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。

M16C/Tiny シリーズ
リアルタイムクロック機能

1. 要約

タイマ機能を使用し、リアルタイムクロック機能を作成します。

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　　: M16C/26A, 26, 28, 29 グループ

M16C/26A, 26, 28, 29 グループと同様の SFR(周辺機能制御レジスタ)を持つ他の M16C ファミリでも本プログラムを使用することができます。ただし、一部の機能を機能追加等で変更している場合がありますのでマニュアルで確認してください。このアプリケーションノートをご使用に際しては十分な評価を行ってください。

3. 動作説明

(1) タイマ A0 をタイマモード、カウントソース fc32（サブクロックの 32 分周）に設定します。
また、タイマ割り込みを 1 秒間隔にするため、タイマ値を（400-1）h に設定します。
ただし、Xcin = 32.768kHz とします。
(2) タイマ A0 のカウント開始フラグを“1”にすると、タイマ A0 のカウンタ値がダウンカウントします。
(3) タイマ A0 のアンダフローのタイミングで、タイマ A0 割り込み要求ビットが“1”に設定されます。
(4) タイマ A0 割り込み要求ビットが“1”に設定される毎に、1 秒カウントアップし、年月日、曜日、時刻を RAM に格納します。
ただし、RAM に格納する値は、16 進数で格納します。
(5) タイマ開始後、西暦 1 年 1 月 1 日、日曜日、0 時 0 分 0 秒からカウント開始します。

4. ソフトウェア説明

4.1 関数説明

表 1 に、使用する関数の説明を示します。

表 1　関数説明

関数名	ラベル名	機能
メイン	main	使用レジスタの設定、使用 RAM の初期化、割り込みの許可、時刻設定処理関数呼び出し
時刻設定処理	time_set	秒、分、時、日、曜日、週データの設定、年月日設定処理関数呼び出し
年月日設定処理	date_set	日、月、年データの設定、閏年判定処理関数呼び出し
閏年判定処理	leap_year_check	閏年判定

4.2 レジスタ説明

表 2 に、使用するレジスタの説明を示します。

※ 設定値は、M16C/26A グループ用の設定値です。
また、テクニカルアップデート（TN-16C-119A）の対応製品は、
M16C/26, 28, 29 です。

表 2　レジスタ説明

レジスタ名		アドレス	設定値	機能
PRCR	プロテクトレジスタ	000Ah	01h 04h 00h	・PD9、PACR、S4C、NDDR レジスタへの 書き込み許可／禁止 ・CM0、CM1、CM2、ROCR、PLC0、PCLKR、 CCLKR レジスタへの書き込み許可／禁止 ※2 ビット目は”1”を書き込んだ後、任意の番地に 書き込みを実行すると”0”になります。 他のビットは”0”になりませんので、 プログラムで”0”にしてください。
PACR	端子割り当て制御レジスタ	025Dh	04h	0～2 ビット目　設定値 001 … 42 ピン版 (M16C/26A) 010 … 64 ピン版 (M16C/28, 29) 011 … 80 ピン版 (M16C/28, 29) 100 … 48 ピン版 (M16C/26A) ※PRCR レジスタの 2 ビット目を”1”にした次の 命令で、PACR レジスタに書き込んでください。 M16C/26 には、このレジスタは存在しません。
IFSR2A	割り込み要因選択レジスタ 2	035Eh	01h	・0 ビット目を必ず”1”に設定してください。 (M16C/26A, 28) ・0 ビット目を必ず”0”に設定してください。 (M16C/29) ※0 ビット目を設定してから割り込みを 許可してください。 M16C/26 には、このレジスタは存在しません。
CM0	システムクロック制御レジスタ 0	0006h	18h	メインクロック、分周なし、メインクロック発振、 Xcin-Xcout 発振機能、Xcin-Xcout 駆動 High、 周辺機能クロック停止しない、 クロック出力機能を P9_0 (M16C/28 はリザーブ) ※テクニカルアップデート(TN-16C-119A)参照
TABSR	カウント開始フラグ	0380h	00h 01h	・タイマ A0 カウント停止 ・タイマ A0 カウント開始
TA0MR	タイマ A0 モードレジスタ	0396h	C0h	カウントソース fc32、ゲート機能なし、 パルス出力なし、タイマモード
TA0	タイマ A0 レジスタ	0386h	03FFh	タイマ A0 割り込みを 1 秒間隔にするため、 (400-1)h に設定してください。
TA0IC	タイマ A0 割り込み制御レジスタ	0055h	00h	割り込みレベル 0、割り込み要求ビット設定

4.3 RAM 説明

表 3 に、使用する RAM の説明を示します。

表 3　RAM 説明

RAM 名	機能	データ長	使用関数
sec_cnt	秒データを 16 進数で格納	1byte	main, time_set
min_cnt	分データを 16 進数で格納	1byte	main, time_set
hour_cnt	時データを 16 進数で格納	1byte	main, time_set
day_cnt	日データを 16 進数で格納	1byte	main, time_set, date_set, leap_year_check
week_cnt	週データを 16 進数で格納 0x00: 日曜日 0x01: 月曜日 0x02: 火曜日 0x03: 水曜日 0x04: 木曜日 0x05: 金曜日 0x06: 土曜日	1byte	main, time_set
month_cnt	月データを 16 進数で格納	1byte	main, date_set, leap_year_check
year_cnt	年データを 16 進数で格納	2byte	main, date_set, leap_year_check

4.4 ROM 説明

表 4 に、使用する ROM の説明を示します。

表 4　ROM 説明

ROM 名	機能	データ長	使用関数
day_max_tbl[12]	月別最大日数データを 16 進数で格納 [0]（ 1 月 ）… 0x1F （ 31 日 ） [1]（ 2 月 ）… 0x1C （ 28 日 ） [2]（ 3 月 ）… 0x1F （ 31 日 ） [3]（ 4 月 ）… 0x1E （ 30 日 ） [4]（ 5 月 ）… 0x1F （ 31 日 ） [5]（ 6 月 ）… 0x1E （ 30 日 ） [6]（ 7 月 ）… 0x1F （ 31 日 ） [7]（ 8 月 ）… 0x1F （ 31 日 ） [8]（ 9 月 ）… 0x1E （ 30 日 ） [9]（ 10 月 ）… 0x1F （ 31 日 ） [10]（ 11 月 ）… 0x1E （ 30 日 ） [11]（ 12 月 ）… 0x1F （ 31 日 ）	12byte	leap_year_check

5. 設定手順

M16C/26A グループで動作させた場合のフローチャートを図に示します。

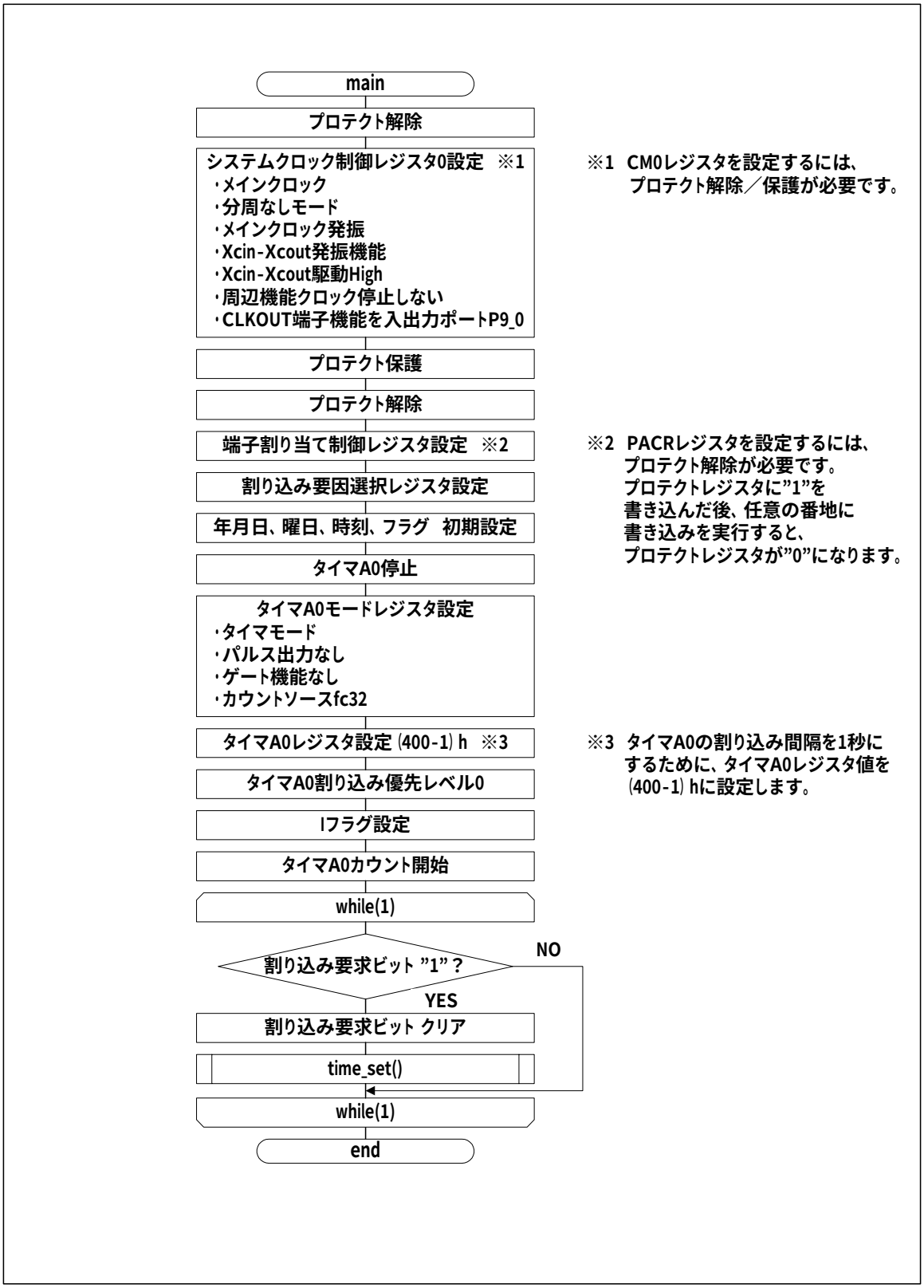

図1 フローチャート（main）

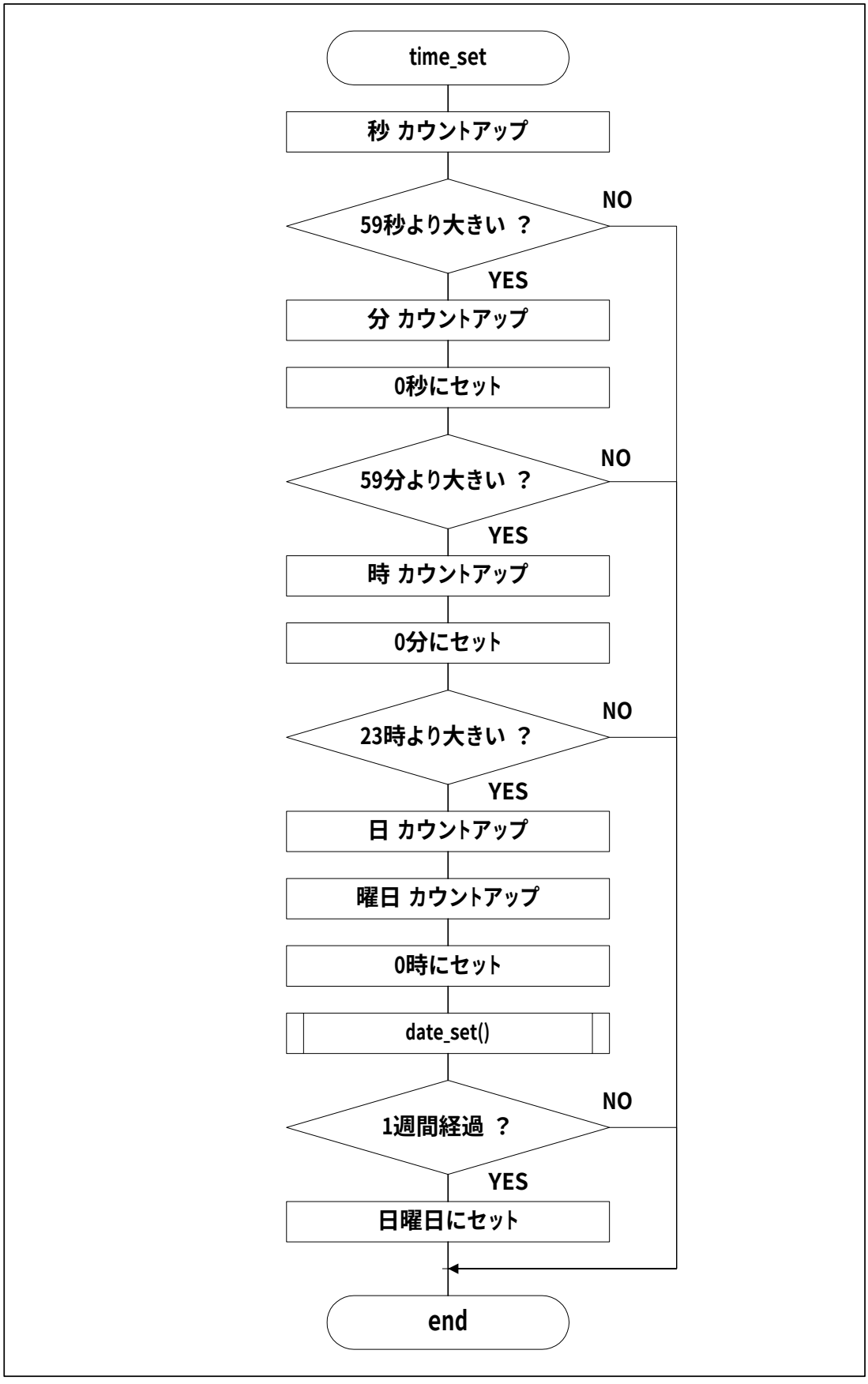

図2 フローチャート（time_set）

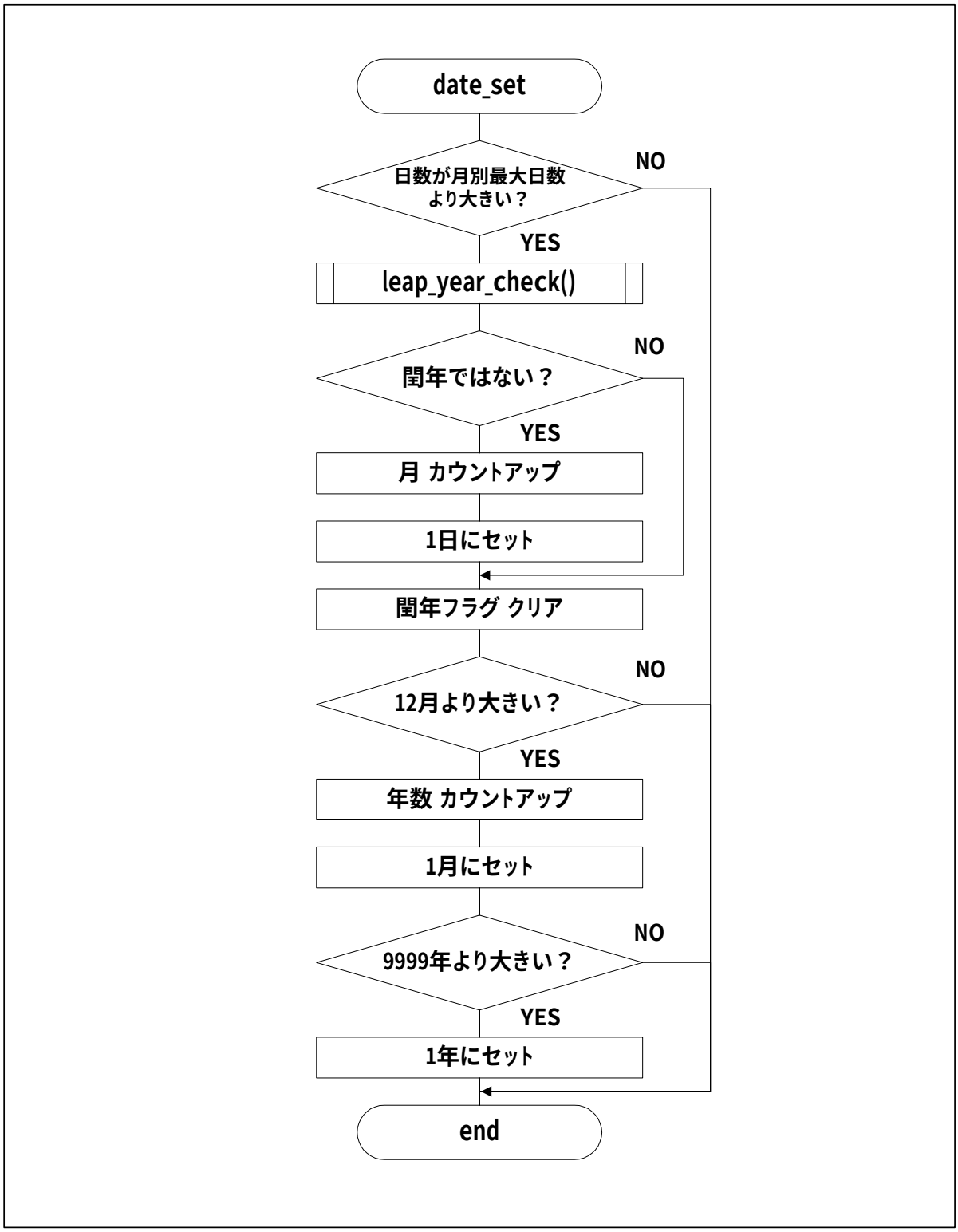

図3 フローチャート（date_set）

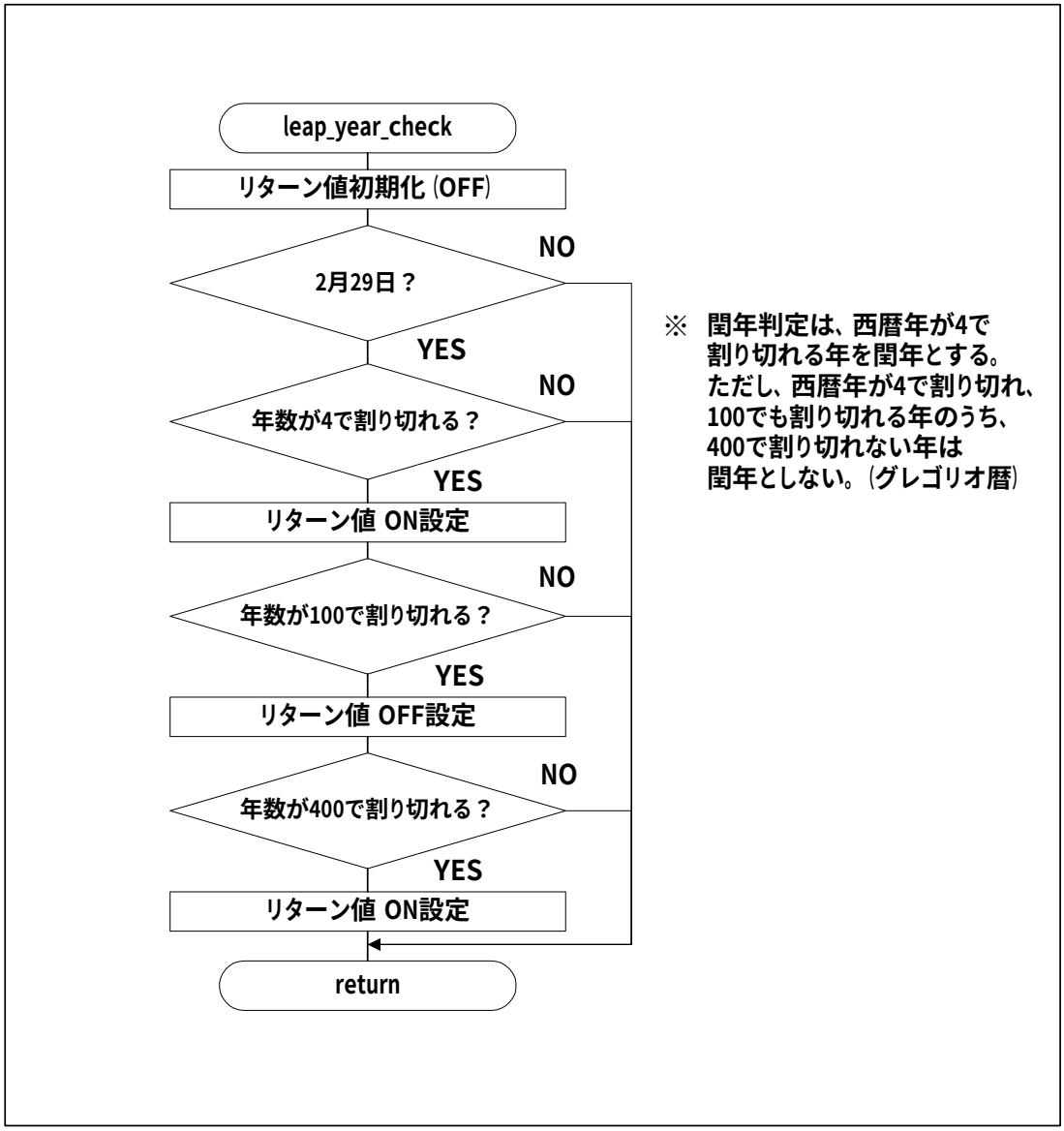

図4 フローチャート（leap_year_check）

6. 参考プログラム

M16C/Tiny シリーズのグループ別で動作させた場合のプログラムを例として示します。

6.1 参考プログラム例（M16C/26A グループ）

```
/***********************************************************************/
/*                                                                     */
/*      M16C/26A Program Collection                                    */
/*                                                                     */
/*      FILE NAME   :   rjj05b0849_src.c                               */
/*      CPU         :   M16C/26A Group                                 */
/*      FUNCTION    :   Real Time Clock function                       */
/*      HISTORY     :   2005.06.30  Ver 1.00                           */
/*                                                                     */
/*      Copyright(C)2005, Renesas Technology Corp.                     */
/*      Copyright(C)2005, Renesas Solutions Corp.                      */
/*      All rights reserved.                                           */
/*                                                                     */
/***********************************************************************/

/***********************************************************************/
/*      Include                                                        */
/***********************************************************************/
#include    "sfr26a.h"

/***********************************************************************/
/*      Prototype declaration                                          */
/***********************************************************************/
void main(void);
void time_set(void);
void date_set(void);
unsigned char leap_year_check(void);

/***********************************************************************/
/*      DEFINE                                                         */
/***********************************************************************/
#define YEAR_MAX    0x270F      /* The maximum value at year   ( 1 to 9999 ) */
#define MONTH_MAX   0x0C        /* The maximum values at month ( 1 to 12 )   */
#define WEEK_MAX    0x06        /* The maximum value on a day of the week    */
                                /*       0: Sunday                           */
                                /*       1: Monday                           */
                                /*       2: Tuesday                          */
                                /*       3: Wednesday                        */
                                /*       4: Thursday                         */
                                /*       5: Friday                           */
```

```
                                /*       6: Saturday                          */
#define HOUR_MAX    0x17        /* The maximum value at hour   ( 0 to 23 )   */
#define MIN_MAX     0x3B        /* The maximum value at minute ( 0 to 59 )   */
#define SEC_MAX     0x3B        /* The maximum value at second ( 0 to 59 )   */

/*****************************************************************************/
/*      RAM                                                                  */
/*****************************************************************************/
unsigned int  year_cnt;         /* Count at year                             */
unsigned char month_cnt;        /* Count of month                            */
unsigned char day_cnt;          /* Count of day                              */
unsigned char week_cnt;         /* Count of week                             */
unsigned char hour_cnt;         /* Count of hour                             */
unsigned char min_cnt;          /* Count of minute                           */
unsigned char sec_cnt;          /* Count of second                           */

/*****************************************************************************/
/*      ROM                                                                  */
/*****************************************************************************/
const unsigned char day_max_tbl[12] = {
            0x1F,               /* January   (1 to 31)                       */
            0x1C,               /* February  (1 to 28)                       */
            0x1F,               /* March     (1 to 31)                       */
            0x1E,               /* April     (1 to 30)                       */
            0x1F,               /* May       (1 to 31)                       */
            0x1E,               /* June      (1 to 30)                       */
            0x1F,               /* July      (1 to 31)                       */
            0x1F,               /* August    (1 to 31)                       */
            0x1E,               /* September (1 to 30)                       */
            0x1F,               /* October   (1 to 31)                       */
            0x1E,               /* November  (1 to 30)                       */
            0x1F                /* December  (1 to 31)                       */
};

/*****************************************************************************/
/*      Pragma                                                               */
/*****************************************************************************/

/*****************************************************************************/
/*      Main Program                                                         */
/*****************************************************************************/
void main(void)
{

    prcr = 0x01;        /* Protect Register */
    /* 00000001B */
    /*        +-------- Enable write to CM0, CM1, CM2, ROCR,        */
```

```
/*                              PLC0, PCLKR and CCLKR registers */
/*                    0 : Write protected                       */
/*                    1 : Write enabled                         */

cm0 = 0x18;         /* System clock control register 0 */
/* 00011000B */
/* ||||||++-------- Clock output function select bit           */
/* ||||||                          (M16C/26A, 29 Group)        */
/* ||||||              00 : I/O port P90                       */
/* ||||||           Reserved bits    (M16C/26, 28 Group)       */
/* ||||||                                                      */
/* |||||+---------- WAIT peripheral function clock stop bit    */
/* |||||               0 : Do not stop peripheral function     */
/* |||||                                    clock in wait mode */
/* |||||               1 : Stop peripheral function clock      */
/* |||||                                          in wait mode */
/* ||||+----------- XCIN-XCOUT drive capacity select bit       */
/* ||||                0 : LOW                                 */
/* ||||                1 : HIGH                                */
/* |||+------------ Port XC select bit                        */
/* |||                 0 : I/O port P86, P87                   */
/* |||                 1 : XCIN-XCOUT generation function      */
/* ||+------------- Main clock stop bit                       */
/* ||                  0 : On                                  */
/* ||                  1 : Off                                 */
/* |+-------------- Main clock division select bit 0           */
/* |                   0 : CM16 and CM17 valid                 */
/* |                         Initial value of CM1 :            */
/* |                                    No division mode     */
/* |                   1 : Division by 8 mode                  */
/* +--------------- System clock select bit                    */
/*                     0 : Main clock, PLL clock,              */
/*                                   or ring oscillator clock */
/*                     1 : Sub-clock                           */

prcr = 0x00;        /* Protect Register */
/* 00000000B */
/*        +-------- Enable write to CM0, CM1, CM2, ROCR,       */
/*                              PLC0, PCLKR and CCLKR registers */
/*                    0 : Write protected                       */
/*                    1 : Write enabled                         */

prcr = 0x04;        /* Protect Register */
/* 00000100B */
/*      +---------- Enable write to PD9, PACR, S4C             */
/*                                          and NDDR registers */
/*                    0: Write protected                        */
```

```
/*                      1: Write enabled                        */

pacr = 0x04;        /* Pin assignment control register */
/* 00000100B */
/*      +++-------- Pin enabling bit                           */
/*                      001 : 42 pin (M16C/26A Group)           */
/*                      010 : 64 pin (M16C/28, 29 Group)        */
/*                      011 : 80 pin (M16C/28, 29 Group)        */
/*                      100 : 48 pin (M16C/26A Group)           */
/*                  All other values are reserved. Do not use.  */

ifsr2a = 0x01;      /* Interrupt request cause select register 2 */
/* 00000001B */
/*        +-------- Must be set to “1”. (M16C/26A, 28 Group)   */
/*                  Must be set to “0”. (M16C/29 Group)        */
/*                  Set this 0th bit before you enable         */
/*                                   interrupt after resetting. */

sec_cnt = 0x00;     /* Intialization of count of second        */
min_cnt = 0x00;     /* Intialization of count of minute        */
hour_cnt = 0x00;    /* Intialization of count of hour          */
week_cnt = 0x00;    /* Intialization of count of week          */
day_cnt = 0x01;     /* Intialization of count of day           */
month_cnt = 0x01;   /* Intialization of count of month         */
year_cnt = 0x0001;  /* Intialization of count of year          */

tabsr = 0x00;       /* Count start flag */
/* 00000000B */
/*        +-------- Timer A0 count start flag                   */
/*                      0 : Stops counting                      */
/*                      1 : Starts counting                     */

ta0mr = 0xC0;       /* Timer A0 mode register */
/* 11000000B */
/* || |||++-------- Operation mode select bit                   */
/* || |||               00 : Timer mode                       */
/* || ||+---------- Pulse output function select bit           */
/* || ||                0 : Pulse is not output                */
/* || ||                    (TAiOUT pin is a normal port pin)  */
/* || ||                1 : Pulse is output                    */
/* || ||                    (TAiOUT pin is a pulse output pin) */
/* || ++----------- Gate function select bit                   */
/* ||                   00 : Gate function not available       */
/* ||                        (TAiIN pin functions as I/O port) */
/* ++--------------- Count source select bit                   */
/*                      11 : fC32                               */
```

```
    ta0 = 0x3FF;        /* Timer A0 register (400-1)h */

    ta0ic = 0x00;       /* Timer A0 Interrupt control register */
    /* 00000000B */
    /*      +++-------- Interrupt priority level select bit         */
    /*                      000 : Level 0                           */

    /* Setting "I" flag permitting */
    asm("FSET I");      /* "I" flag */

    tabsr = 0x01;       /* Count start flag */
    /* 00000001B */
    /*        +-------- Timer A0 count start flag                   */
    /*                      0 : Stops counting                      */
    /*                      1 : Starts counting                     */

    while(1){
          /* Loop */
        if(ir_ta0ic == 1){
            /* When the interrupt request bit is "1" */
          ir_ta0ic = 0;       /* Interrupt request bit clear */
          time_set();         /* Time_set Function          */
      }
    }

}


void time_set(void)
{

    sec_cnt++;                          /* Count of second addition */

    if(sec_cnt > SEC_MAX){
        min_cnt++;                      /* Count of minute addition */
        sec_cnt = 0;                    /* Count of second clear    */

        if(min_cnt > MIN_MAX){
            hour_cnt++;                 /* Count of hour addition */
            min_cnt = 0;                /* Count of minute clear  */

            if(hour_cnt > HOUR_MAX){
                day_cnt++;              /* Count of day addition  */
                week_cnt++;             /* Count of week addition */
                hour_cnt = 0;           /* Count of hour clear    */

                date_set();             /* date_set Function      */
```

```
                if(week_cnt > WEEK_MAX){
                    week_cnt = 0;       /* Count of week clear */

                }
            }
        }
    }
}


void date_set(void)
{

    unsigned char leap_year_flg;        /* Set flag at leap year */
                                        /* 0: Not set            */
                                        /* 1: Set                */

    if( day_cnt > day_max_tbl[month_cnt-1] ){
                /* When you exceed the maximum days */
        leap_year_flg = leap_year_check();  /* leap_year_check Function */

        if(leap_year_flg == 0){
                /* When it is not a leap year */
            month_cnt++;                    /* Count of month addition */
            day_cnt = 0x01;                 /* Count of day clear      */
        }

        leap_year_flg = 0;                  /* Set flag at leap year clear */

        if(month_cnt > MONTH_MAX){
                /* When you exceed the maximum months */
            year_cnt++;                     /* Count of year addition */
            month_cnt = 0x01;               /* Count of month clear   */

            if(year_cnt > YEAR_MAX){
                /* When you exceed the maximum years */
                year_cnt = 0x0001;          /* Count of year clear */
            }
        }
    }
}


unsigned char leap_year_check(void)
{
```

```
    unsigned char rtn;      /* Return value */
                            /*            0: Not set leap year */
                            /*            1: Set leap year     */


    rtn= 0;                         /* Return value initialization */

    if( (month_cnt == 0x02)&&(day_cnt == 0x1D) ){
                /* Comparison at leap year */

        /*************************************************************/
        /* The judgment makes year when the age can be fourthly divided */
        /* at the Christian era a leap year at the leap year.        */
        /* However, year that cannot be divided by 400 is not made a */
        /* leap year among as many as 100 years that can be fourthly */
        /* divided at the Christian era the age, can be divided.     */
        /* (Gregorian)                                               */
        /*************************************************************/
        if( (year_cnt%4) == 0 ){
                /* When years can be fourthly divided */
            rtn = 1;                /* Set leap year */

            if( (year_cnt%100) == 0 ){
                /* When years can be divided by 100 */
                rtn = 0;            /* Not set leap year */

                if( (year_cnt%400) == 0 ){
                /* When years can be divided by 400 */
                    rtn = 1;        /* Set leap year */
                }
            }
        }
    }

    return(rtn);

}
```

6.2 参考プログラム例（M16C/26 グループ）

```
/************************************************************************/
/*                                                                      */
/*      M16C/26 Program Collection                                      */
/*                                                                      */
/*      FILE NAME   :   rjj05b0849_src.c                                */
/*      CPU         :   M16C/26 Group                                   */
/*      FUNCTION    :   Real Time Clock function                        */
/*      HISTORY     :   2005.06.30  Ver 1.00                            */
/*                                                                      */
/*      Copyright(C)2005, Renesas Technology Corp.                      */
/*      Copyright(C)2005, Renesas Solutions Corp.                       */
/*      All rights reserved.                                            */
/*                                                                      */
/************************************************************************/

/************************************************************************/
/*      Include                                                         */
/************************************************************************/
#include    "sfr26.h"

/************************************************************************/
/*      Prototype declaration                                           */
/************************************************************************/
void main(void);
void time_set(void);
void date_set(void);
unsigned char leap_year_check(void);

/************************************************************************/
/*      DEFINE                                                          */
/************************************************************************/
#define YEAR_MAX    0x270F      /* The maximum value at year   ( 1 to 9999 ) */
#define MONTH_MAX   0x0C        /* The maximum values at month ( 1 to 12 )   */
#define WEEK_MAX    0x06        /* The maximum value on a day of the week    */
                                /*       0: Sunday                           */
                                /*       1: Monday                           */
                                /*       2: Tuesday                          */
                                /*       3: Wednesday                        */
                                /*       4: Thursday                         */
                                /*       5: Friday                           */
                                /*       6: Saturday                         */
#define HOUR_MAX    0x17        /* The maximum value at hour   ( 0 to 23 )   */
#define MIN_MAX     0x3B        /* The maximum value at minute ( 0 to 59 )   */
```

```
#define SEC_MAX     0x3B        /* The maximum value at second ( 0 to 59 )   */

/************************************************************************/
/*      RAM                                                             */
/************************************************************************/
unsigned int  year_cnt;         /* Count at year                        */
unsigned char month_cnt;        /* Count of month                       */
unsigned char day_cnt;          /* Count of day                         */
unsigned char week_cnt;         /* Count of week                        */
unsigned char hour_cnt;         /* Count of hour                        */
unsigned char min_cnt;          /* Count of minute                      */
unsigned char sec_cnt;          /* Count of second                      */

/************************************************************************/
/*      ROM                                                             */
/************************************************************************/
const unsigned char day_max_tbl[12] = {
            0x1F,               /* January   (1 to 31)                  */
            0x1C,               /* February  (1 to 28)                  */
            0x1F,               /* March     (1 to 31)                  */
            0x1E,               /* April     (1 to 30)                  */
            0x1F,               /* May       (1 to 31)                  */
            0x1E,               /* June      (1 to 30)                  */
            0x1F,               /* July      (1 to 31)                  */
            0x1F,               /* August    (1 to 31)                  */
            0x1E,               /* September (1 to 30)                  */
            0x1F,               /* October   (1 to 31)                  */
            0x1E,               /* November  (1 to 30)                  */
            0x1F                /* December  (1 to 31)                  */
};

/************************************************************************/
/*      Pragma                                                          */
/************************************************************************/

/************************************************************************/
/*      Main Program                                                    */
/************************************************************************/
void main(void)
{

    prcr = 0x01;        /* Protect Register */
    /* 00000001B */
    /*        +-------- Enable write to CM0, CM1, CM2, ROCR,        */
    /*                                 PLC0, PCLKR and CCLKR registers */
    /*                     0 : Write protected                      */
    /*                     1 : Write enabled                        */
```

```
/* Technical Update          */
/* TN-16C-119A/J  (Japanese) */
/* TN-16C-119A/EA (English)  */
cm0 = 0x18;          /* System clock control register 0 */
cm0 = 0x10;          /* System clock control register 0 */
/* 00010000B */
/* ||||||++-------- Clock output function select bit            */
/* ||||||                              (M16C/26A, 29 Group)      */
/* ||||||              00 : I/O port P90                         */
/* ||||||           Reserved bits    (M16C/26, 28 Group)        */
/* ||||||                                                       */
/* |||||+---------- WAIT peripheral function clock stop bit     */
/* |||||               0 : Do not stop peripheral function     */
/* |||||                                     clock in wait mode */
/* |||||               1 : Stop peripheral function clock      */
/* |||||                                           in wait mode */
/* ||||+----------- XCIN-XCOUT drive capacity select bit        */
/* ||||                0 : LOW                                  */
/* ||||                1 : HIGH                                 */
/* |||+------------ Port XC select bit                         */
/* |||                 0 : I/O port P86, P87                    */
/* |||                 1 : XCIN-XCOUT generation function       */
/* ||+------------- Main clock stop bit                        */
/* ||                  0 : On                                   */
/* ||                  1 : Off                                  */
/* |+-------------- Main clock division select bit 0           */
/* |                   0 : CM16 and CM17 valid                  */
/* |                         Initial value of CM1 :             */
/* |                                      No division mode     */
/* |                   1 : Division by 8 mode                   */
/* +--------------- System clock select bit                    */
/*                     0 : Main clock, PLL clock,               */
/*                                   or ring oscillator clock */
/*                     1 : Sub-clock                            */

prcr = 0x00;         /* Protect Register */
/* 00000000B */
/*        +-------- Enable write to CM0, CM1, CM2, ROCR,        */
/*                                 PLC0, PCLKR and CCLKR registers */
/*                     0 : Write protected                      */
/*                     1 : Write enabled                        */

sec_cnt = 0x00;      /* Intialization of count of second        */
min_cnt = 0x00;      /* Intialization of count of minute        */
hour_cnt = 0x00;     /* Intialization of count of hour          */
week_cnt = 0x00;     /* Intialization of count of week          */
```

```
day_cnt = 0x01;     /* Intialization of count of day            */
month_cnt = 0x01;   /* Intialization of count of month          */
year_cnt = 0x0001;  /* Intialization of count of year           */

tabsr = 0x00;       /* Count start flag */
/* 00000000B */
/*        +-------- Timer A0 count start flag                   */
/*                     0 : Stops counting                       */
/*                     1 : Starts counting                      */

ta0mr = 0xC0;       /* Timer A0 mode register */
/* 11000000B */
/* || |||++-------- Operation mode select bit                   */
/* || |||              00 : Timer mode                         */
/* || ||+---------- Pulse output function select bit           */
/* || ||               0 : Pulse is not output                  */
/* || ||                   (TAiOUT pin is a normal port pin)    */
/* || ||               1 : Pulse is output                      */
/* || ||                   (TAiOUT pin is a pulse output pin)   */
/* || ++----------- Gate function select bit                    */
/* ||                  00 : Gate function not available         */
/* ||                       (TAiIN pin functions as I/O port)   */
/* ++--------------- Count source select bit                    */
/*                     11 : fC32                                */

ta0 = 0x3FF;        /* Timer A0 register (400-1)h */

ta0ic = 0x00;       /* Timer A0 Interrupt control register */
/* 00000000B */
/*      +++-------- Interrupt priority level select bit         */
/*                     000 : Level 0                            */

/* Setting "I" flag permitting */
asm("FSET I");      /* "I" flag */

tabsr = 0x01;       /* Count start flag */
/* 00000001B */
/*        +-------- Timer A0 count start flag                   */
/*                     0 : Stops counting                       */
/*                     1 : Starts counting                      */

while(1){
        /* Loop */
      if(ir_ta0ic == 1){
           /* When the interrupt request bit is "1" */
        ir_ta0ic = 0;       /* Interrupt request bit clear */
        time_set();         /* Time_set Function          */
```

```
        }
    }

}


void time_set(void)
{

    sec_cnt++;                          /* Count of second addition */

    if(sec_cnt > SEC_MAX){
        min_cnt++;                      /* Count of minute addition */
        sec_cnt = 0;                    /* Count of second clear    */

        if(min_cnt > MIN_MAX){
            hour_cnt++;                 /* Count of hour addition */
            min_cnt = 0;                /* Count of minute clear  */

            if(hour_cnt > HOUR_MAX){
                day_cnt++;              /* Count of day addition  */
                week_cnt++;             /* Count of week addition */
                hour_cnt = 0;           /* Count of hour clear    */

                date_set();             /* date_set Function      */

                if(week_cnt > WEEK_MAX){
                    week_cnt = 0;       /* Count of week clear */

                }
            }
        }
    }
}


void date_set(void)
{

    unsigned char leap_year_flg;        /* Set flag at leap year */
                                        /* 0: Not set            */
                                        /* 1: Set                */

    if( day_cnt > day_max_tbl[month_cnt-1] ){
                /* When you exceed the maximum days */
        leap_year_flg = leap_year_check();  /* leap_year_check Function */
```

```
        if(leap_year_flg == 0){
                /* When it is not a leap year */
            month_cnt++;                    /* Count of month addition */
            day_cnt = 0x01;                 /* Count of day clear      */
        }

        leap_year_flg = 0;                  /* Set flag at leap year clear */

        if(month_cnt > MONTH_MAX){
                /* When you exceed the maximum months */
            year_cnt++;                     /* Count of year addition */
            month_cnt = 0x01;               /* Count of month clear   */

            if(year_cnt > YEAR_MAX){
                /* When you exceed the maximum years */
                year_cnt = 0x0001;          /* Count of year clear */
            }
        }
    }
}


unsigned char leap_year_check(void)
{

    unsigned char rtn;      /* Return value */
                            /*            0: Not set leap year */
                            /*            1: Set leap year     */


    rtn= 0;                         /* Return value initialization */

    if( (month_cnt == 0x02)&&(day_cnt == 0x1D) ){
                /* Comparison at leap year */

        /*************************************************************/
        /* The judgment makes year when the age can be fourthly divided */
        /* at the Christian era a leap year at the leap year.        */
        /* However, year that cannot be divided by 400 is not made a    */
        /* leap year among as many as 100 years that can be fourthly    */
        /* divided at the Christian era the age, can be divided.        */
        /* (Gregorian)                                               */
        /*************************************************************/
        if( (year_cnt%4) == 0 ){
                /* When years can be fourthly divided */
            rtn = 1;                /* Set leap year */
```

```
            if( (year_cnt%100) == 0 ){
                /* When years can be divided by 100 */
                rtn = 0;            /* Not set leap year */

                if( (year_cnt%400) == 0 ){
                /* When years can be divided by 400 */
                    rtn = 1;        /* Set leap year */
                }
            }
        }
    }

    return(rtn);

}
```

6.3 参考プログラム例（M16C/28, 29 グループ）

M16C/28 グループで動作させた場合のプログラムを例として示します。

6.3.1 参考プログラム例（M16C/28 グループ）

```
/************************************************************************/
/*                                                                      */
/*      M16C/28 Program Collection                                      */
/*                                                                      */
/*      FILE NAME   :   rjj05b0849_src.c                                */
/*      CPU         :   M16C/28 Group                                   */
/*      FUNCTION    :   Real Time Clock function                        */
/*      HISTORY     :   2005.06.30  Ver 1.00                            */
/*                                                                      */
/*      Copyright(C)2005, Renesas Technology Corp.                      */
/*      Copyright(C)2005, Renesas Solutions Corp.                       */
/*      All rights reserved.                                            */
/*                                                                      */
/************************************************************************/

/************************************************************************/
/*      Include                                                         */
/************************************************************************/
#include    "sfr28.h"

/************************************************************************/
/*      Prototype declaration                                           */
/************************************************************************/
void main(void);
void time_set(void);
void date_set(void);
unsigned char leap_year_check(void);

/************************************************************************/
/*      DEFINE                                                          */
/************************************************************************/
#define YEAR_MAX    0x270F      /* The maximum value at year   ( 1 to 9999 ) */
#define MONTH_MAX   0x0C        /* The maximum values at month ( 1 to 12 )   */
#define WEEK_MAX    0x06        /* The maximum value on a day of the week    */
                                /*       0: Sunday                           */
                                /*       1: Monday                           */
                                /*       2: Tuesday                          */
                                /*       3: Wednesday                        */
                                /*       4: Thursday                         */
```

```
                                /*       5: Friday                          */
                                /*       6: Saturday                        */
#define HOUR_MAX    0x17        /* The maximum value at hour   ( 0 to 23 )  */
#define MIN_MAX     0x3B        /* The maximum value at minute ( 0 to 59 )  */
#define SEC_MAX     0x3B        /* The maximum value at second ( 0 to 59 )  */

/************************************************************************/
/*      RAM                                                             */
/************************************************************************/
unsigned int  year_cnt;         /* Count at year                        */
unsigned char month_cnt;        /* Count of month                       */
unsigned char day_cnt;          /* Count of day                         */
unsigned char week_cnt;         /* Count of week                        */
unsigned char hour_cnt;         /* Count of hour                        */
unsigned char min_cnt;          /* Count of minute                      */
unsigned char sec_cnt;          /* Count of second                      */

/************************************************************************/
/*      ROM                                                             */
/************************************************************************/
const unsigned char day_max_tbl[12] = {
            0x1F,               /* January   (1 to 31)                  */
            0x1C,               /* February  (1 to 28)                  */
            0x1F,               /* March     (1 to 31)                  */
            0x1E,               /* April     (1 to 30)                  */
            0x1F,               /* May       (1 to 31)                  */
            0x1E,               /* June      (1 to 30)                  */
            0x1F,               /* July      (1 to 31)                  */
            0x1F,               /* August    (1 to 31)                  */
            0x1E,               /* September (1 to 30)                  */
            0x1F,               /* October   (1 to 31)                  */
            0x1E,               /* November  (1 to 30)                  */
            0x1F                /* December  (1 to 31)                  */
};

/************************************************************************/
/*      Pragma                                                          */
/************************************************************************/

/************************************************************************/
/*      Main Program                                                    */
/************************************************************************/
void main(void)
{

    prcr = 0x01;        /* Protect Register */
    /* 00000001B */
```

```
/*        +-------- Enable write to CM0, CM1, CM2, ROCR,         */
/*                            PLC0, PCLKR and CCLKR registers */
/*                    0 : Write protected                      */
/*                    1 : Write enabled                        */

/* Technical Update          */
/* TN-16C-119A/J  (Japanese) */
/* TN-16C-119A/EA (English)  */
cm0 = 0x18;         /* System clock control register 0 */
cm0 = 0x10;         /* System clock control register 0 */
/* 00010000B */
/* ||||||++-------- Clock output function select bit           */
/* ||||||                          (M16C/26A, 29 Group)       */
/* ||||||              00 : I/O port P90                      */
/* ||||||           Reserved bits    (M16C/26, 28 Group)      */
/* ||||||                                                     */
/* |||||+---------- WAIT peripheral function clock stop bit    */
/* |||||               0 : Do not stop peripheral function    */
/* |||||                                    clock in wait mode */
/* |||||               1 : Stop peripheral function clock     */
/* |||||                                          in wait mode */
/* ||||+----------- XCIN-XCOUT drive capacity select bit       */
/* ||||                0 : LOW                                */
/* ||||                1 : HIGH                               */
/* |||+------------ Port XC select bit                        */
/* |||                 0 : I/O port P86, P87                  */
/* |||                 1 : XCIN-XCOUT generation function     */
/* ||+------------- Main clock stop bit                       */
/* ||                  0 : On                                 */
/* ||                  1 : Off                                */
/* |+-------------- Main clock division select bit 0          */
/* |                   0 : CM16 and CM17 valid                */
/* |                         Initial value of CM1 :           */
/* |                                   No division mode     */
/* |                   1 : Division by 8 mode                 */
/* +--------------- System clock select bit                   */
/*                     0 : Main clock, PLL clock,             */
/*                                   or ring oscillator clock */
/*                     1 : Sub-clock                          */

prcr = 0x00;        /* Protect Register */
/* 00000000B */
/*        +-------- Enable write to CM0, CM1, CM2, ROCR,         */
/*                            PLC0, PCLKR and CCLKR registers */
/*                    0 : Write protected                      */
/*                    1 : Write enabled                        */
```

```
prcr = 0x04;        /* Protect Register */
/* 00000100B */
/*      +---------- Enable write to PD9, PACR, S4C              */
/*                                        and NDDR registers */
/*                    0: Write protected                       */
/*                    1: Write enabled                         */

pacr = 0x02;        /* Pin assignment control register */
/* 00000010B */
/*      +++-------- Pin enabling bit                           */
/*                    001 : 42 pin (M16C/26A Group)            */
/*                    010 : 64 pin (M16C/28, 29 Group)         */
/*                    011 : 80 pin (M16C/28, 29 Group)         */
/*                    100 : 48 pin (M16C/26A Group)            */
/*                  All other values are reserved. Do not use. */

ifsr2a = 0x01;      /* Interrupt request cause select register 2 */
/* 00000001B */
/*        +--------- Must be set to "1". (M16C/26A, 28 Group)  */
/*                   Must be set to "0". (M16C/29 Group)       */
/*                   Set this 0th bit before you enable        */
/*                                    interrupt after resetting. */

sec_cnt = 0x00;     /* Intialization of count of second         */
min_cnt = 0x00;     /* Intialization of count of minute         */
hour_cnt = 0x00;    /* Intialization of count of hour           */
week_cnt = 0x00;    /* Intialization of count of week           */
day_cnt = 0x01;     /* Intialization of count of day            */
month_cnt = 0x01;   /* Intialization of count of month          */
year_cnt = 0x0001;  /* Intialization of count of year           */

tabsr = 0x00;       /* Count start flag */
/* 00000000B */
/*        +--------- Timer A0 count start flag                  */
/*                    0 : Stops counting                       */
/*                    1 : Starts counting                      */

ta0mr = 0xC0;       /* Timer A0 mode register */
/* 11000000B */
/* || |||++-------- Operation mode select bit                  */
/* || |||             00 : Timer mode                        */
/* || ||+---------- Pulse output function select bit          */
/* || ||             0 : Pulse is not output                  */
/* || ||                 (TAiOUT pin is a normal port pin)    */
/* || ||             1 : Pulse is output                      */
/* || ||                 (TAiOUT pin is a pulse output pin)   */
/* || ++----------- Gate function select bit                  */
```

```
    /* ||                  00 : Gate function not available        */
    /* ||                       (TAiIN pin functions as I/O port)  */
    /* ++--------------- Count source select bit                   */
    /*                     11 : fC32                               */

    ta0 = 0x3FF;        /* Timer A0 register (400-1)h */

    ta0ic = 0x00;       /* Timer A0 Interrupt control register */
    /* 00000000B */
    /*      +++-------- Interrupt priority level select bit       */
    /*                     000 : Level 0                           */

    /* Setting "I" flag permitting */
    asm("FSET I");      /* "I" flag */

    tabsr = 0x01;       /* Count start flag */
    /* 00000001B */
    /*        +-------- Timer A0 count start flag                  */
    /*                     0 : Stops counting                      */
    /*                     1 : Starts counting                     */

    while(1){
          /* Loop */
        if(ir_ta0ic == 1){
            /* When the interrupt request bit is "1" */
          ir_ta0ic = 0;       /* Interrupt request bit clear */
          time_set();         /* Time_set Function          */
        }
    }

}


void time_set(void)
{

    sec_cnt++;                          /* Count of second addition */

    if(sec_cnt > SEC_MAX){
        min_cnt++;                      /* Count of minute addition */
        sec_cnt = 0;                    /* Count of second clear    */

        if(min_cnt > MIN_MAX){
            hour_cnt++;                 /* Count of hour addition */
            min_cnt = 0;                /* Count of minute clear  */

            if(hour_cnt > HOUR_MAX){
```

```
                day_cnt++;              /* Count of day addition  */
                week_cnt++;             /* Count of week addition */
                hour_cnt = 0;           /* Count of hour clear    */

                date_set();             /* date_set Function      */

                if(week_cnt > WEEK_MAX){
                    week_cnt = 0;       /* Count of week clear */

                }
            }
        }
    }
}


void date_set(void)
{

    unsigned char leap_year_flg;        /* Set flag at leap year */
                                        /* 0: Not set            */
                                        /* 1: Set                */

    if( day_cnt > day_max_tbl[month_cnt-1] ){
                /* When you exceed the maximum days */
        leap_year_flg = leap_year_check();  /* leap_year_check Function */

        if(leap_year_flg == 0){
                /* When it is not a leap year */
            month_cnt++;                    /* Count of month addition */
            day_cnt = 0x01;                 /* Count of day clear      */
        }

        leap_year_flg = 0;                  /* Set flag at leap year clear */

        if(month_cnt > MONTH_MAX){
                /* When you exceed the maximum months */
            year_cnt++;                     /* Count of year addition */
            month_cnt = 0x01;               /* Count of month clear   */

            if(year_cnt > YEAR_MAX){
                /* When you exceed the maximum years */
                year_cnt = 0x0001;          /* Count of year clear */
            }
        }
    }
}
```

```
unsigned char leap_year_check(void)
{

    unsigned char rtn;      /* Return value */
                            /*            0: Not set leap year */
                            /*            1: Set leap year     */


    rtn= 0;                         /* Return value initialization */

    if( (month_cnt == 0x02)&&(day_cnt == 0x1D) ){
                /* Comparison at leap year */

        /*****************************************************************/
        /* The judgment makes year when the age can be fourthly divided */
        /* at the Christian era a leap year at the leap year.           */
        /* However, year that cannot be divided by 400 is not made a    */
        /* leap year among as many as 100 years that can be fourthly    */
        /* divided at the Christian era the age, can be divided.        */
        /* (Gregorian)                                                  */
        /*****************************************************************/
        if( (year_cnt%4) == 0 ){
                /* When years can be fourthly divided */
            rtn = 1;                /* Set leap year */

            if( (year_cnt%100) == 0 ){
                /* When years can be divided by 100 */
                rtn = 0;            /* Not set leap year */

                if( (year_cnt%400) == 0 ){
                /* When years can be divided by 400 */
                    rtn = 1;        /* Set leap year */
                }
            }
        }
    }

    return(rtn);

}
```

7. 参考ドキュメント

ハードウェアマニュアル

M16C/26A, 26, 28, 29 グループハードウェアマニュアル Rev.1.0
（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。）

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 テクノロジホームページから入手してください。）

8. ホームページとサポート窓口

ルネサス テクノロジ M16C ホームページ

http://japan.renesas.com/m16c

M16C ファミリ MCU 技術サポート窓口

E-mail: csc@renesas.com

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2005.07.01	-	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。