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

MESC TECHNICAL NEWS

No. M16C-40-9912

M16C/80 グループ　10 進演算命令(DSUB、DSBB、DADD、DADC)使用に関する注意事項

1.　対象品種

・M16C/80 グループ

2.　注意事項

10 進演算命令(DSUB、DSBB、DADD、DADC)実行中に DMA 転送が発生すると、演算結果が正しく出力されません。

3.　対　策

DMA を使用する場合、10 進演算命令は使用しないでください。
16 進 -10 進変換、10 進 -16 進変換のプログラム例を次ページ以降に添付します。
10 進演算を行う場合は 16 進で演算を行い、10 進に変換してください。

16 進 -10 進変換プログラム(2 バイト)

```
.SECTION    PROG,       CODE
.GLB  HEXtoBCD_2byte
;===============================================================================
;       表題：ＨＥＸコードからＢＣＤコードへの変換
;       概要：ＨＥＸ２バイトコードをＢＣＤ４バイトコードに変換
;       入力：----------------------> 出力：
;       R0  ()                      R0  (ＢＣＤコード下位４桁)
;       R1  (ＨＥＸコード入力)      R1  (不定)
;       R2  ()                      R2  (ＢＣＤコード上位１桁)
;       R3  ()                      R3  (不定)
;       A0  ()                      A0  (未使用)
;       A1  ()                      A1  (未使用)
;       スタック使用量：なし
;       注釈：
;===============================================================================
HEXtoBCD_2byte:                             ; HEXtoBCD_2byte{
        mov.w #0,R0                         ; ＢＣＤエリアの初期化
        mov.w #0,R2                         ;
HEXtoBCD_2byte5:                            ; ５桁目カウント
        mov.w R1,R3                         ; 引く前に R1 を R3 に退避
        sub.w  #10000,      R1              ; R1 = R1 - 10000
        jltu    HEXtoBCD_2byte4             ; ４桁目以降へ
        inc.w   R2                          ; ５桁目 ++
        jmp             HEXtoBCD_2byte5
HEXtoBCD_2byte4:                            ; ４桁目カウント
        mov.w R3,    R1                     ; R1 復帰
HEXtoBCD_2byte4Loop:
        mov.w R1,    R3                     ; R1 退避
        sub.w  #1000,R1                     ; R1 = R1 - 1000
        jltu    HEXtoBCD_2byte3             ; if( R1 < 1000 ) 3 桁目へ
        inc.b   R0H                         ; ４桁目 ++
        jmp             HEXtoBCD_2byte4Loop
HEXtoBCD_2byte3:                            ; ３桁目カウント
        mov.w R3,    R1                     ; R1 復帰
        shl.b   #4,     R0H                 ; R0H << 4 ( ４桁目をシフト)
HEXtoBCD_2byte3Loop:
        mov.w R1,    R3                     ; R1 退避
        sub.w  #100,R1
        jltu    HEX_toBCD_2byte2            ; if( R1 < 100 ) 2 桁目へ
        inc.b   R0H                         ; ３桁目 ++
        jmp    HEXtoBCD_2byte3Loop
HEX_toBCD_2byte2:                           ; 2 桁目カウント
        mov.w R3,    R1                     ; R1 復帰
HEXtoBCD_2byte2Loop:
        mov.w R1,    R3                     ; R1 退避
        sub.w  #10, R1                      ; R1 = R1 - 10
        jltu    HEX_toBCD_2byte1            ; if( R1 < 10 ) 1 桁目へ
        inc.b   R0L                         ; 2 桁目 ++
        jmp    HEXtoBCD_2byte2Loop
HEX_toBCD_2byte1:
        shl.b   #4,R0L                      ; R0L << 4 ( ２桁目をシフト)
        add.w  R3,R0                        ; 残り１桁を加算
rts                                         ; }
.END
```

10 進 -16 進変換プログラム(2 バイト)

```
.SECTION    PROG,       CODE
.GLB  BCDtoHEX_2byte
;=============================================================================
;       表題：ＢＣＤコードからＨＥＸコードへの変換
;       概要：ＢＣＤ２バイトコードをＨＥＸ２バイトコードに変換
;       入力：----------------------> 出力：
;       R0 (ＢＣＤコード入力 )      R0 (不定)
;       R1 ()                       R1 (ループ変数)
;       R2 ()                       R2 (ＨＥＸコード出力)
;       R3 ()                       R3 (未使用)
;       A0 ()                       A0 (未使用)
;       A1 ()                       A1 (未使用)
;       スタック使用量：なし
;       注釈：
;=============================================================================
BCDtoHEX_2byte:                                 ; BCDtoHEX_2byte{
        mov.w #0,     R2                        ;       R2 = 0 ( ＨＥＸエリアの初期化 )
        mov.b #16,    R1H                       ;       R1H = 16 ( シフトループ回数設定 )

BCDtoHex_2byte_Shift:                           ;       while( R1H ){
        shl.w   #-1,    R0                      ;               R0 を右シフト 剰余 --> C フラグ
        rorc.w R2                               ;               C フラグ付き右回転

        mov.b #4,               R1L             ;               R1L = 4 ( ローテイトループ回数設定 )
BCDtoHex_2byte_Rot:                             ;               while( R1L ){
        btst    3,      R0L                     ;                       補正チェック
        jnc             BCDtoHEX_2byte_NoAdj    ;                       if ( b3 == 1 ){
        sub.w  #3,      R0                      ;                               補正する
BCDtoHEX_2byte_NoAdj:                           ;                       }
        rot.w   #-4, R0                         ;                       4bit 分ローテイトする
        adjnz.b         #-1, R1L, BCDtoHex_2byte_Rot ;                  R1L--
                                                ;               }
        adjnz.b         #-1, R1H, BCDtoHex_2byte_Shift ;        R1H--
                                                ;       }
rts                                             ; }/* RTS */

.END
```

10 進 -16 進変換プログラム(1 バイト)

```
.SECTION    PROG,       CODE
.GLB  BCDtoHEX_1byte
;=============================================================================
;       表題：ＢＣＤコードからＨＥＸコードへの変換
;       概要：ＢＣＤ１バイトコードをＨＥＸ１バイトコードに変換
;       入力：----------------------> 出力：
;       R0L ()                      R0L (ＨＥＸコード)
;       R0H (ＢＣＤコード)          R0H (不定)
;       R1L ()                      R1L (不定)
;       R1H ()                      R1H (未使用)
;       R2  ()                      R2  (未使用)
;       R3  ()                      R3  (未使用)
;       A0  ()                      A0  (未使用)
;       A1  ()                      A1  (未使用)
;       スタック使用量：なし
;       注釈：
;=============================================================================
BCDtoHEX_1byte:
        mov.b #0,R0L                                ; ＨＥＸエリアの初期化
        mov.b #8,R1L                                ; ループ回数の設定
BCDtoHEX_1byte_10:
        shl.b   #-1,R0H                             ; 最上位ビットのシフト
        rorc.b  R0L                                 ;
        btst    3,R0H                               ;
        jeq     BCDtoHEX_1byte_20                   ;
        sub.b   #3,R0H                              ;
BCDtoHEX_1byte_20:
        adjnz.b         #-1,R1L,BCDtoHEX_1byte_10   ; --> 次のＢＣＤ桁実行へ
        rts

.END
```

16 進 -10 進変換プログラム(1 バイト)

```
.SECTION    PROG,       CODE
.GLB  HEXtoBCD_1byte
;===============================================================================
;       表題：ＨＥＸコードからＢＣＤコードへの変換
;       概要：ＨＥＸ１バイトコードをＢＣＤ２バイトコードに変換
;       入力：----------------------> 出力：
;       R0  ()                      R0  (ＢＣＤコード)
;       R1L (ＨＥＸコード)          R1L (不定)
;       R1H ()                      R1H (未使用)
;       R2  ()                      R2  (不定)
;       R3  ()                      R3  (未使用)
;       A0  ()                      A0  (未使用)
;       A1  ()                      A1  (未使用)
;       スタック使用量：なし
;       注釈：
;===============================================================================
HEXtoBCD_1byte:
        mov.w #0,R0                     ; ＢＣＤエリアの初期化

HEXtoBCD_1byte_10:
        mov.w R1,R2                     ; ＨＥＸデータ退避
        sub.b  #0Ah,R1L                 ; 桁上がりチェック
        jltu     HEXtoBCD_1byte_END     ; 桁上がりがなければ終了
        add.b  #10H,R0L                 ; 桁上がり分を BCD の上位桁に加算
        cmp.b #0A0H,R0L                 ; ２桁目の桁上がりチェック
        jne     HEXtoBCD_1byte_10       ; 090H 以下ならジャンプ
        add.b  #01H,R0H                 ; ２桁目の桁上がり分を BCD の最上位桁に加算
        mov.b #0H,R0L                   ; BCD の下位をリセット
        jmp     HEXtoBCD_1byte_10       ; 桁上がりがなくなるまで繰り返す。

HEXtoBCD_1byte_END:
        mov.w R2,R1                     ; ＨＥＸデータ復帰
        add.b  R1L,R0L                  ; BCD の下位桁を設定
        rts                             ;

.END
```