
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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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。

32192/32195/32196 グループ

シリアルインタフェースを使用した I²C バス制御例

1. 要約

この資料は 32192/32195/32196 グループの周辺機能を使用した、I²C バス制御の参考プログラム例を掲載しています。

2. はじめに

この資料で説明する応用例は次のマイコン、条件での利用に適用されます。

・マイコン　：32192 グループ（M32192F8VFP、M32192F8UFP、M32192F8TFP、
　　M32192F8VWG、M32192F8UWG、M32192F8TWG）
　　32195 グループ（M32195F4VFP、M32195F4UFP、M32195F4TFP）
　　32196 グループ（M32196F8VFP、M32196F8UFP、M32196F8TFP）
・動作周波数　：128～160MHz（参考プログラムは 160MHz を想定して作成しています）
・動作ボード　：32192μT-Engine　R0P3219TR001MRK

3. 接続方法

3.1 接続図

マイコンとデバイスとの接続方法は図 3.1.1 で示します。

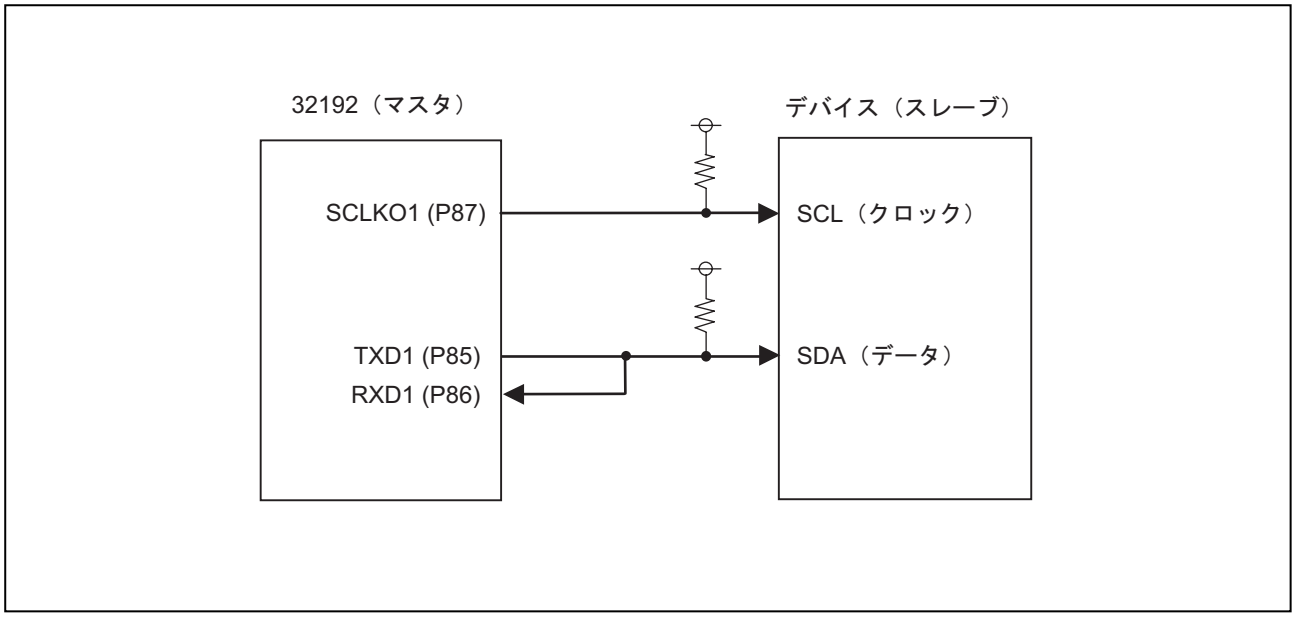

図 3.1.1 接続図

3.2 制御方法

本参考プログラム例では、CSIO モード（クロック同期形シリアルインタフェース）と汎用ポート出力を組み合わせて I²C バス制御波形を出力します。スタートコンディションとストップコンディション出力時は、マイコンの端子を汎用ポート出力として設定し、"H"→"L"、または"L"→"H"へ出力を変化させます。データバイト部分では、9 ビット CSIO モードの出力として、8 ビットのデータと、1 ビットの ACK または NAK を出力します。ACK 時は"L"を出力、NAK 時は"H"を出力します。

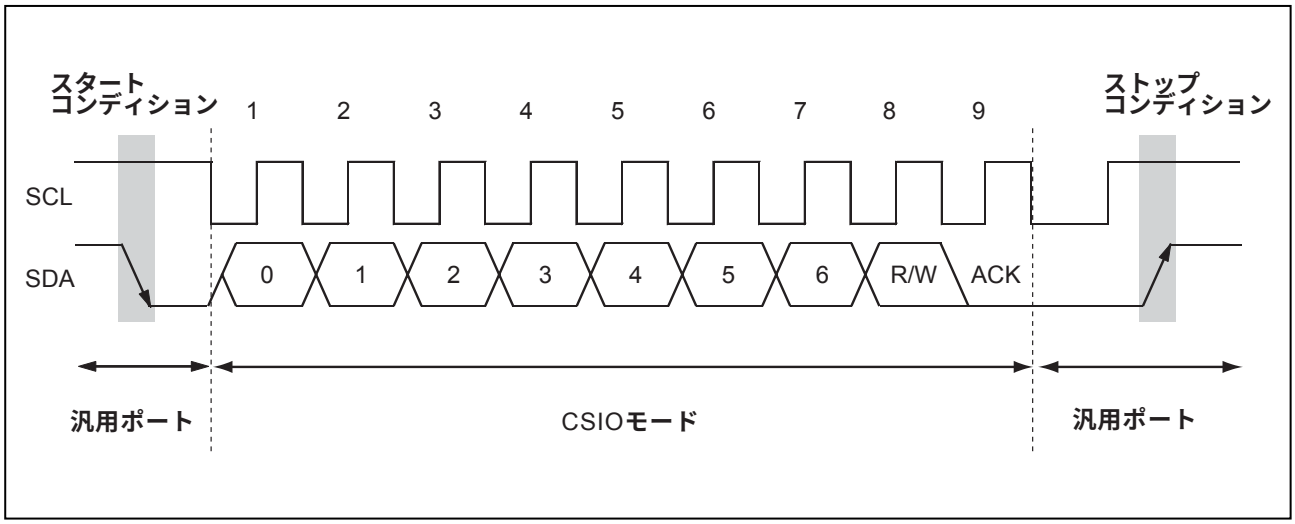

図 3.2.1 出力波形例

3.3 通信フォーマット

(1) マスタライト

Memory Address で指定したアドレスから複数バイトの Write Data を書き込みます。
指定バイトの Write Data 出力後、ストップコンディションを出力します。

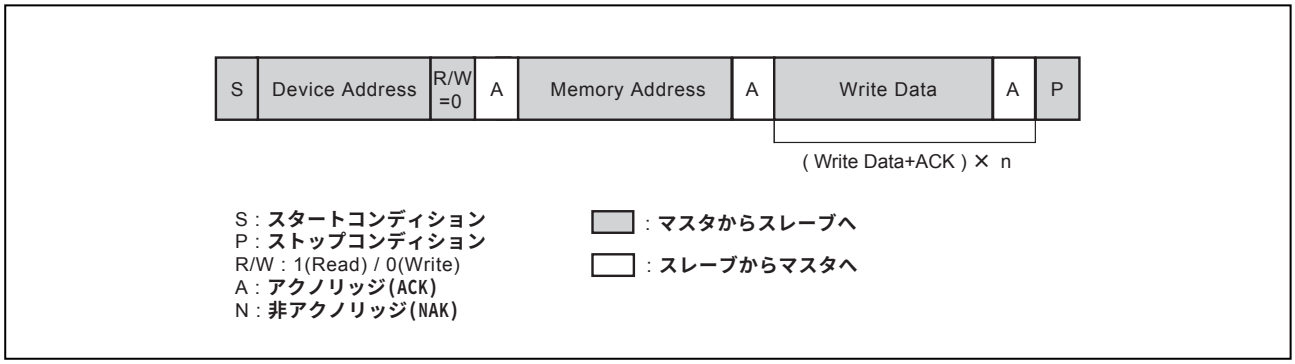

図 3.3.1 マスタライト

(2) マスタリード

Memory Address で指定したアドレスから Read Data を読み込みます。
Read Data 入力後、アクノリッジを出力することで複数バイトの Read Data を読み込むことができます。
指定バイトの Read Data を入力後、非アクノリッジを出力し、ストップコンディションを出力します。

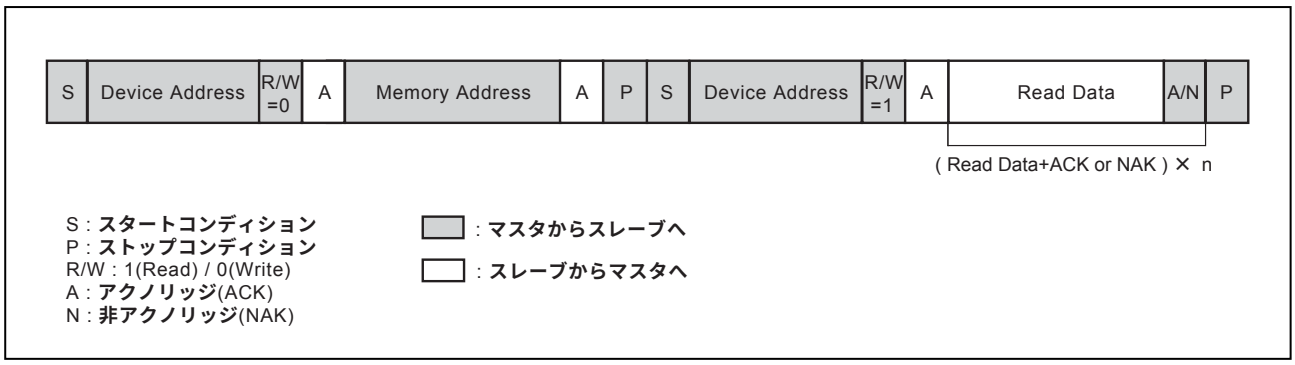

図 3.3.2 マスタリード

4. I2C バス制御参考プログラム

4.1 参考プログラムの概要

本参考プログラム例では、マスタ、スレーブの一対一通信を想定して作成しています。
SIO1 の CSIO モード（クロック同期形シリアルインタフェース）と汎用ポート出力を組み合わせて I2C バス制御波形を出力します。また、次の条件で動作させています。

- 通信条件として 9 ビット CSIO モード、内部クロック、MSB ファースト、100kbps を使用しています。
- ACK または NAK の判定処理は行っていません。
- 通信開始前のバス状態のチェック、アービトレーションに負けた場合の処理は行っていません。
- ポート動作モードレジスタの設定を汎用ポートからシリアルモードへの変更時、以前に出力したシリアルデータの最終ビットの状態により出力レベルが決定されます。そのため、初期化処理とストップコンディション出力時にダミーデータとして H'0000 を送信バッファに書き込んでいます。（ただしポート動作モードレジスタは汎用ポートに設定しているため、データは出力されません。）
- ストップコンディション出力処理では、ダミーデータの送信処理を行っているため、ダミーデータの送信分の時間がかかります。
- TML0 タイマを使用して、ウェイト時間長を調整します。

注. - 使用するデバイスにより、スタートコンディション、ストップコンディション出力時のウェイト時間の調整が必要です。
- ダミーデータの送信バッファへの書き込みは、送信許可ビットを送信許可に設定した後に行ってください。
- 送信許可ビットを送信禁止に設定した場合、再度ダミーデータ（H'0000）の送信処理が必要になります。

4.2 参考プログラムの解説

注．使用しているレジスタを（レジスタ名：ビット名）と表記しています。

4.2.1 I²C 初期化関数（i2c_init()）

(1) **ポート入力特別機能制御レジスタのポート入力許可ビットを入力許可に設定** (PICNT：PIEN0)
(2) **使用するポートの端子設定**
・P85、P87 からの出力を"H"にする （P8DATA）
・P85、P87 を出力モードに設定 （P8DIR）
・P85、P86、P87 を P8SMOD 設定のために"0"クリア （P8MOD）
・P85、P86、P87 をシリアル端子に設定 （P8SMOD）（**注** 1）
・P85、P87 を汎用ポート、P86 を RXD1 に設定 （P8MOD）

注 1. P85、P86、P87 **はシリアル端子とポート端子用にのみ使用します。**
そのため、以降の P8SMOD **の設定は省略しています。**

(3) SIO1 **送信制御レジスタの設定** （S1TCNT：CDIV, TEN）
・ボーレートジェネレータのカウントソースは 1 分周、送信禁止に設定
(4) SIO1 **受信制御レジスタを受信禁止に設定** （S1RCNT：REN）
(5) **割り込み設定**
・SIO1 送信割り込み禁止に設定 （ISIO1TXCR：ILEVEL）
・SIO1 受信割り込み禁止に設定 （ISIO1RXCR：ILEVEL）
・SIO1 送信割り込み要求禁止に設定 （SI03MASK：T1MASK）
・SIO1 受信割り込み要求禁止に設定 （SI03MASK：R1MASK）
(6) SIO1 **送受信モードレジスタの設定** （S1MOD：SMOD, CKS）
・CSIO モード、内部クロックに設定
(7) SIO1 **特殊モードレジスタの設定** （S1SMOD：CSIBL, SELCKS, SELFST, SEL3PNT, CKPOL）
・9 ビット、f(BCLK)/2、MSB ファースト、3 ポイントサンプリング無効、SCLK 立下りで送信データ出力/立ち上がりで受信データ取り込みに設定
(8) SIO1 **ボーレートレジスタの設定** （S1BAUR）
・ボーレートを 100kbps に設定
(9) **送信許可に設定** （S1TCNT：TEN）
(10) SIO1 **送信バッファにダミーデータを書き込み** （S1TXB）
(11) SIO1 **送信ステータスビットが**"0"**になるのを待つ** （S1TCNT：TSTAT）

4.2.2 各種初期化関数（init_func()）

(1) I²C **初期化関数の呼び出し**

4.2.3 メイン関数（main()）

(1) **割り込み禁止関数の呼び出し**
(2) **各種初期化関数の呼び出し**
(3) **割り込み許可関数の呼び出し**
(4) **マスタライト関数の呼び出し**
(5) **マスタリード関数の呼び出し**

4.2.4 スタートコンディション出力関数（i2c_start()）

(1) P85、P87 **の出力を**"H"**にする** （P8DATA）
(2) P85 **の出力を**"L"**にする** （P8DATA）
(3) **スタートコンディション出力後のウェイト**
(4) P85 **を** TXD1、P87 **を** SCLKO1 **に設定** （P8MOD）
(5) SIO1 **送信許可に設定** （S1TCNT：TEN）

4.2.5 ストップコンディション出力関数（i2c_stop()）

(1) SIO1 の送信動作停止中を確認 （S1TCNT：TSTAT）
(2) SIO1 の受信動作停止中を確認 （S1RCNT：RSTAT）
(3) P85、P87 からの出力を"L"にする値を設定 （P8DATA）
(4) P85、P87 を出力モードに設定 （P8DIR）
(5) P85、P87 を汎用ポートに設定 （P8MOD）
(6) SIO1 送信バッファにダミーデータを書き込み （S1TXB）
(7) ACK/NAK 出力後のウェイト
(8) P87 の出力を"H"に設定 （P8DATA）
(9) ストップコンディション出力前のウェイト
(10) P85 の出力を"H"に設定 （P8DATA）
(11) ストップコンディション出力後のウェイト
(12) SIO1 送信ステータスビットが"0"になるのを待つ （S1TCNT：TSTAT）

4.2.6 マスタライト関数（i2c_write()）

(1) スタートコンディション出力
(2) スレーブのデバイスアドレス出力（R/W ビットは"0"）
(3) スレーブのメモリアドレス出力
(4) 引数で指定された回数データ出力
(5) ストップコンディション出力

4.2.7 マスタリード関数（i2c_read()）

(1) スタートコンディション出力
(2) スレーブのデバイスアドレス出力（R/W ビットは"0"）
(3) スレーブのメモリアドレス出力
(4) ストップコンディション出力
(5) スタートコンディション出力
(6) スレーブのデバイスアドレス出力（R/W ビットは"1"）
(7) 引数で指定された回数データを読み出し、読み出したデータを保存
(8) ストップコンディション出力

4.2.8 データ送信関数（csio_txd_data()）

(1) 出力データ（8 ビット）をシフトし、9 ビットデータに変換する
(2) SIO1 送信データを送信バッファに書き込む （S1TXB）
(3) SIO1 送信ステータスビットが"0"になるのを待つ （S1TCNT：TSTAT）
(4) ACK の遅延対策の調整用ウェイト

4.2.9 データ受信関数（csio_rxd_data()）

(1) P85 を汎用ポートに設定 （P8MOD）（ダミーデータが出力されないようにするため）
(2) SIO1 受信許可に設定 （S1RCNT：REN）
(3) SIO1 受信動作を行うため、ダミーデータを送信バッファに書き込む （S1TXB）
(4) SIO1 受信完了ビットが"1"になるのを待つ （S1RCNT：RFIN）
(5) SIO1 受信データを読み出す （S1RXB）
(6) 受信データ（9 ビット）をシフトし、8 ビットデータに変換する
(7) SIO1 受信禁止に設定 （S1RCNT：REN）
(8) 受信データを上位関数へ返す

4.2.10 ウェイト関数（wait_us()）

(1) TML のクロックソースを BCLK/4 に設定 （CNTCKSEL、TML0CR）
(2) 引数で指定された値×μ秒の時間分をウェイトする

4.3 参考プログラム例

下記に I²C バス制御の参考プログラム例を示します。
尚、下記の参考プログラム例では、SFR 定義ファイルが必要です。最新の SFR 定義ファイルはホームページよりダウンロードできます。SFR 定義ファイル使用時は、お客様の環境に合わせてパスの設定をおこなってください。

4.3.1 i2c.c

```
/*""FILE COMMENT""*****************************************************
 *      M32R C Programming              Rev. 1.00
 *              < Sample Program for 32192/32195/32196 >
 *              < I2C bus control program >
 *
 *      Copyright (c) 2006 Renesas Technology Corporation
 *                      All Rights Reserved
 ***********************************************************************/

/************************************************************************/
/*              Include file                                            */
/************************************************************************/

#include                "..¥inc¥sfr32192_pragma.h"

/************************************************************************/
/*              Function prototype declaration                          */
/************************************************************************/

        void            i2c_init(void);                 /* I2C initialization */
        void            init_func(void);                /* Initial setup function */
        void            main(void);                     /* Main function */
        void            i2c_start(void);                /* make start condition */
        void            i2c_stop(void);                 /* make stop condition */
        void            i2c_write(UCHAR ,UCHAR ,UCHAR ,const UCHAR *);  /* write main function */
        void            i2c_read(UCHAR ,UCHAR ,UCHAR ,UCHAR *); /* read main function */
        void            csio_txd_data(UCHAR);           /* output data function */
        UCHAR           csio_rxd_data(void);            /* input data function */
        void            wait_us(ULONG);                 /* wait function */

/************************************************************************/
/*              Externally referenced variable                          */
/************************************************************************/

extern  void            DisInt( void );                 /* Interrupt disable function */
extern  void            EnInt( void );                  /* Interrupt enable function */

/************************************************************************/
/*              Define macro                                            */
/************************************************************************/

#define                 I2C_SLV_ADR             0x3C    /* Slave address (7 bit) 011 1100B */
#define                 I2C_MEM_ADR             0x61    /* Memory address */

#define                 I2C_WRITE_MODE          0x00u   /* Write mode (R/W=0) */
#define                 I2C_READ_MODE           0x01u   /* Read mode  (R/W=1) */
#define                 I2C_ADR_MASK            0xFEu   /* Address mask data */

#define                 I2C_ACK                 0x0000u /* ACK (0) */
#define                 I2C_ACK_MASK            0xFFFEu /* ACK mask data */

#define                 WAIT_FOR_START          2ul     /* Waiting time after output of start condition */
#define                 WAIT_FOR_STOP1          9ul     /* Waiting time 1 (after output of ACK/NAK) */
#define                 WAIT_FOR_STOP2          20ul    /* Waiting time 2 (before output of stop condition) */
#define                 WAIT_FOR_STOP3          20ul    /* Waiting time 3 (after output of stop condition) */
#define                 WAIT_FOR_ACK_DELAY      9ul     /* waiting for delay of ACK */

/* Setting port operation mode */
                                                        /* 0123 4567    */
#define                 P8DATA_INI              0x05u   /* 0000 0101B   P8 data register                */
                                                        /* |||| |||+--- port pin level : "high"        */
                                                        /* |||| ||+---- don't care                      */
                                                        /* |||| |+----- port pin level : "high"        */
                                                        /* ++++-+------ don't care                      */

```

```
                                                /* 0123 4567    */
#define                 P8DIR_INI               0x05u   /* 0000 0101B   P8 direction register          */
                                                /* |||| |||+--- output mode                    */
                                                /* |||| ||+---- don't care                     */
                                                /* |||| |+----- output mode                    */
                                                /* ++++-+------ don't care                     */

                                                /* 0123 4567    */
#define                 P8MOD_MASK              0x07u   /* 0000 0111B   P8 operation mode register     */
#define                 P8MOD_INI               0x02u   /* 0000 0010B   P8 operation mode register     */
                                                /* |||| |||+--- 0:P87, 1:SCLKI1/SCLKO1/TO21    */
                                                /* |||| ||+---- 0:P86, 1:RXD1/TO22             */
                                                /* |||| |+----- 0:P85, 1:TXD1/TO23             */
                                                /* ++++-+------ don't care                     */

                                                /* 0123 4567    */
#define                 P8SMOD_MASK             0x07u   /* 0000 0111B   P8 peripheral function select register */
                                                /* |||| |||+--- 0:SCLKI1/SCLKO1, 1:TO21        */
                                                /* |||| ||+---- 0:RXD1,          1:TO22        */
                                                /* |||| |+----- 0:TXD1,          1:TO23        */
                                                /* ++++-+------ don't care                     */

                                                /* 0123 4567    */
#define                 SCL_BIT                 0x01u   /* 0000 0001B   */
                                                /* |||| |||+--- SCL                           */
                                                /* ++++-+++---- don't care                     */

                                                /* 0123 4567    */
#define                 SDA_BIT                 0x04u   /* 0000 0100B   */
                                                /* |||| ||++--- don't care                     */
                                                /* |||| |+----- SDA                           */
                                                /* ++++-+------ don't care                     */

/* Setting serial interface */
                                                /* 0123 4567    */
#define                 S1TCNT_INI              0x00u   /* 0000 0000B   SIO1 transmit control register  */
                                                /* |||| |||+--- Disable transmission           */
                                                /* |||| +++---- don't care                     */
                                                /* ||++-------- 1 frequency dividing           */
                                                /* ++---------- don't care                     */

                                                /* 0123 4567    */
#define                 S1RCNT_INI              0x00u   /* 0000 0000B   SIO1 receive control register   */
                                                /* |||| ++++--- don't care                     */
                                                /* |||+-------- Disable reception              */
                                                /* +++--------- don't care                     */

                                                /* 0123 4567    */
#define                 S1MOD_INI               0x80u   /* 1000 0000B   SIO1 mode register             */
                                                /* |||| ++++--- don't care ( UART mode data )   */
                                                /* |||+-------- Select clock(specified separately) */
                                                /* +++--------- CSIO                           */

                                                /* 0123 4567    */
#define                 S1SMOD_INI              0x14u   /* 0001 0100B   SIO1 special mode register      */
                                                /* |||| |||+--- transmit at SCLK falling edge   */
                                                /* |||| ||+---- 3-point sampling valid         */
                                                /* |||| |+----- MSB first                       */
                                                /* |||| +------ f(BCLK)/2                      */
                                                /* ++++-------- 9-bit CSIO                     */

#define                 SIO1_BAUR_100K          (100-1) /* 20MHz / (100kbps * 1 * 2) = 100 */
#define                 ILEVEL_7                0x07u   /* Interrupt Disable */

#define                 DUMMY_DATA              0x0000  /* Dummy transmission data */

/************************************************************************/
/*            Global variable                                           */
/************************************************************************/


/*""FUNC COMMENT""*************************************************************
 * Function name: i2c_init(void)
 *-----------------------------------------------------------------------
 * Description  : Initialize I2C
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
```

```
144  * Returns      : -
145  *------------------------------------------------------------------------
146  * Notes        : -
147  *"""FUNC COMMENT END"""*************************************************/
148 void     i2c_init(void)
149 {
150         PICNT  |= PIEN0;                                /* Enable port input */
151
152         P8DATA |= P8DATA_INI;                           /* P87(SCL), P85(SDA) High */
153         P8DIR  |= P8DIR_INI;                            /* P87(SCL), P85(SDA) output */
154         P8MOD  &= ~P8MOD_MASK;                          /* P85 - P87 : 0 clear for P8SMOD setting  */
155         P8SMOD &= ~P8SMOD_MASK;                         /* P85 - P87 : SIO mode  */
156         P8MOD  |= P8MOD_INI;                            /* P85, P87: port/ P86: RXD1 */
157
158         S1TCNT = S1TCNT_INI;                            /* Disable transmission  */
159         S1RCNT = S1RCNT_INI;                            /* Disable reception */
160
161         ISIO1TXCR = ILEVEL_7;                           /* Set SIO1 transmit interrupt priority level */
162         ISIO1RXCR = ILEVEL_7;                           /* Set SIO1 receive interrupt priority level */
163         SI03MASK &= ~T1MASK;                            /* Disable SIO1 transmit interrupt request */
164         SI03MASK &= ~R1MASK;                            /* Disable SIO1 receive interrupt request */
165
166         S1MOD  = S1MOD_INI;                             /* Set data format */
167         S1SMOD = S1SMOD_INI;                            /* Set Transmit/Receive clock polarity */
168         S1BAUR = SIO1_BAUR_100K;                        /* Set baudrate */
169         S1TCNT |= TEN;                                  /* Enable transmission */
170
171         S1TXB = DUMMY_DATA;                             /* Dummy write */
172         while( (S1TCNT & TSTAT) == TSTAT){              /* Waiting for end of transmission */
173                 ;
174         }
175 }
176
177 /*"""FUNC COMMENT"""*****************************************************
178  * Function name: init_func()
179  *------------------------------------------------------------------------
180  * Description  : Call various initialization functions
181  *------------------------------------------------------------------------
182  * Argument     : -
183  *------------------------------------------------------------------------
184  * Returns      : -
185  *------------------------------------------------------------------------
186  * Notes        : -
187  *"""FUNC COMMENT END"""*************************************************/
188 void init_func(void)
189 {
190         i2c_init();                                     /* i2c initialize */
191 }
192
193 /*"""FUNC COMMENT"""*****************************************************
194  * Function name: main()
195  *------------------------------------------------------------------------
196  * Description  : -
197  *------------------------------------------------------------------------
198  * Argument     : -
199  *------------------------------------------------------------------------
200  * Returns      : -
201  *------------------------------------------------------------------------
202  * Notes        : -
203  *"""FUNC COMMENT END"""*************************************************/
204 void main(void)
205 {
206 const   UCHAR   SetData[]={0x05};                       /* setting data */
207         UCHAR   ReadBuff[ 4 ];
208
209         DisInt();                                       /* Disable interrupt */
210
211         init_func();
212
213         EnInt();                                        /* Enable interrupt */
214
215         /* master write */
216         i2c_write((UCHAR)I2C_SLV_ADR, (UCHAR)I2C_MEM_ADR, (UCHAR)(sizeof(SetData)), SetData);
217
218         /* master read */
219         i2c_read( (UCHAR)I2C_SLV_ADR, (UCHAR)I2C_MEM_ADR, (UCHAR)1, ReadBuff );
220
221         while( 1 ){
```

```
222                 ;
223         }
224 }
225
226 /*""FUNC COMMENT""************************************************************
227  * Function name: i2c_start(void)
228  *-----------------------------------------------------------------------
229  * Description  : Start condition
230  *-----------------------------------------------------------------------
231  * Argument     : -
232  *-----------------------------------------------------------------------
233  * Returns      : -
234  *-----------------------------------------------------------------------
235  * Notes        : -
236  *""FUNC COMMENT END""*********************************************************/
237 void i2c_start(void)
238 {
239         P8DATA |=  (SCL_BIT | SDA_BIT);                 /* SCL, SDA High */
240         P8DATA &= ~SDA_BIT;                             /* SDA Low */
241
242         wait_us( WAIT_FOR_START );
243
244         P8MOD |= (SCL_BIT | SDA_BIT);                   /* change to SIO mode */
245
246         S1TCNT |= TEN;                                  /* Enable transmission */
247 }
248
249 /*""FUNC COMMENT""************************************************************
250  * Function name: i2c_stop(void)
251  *-----------------------------------------------------------------------
252  * Description  : Stop condition
253  *-----------------------------------------------------------------------
254  * Argument     : -
255  *-----------------------------------------------------------------------
256  * Returns      : -
257  *-----------------------------------------------------------------------
258  * Notes        : -
259  *""FUNC COMMENT END""*********************************************************/
260 void i2c_stop(void)
261 {
262         while( (S1TCNT & TSTAT) == TSTAT ){             /* Check for end of transmission */
263                 ;
264         }
265         while( (S1RCNT & RSTAT) == RSTAT ){             /* Check for reception finished */
266                 ;
267         }
268
269         P8DATA &= ~(SCL_BIT | SDA_BIT);                 /* SCL, SDA Low output */
270         P8DIR  |=  (SCL_BIT | SDA_BIT);                 /* output mode */
271         P8MOD  &= ~(SCL_BIT | SDA_BIT);                 /* change to port input/output */
272
273         S1TXB = DUMMY_DATA;                             /* Dummy write */
274
275         wait_us( WAIT_FOR_STOP1 );                      /* wait No.1 */
276         P8DATA |= SCL_BIT;                              /* SCL High output */
277         wait_us( WAIT_FOR_STOP2 );                      /* wait No.2 */
278         P8DATA |= SDA_BIT;                              /* SDA High output */
279         wait_us( WAIT_FOR_STOP3 );                      /* wait No.3 */
280
281         while( (S1TCNT & TSTAT) == TSTAT ){             /* End confirmation of dummy writing */
282                 ;
283         }
284
285         /* Wait timing ***************************************
286          *        function start        function end         *
287          *                                                   *
288          *             |        _____________ |             *
289          *      SCL    | |_______|            |             *
290          *             |                      |             *
291          *             |                _____ |             *
292          *      SDA    | _______________|     |             *
293          *             |                      |             *
294          *      wait->      No.1   No.2   No.3              *
295          *****************************************************/
296
297 }
298
299 /*""FUNC COMMENT""************************************************************
```

```
 * Function name: i2c_write(UCHAR ucSlaveAddr,UCHAR ucMemAddr,UCHAR ucDataCount,const UCHAR *ucSendBuffer)
 *------------------------------------------------------------------------
 * Description  : Master write
 *------------------------------------------------------------------------
 * Argument     : UCHAR ucSlaveAddr   : Slave address
 *              : UCHAR ucMemAddr     : Memory address
 *              : UCHAR ucDataCount   : Number of transmission data
 *              : UCHAR *ucSendBuffer : Transmission data storage position
 *------------------------------------------------------------------------
 * Returns      : -
 *------------------------------------------------------------------------
 * Notes        : -
 *"""FUNC COMMENT END"""*****************************************************/
void i2c_write(UCHAR ucSlaveAddr,UCHAR ucMemAddr,UCHAR ucDataCount,const UCHAR *ucSendBuffer)
{
        i2c_start();                                    /* Start condition output */
        csio_txd_data((UCHAR)(((UCHAR)(ucSlaveAddr << 1u) & I2C_ADR_MASK) | I2C_WRITE_MODE));
                                                        /* Slave address+R/W bit(0) */
        csio_txd_data(ucMemAddr);                       /* Memory address output */
        while( (ucDataCount--) != 0 ){
                csio_txd_data(*ucSendBuffer++);         /* Data output */
        }
        i2c_stop();                                     /* Stop condition output */
}

/*"""FUNC COMMENT"""*************************************************************
 * Function name: i2c_read(UCHAR ucSlaveAddr,UCHAR ucMemAddr,UCHAR ucDataCount,UCHAR *ucReadBuffer)
 *------------------------------------------------------------------------
 * Description  : Master read
 *------------------------------------------------------------------------
 * Argument     : UCHAR ucSlaveAddr   : Slave address
 *              : UCHAR ucMemAddr     : Memory address
 *              : UCHAR ucDataCount   : Number of reading data
 *              : UCHAR *ucReadBuffer : Reading data storage position
 *------------------------------------------------------------------------
 * Returns      : -
 *------------------------------------------------------------------------
 * Notes        : -
 *"""FUNC COMMENT END"""*****************************************************/
void i2c_read(UCHAR ucSlaveAddr,UCHAR ucMemAddr,UCHAR ucDataCount,UCHAR *ucReadBuffer)
{
        /* Setting Read address */
        i2c_start();                                    /* Start condition output */
        csio_txd_data((UCHAR)(((UCHAR)(ucSlaveAddr << 1u) & I2C_ADR_MASK) | I2C_WRITE_MODE));
                                                        /* Slave address+R/W bit(0) */
        csio_txd_data(ucMemAddr);                       /* Memory address output */
        i2c_stop();                                     /* Stop condition output */

        /* Read register data */
        i2c_start();
        csio_txd_data((UCHAR)(((UCHAR)(ucSlaveAddr << 1u) & I2C_ADR_MASK) | I2C_READ_MODE));
                                                        /* Slave address+R/W bit(1) */
        while( (ucDataCount--) != 0){
                *ucReadBuffer++ = csio_rxd_data();      /* Data read */
        }
        i2c_stop();
}

/*"""FUNC COMMENT"""*************************************************************
 * Function name: csio_txd_data(UCHAR ucSendData)
 *------------------------------------------------------------------------
 * Description  : Data transmission
 *------------------------------------------------------------------------
 * Argument     : UCHAR ucSendData : send data
 *------------------------------------------------------------------------
 * Returns      : -
 *------------------------------------------------------------------------
 * Notes        : Only ACK is output (NAK is not output)
 *"""FUNC COMMENT END"""*****************************************************/
void csio_txd_data(UCHAR ucSendData)
{
        USHORT usSend = 0;

        usSend = (USHORT)(((UCHAR)(ucSendData << 1u) & I2C_ACK_MASK) | I2C_ACK);
                                                        /* LSB is ACK (0) */
        S1TXB = usSend;

        while( (S1TCNT & TSTAT) == TSTAT ){             /* Waiting for end of transmission */
```

```
                ;
        }

        wait_us( WAIT_FOR_ACK_DELAY );

}

/*""FUNC COMMENT""************************************************************
 * Function name: csio_rxd_data(void)
 *-----------------------------------------------------------------------
 * Description  : Data reception
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : Receive data
 *-----------------------------------------------------------------------
 * Notes        : ACK or NAK is not judged
 *""FUNC COMMENT END""*********************************************************/
UCHAR  csio_rxd_data(void)
{
        USHORT usData;
        UCHAR  ucRet;

        P8MOD &= ~SDA_BIT;                              /* change to port input/output */
        S1RCNT |= REN;                                  /* Enable reception */

        S1TXB = DUMMY_DATA;                             /* Dummy write */
        while( (S1RCNT & RFIN) != RFIN ){               /* Waiting for reception finished */
                ;
        }

        usData = S1RXB;
        usData = (USHORT)(usData >> 1u);                /* delete ACK bit */
        ucRet = (UCHAR)(0x00ffu & usData);

        S1RCNT &= ~REN;                                 /* Disable reception */

        return ucRet;
}

/*""FUNC COMMENT""************************************************************
 * Function name: wait_us(ULONG count)
 *-----------------------------------------------------------------------
 * Description  : wait(microsecond)
 *-----------------------------------------------------------------------
 * Argument     : ULONG count : wait counter
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : -
 *""FUNC COMMENT END""*********************************************************/
void wait_us(ULONG count)
{
        ULONG   cnt;                                    /* TML Counter Value */

        CNTCKSEL = 0x00;                                /* TML clock select(BCLK/4) */
        TML0CR &= ~TML0CKS;

        cnt = TML0CT;

        while((TML0CT-cnt)<count * 10ul){
                ;
        }
}
```

5. 参考ドキュメント

・32192/32195/32196 グループ ハードウェアマニュアル Rev.1.10
・M3T-CC32R V.5.00 ユーザーズマニュアル（コンパイラ編）
・M3T-AS32R V.5.00 ユーザーズマニュアル（アセンブラ編）
・M32R-FPU ソフトウェアマニュアル Rev.1.01

（最新版をルネサス テクノロジホームページから入手してください。）

6. ホームページとサポート窓口

ルネサス テクノロジホームページ
http://www.renesas.com/

ルネサス製品全般に関するお問合せと M32R ファミリに関する技術的なお問合せ先
コンタクトセンタ：csc@renesas.com

改訂記録	32192/32195/32196 グループ シリアルインタフェースを使用した I²C バス制御例 アプリケーションノート

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2006.10.17	ー	初版発行

安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ(http://www.renesas.com)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任は負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。