

COMENTARIO TÉCNICO

Buceando en el HC908.....



Por Ing. Daniel Di Lella

Dedicated Distributor Field Application Engineer

For Freescale Semiconductors Products & Technical Consult

Dto. Técnico Electrocomponentes S.A.

fae@electrocomponentes.com

dilella@arnet.com.ar

“Como sacarle el mayor provecho a la familia HC908APxx”

2da. y última parte:

Continuando con el artículo anterior, en donde se detallaron los errores en la máscara de la familia de microcontroladores HC908APxx y como solucionarlos, abordaremos en este los errores en las sub rutinas internas en ROM de esta familia.

Rutinas Residentes en ROM.

La familia HC908AP contiene una serie de rutinas internas implementadas en ROM, estas rutinas nos ayudan a manejar en forma más sencilla los procesos de borrado y programación de la memoria FLASH tanto en el “modo monitor”, utilizado para programar / borrar al MCU, como en el “modo usuario” para que nuestros programas tengan la posibilidad de almacenar datos “no volátiles” como si utilizáramos una EEPROM externa. En numerosos artículos se han explicado las ventajas de ello y lo sencillo de su uso.

De todas las rutinas existentes en la ROM de los AP, hay 4 de ellas que presentan errores de código que generan el no-funcionamiento de ellas o bien el borrado no buscado de la zona de vectores (más precisamente el Reset Vector) y ellas son:

- ERARNGE
- MON_ERARNGE
- EE_WRITE
- EE_READ

Para solucionar el problema, sugiero leer atentamente las fe de erratas de freescale anteriormente comentadas. Sin embargo les adjunto dos rutinas que deben ser copiadas en flash (agregadas a vuestros programas) y que permiten realizar borrados muy confiables de páginas (512 bytes por página) Flash y el borrado en “masa” (Mass Erase) de toda la memoria Flash del microcontrolador.

Estas rutinas tienen la virtud de “auto copiarse” en la memoria RAM en forma automática cuando se las invoca y además ocupan poco espacio (solo 60 bytes) en la misma RAM una vez copiadas allí.

```

base 10T                ;Base Decimal por Default
include 'ap32_registers.inc' ;Equates grales. p/ AP32

*****
* EQUATES *
*****

RAMSPACE      EQU $0060          ;Czo de la RAM uso GRAB/BORR. FLASH
DATSTRC       EQU RAMSPACE+8    ;RAM para uso del "MONITOR ROM"
Q_RAM_Blk_Erase EQU $0080      ;Czo RAM para programa LIMPIEZA GRAL. FLASH
RAMUSER       EQU $00C0          ;Comienzo de la RAM p/ usuario
RAMEND        EQU $0860          ;Fin de la RAM + 1 en el AP32 / AP64
ROMSTART      EQU $7860          ;Czo de la FLASH para PROGRAMA
PRGRNGE      EQU $FC34          ;Sub-Rutina en ROM escritura datos en FLASH

*****
* EQUATES P/ BORRADO DE FLASH *
*****

b_ERASE       EQU 2              ;Bits control bomba de carga
b_HVEN        EQU 8
b_MASS        EQU 6
MORVALUE      EQU %11111111     ;Valor standard del MOR

BLOCKSIZE     EQU $200
TESTCOUNT    EQU $20
FLASHSTART    EQU $860
RAM_BEGIN     EQU $60
RAM_END       EQU $85F

*****
* VARIABLES EN RAM *
*****

ORG RAMSPACE   ; Comienzo RAM para AP32
               ; reservada hasta DATSTRC P/ uso
               ; rutinas ROM prog/erase FLASH.

ORG DATSTRC    ; RAM p/ uso en prog/erase flash

CPUSPD        RMB 1              ; CPU speed -> 4 x Fop aprox. (FOSC)
DATASIZE      RMB 1              ; N° de Bytes de datos a grabar.
STARTADDR     RMB 2              ; Start Address p/ R o W un rango
DATA          RMB 1              ; DATOS en RAM transitorios

V_FLASH_ADDR  RMB 2              ; Address czo. página a borrar
               ; cargar con $FFFF p/ MASS ERASE

```

```

;-----
;-----
*****
* MASS_ERASE - Rutina de borrado GRAL. de la FLASH en los HC908APxx *
* Toda la rutina está en FLASH y luego se copia a RAM p/ ejecutarla *
* desde allí. (FBUS = 2,4576 MHZ) *
* *
* VARIABLES INVOLUCRADAS: *
* *
* V_FLASH_ADDR -> Dirección de czo. página a borrar (512 bytes) *
* En este caso se debe cargar con $FFFF *
* *
* ATENCION: Reservar 60 BYTES en la RAM a partir de "Q_RAM_Blк_Erase" *
*****

```

MASS_ERASE

```

    lda #$FF
    sta FLBPR      ; Without flash protection !!

```

*--- erase block -----

erase1_M:

```

    pshh          ; "H:X" content will be corrupted in the jump
    pshx          ;

```

```

    jsr FLASH_ERASE_M

```

```

    pulx          ;
    pulh

```

```

ICPCLEAN_M  NOP          ; LOOP infinito hasta P.O.R del MCU
            BRA ICPCLEAN_M      ;

```

```

    RTS          ;

```

* Block Erase *

FLASH_ERASE_M:

```

    bsr BlkErase2RAM_M    ; copy block erase routine to RAM
    ldhx V_FLASH_ADDR    ; load H:X with the block address
    lda #b_MASS          ; MUST load Acc with b_MASS (ERASE+MASS=1)
    jsr Q_RAM_Blк_Erase  ; execute block erase in RAM
    rts

```

*-----

BlkErase2RAM_M:

```

    ldhx #Blk_Erase_Len_M ; get blk erase routine length
                        ; NB: Assume "Blk_Erase_Len" is one byte long

```

```

BE2RAM1_M:
    lda (Block_Erase-1),x      ; load from FLASH
    sta {Q_RAM_Blck_Erase-1},x ; copy to RAM
    dbnzx BE2RAM1_M           ; NB: Assume "Blk_Erase_Len" is one byte long
                                ; need modification if length over 1 byte
    rts

```

```

Block_Erase_M:
    sta FLCR                   ; set ERASE bit
    sta ,x                     ; write any data to block
    bsr Dly_5us_M
    lda FLCR
    ora #b_HVEN
    sta FLCR                   ; set HVEN bit

```

* ----- *

```

    ldx #200                   ;(2)

```

```

Blk_Erase_Time_M:
    bsr Dly_1ms_M              ;[14]
    dbnzx Blk_Erase_Time_M     ;(3)
    ldhx #FLCR
    lda #%00001000             ; clear ERASE bit
    sta ,x
    bsr Dly_100us_M
    clr ,x                     ; clear HVEN bit
    rts

```

Blk_Erase_Exit_M:

* ----- *

```

Dly_5us_M:
    ;For 2,5 MHz bus
    ; [4] cycles
    lda #2                     ; [2]
    dbnza $                     ; [3]
    rts                         ; [4]

```

```

Dly_100us_M:
    ; [4] cycles
    lda #50                    ; [2]
    dbnza $                     ; [3]
    rts                         ; [4]

```

```

Dly_1ms_M:
    ; [4] cycles
    lda #128                   ; [2]
    mov PORTB,PORTB            ; dummy for 15 bus clk
    mov PORTB,PORTB            ;
    mov PORTB,PORTB            ;
    dbnza $                     ; [3]
    rts                         ; [4]

```

Dly_5us_Exit_M:

```

Blk_Erase_Len_M equ {Dly_5us_Exit_M - Block_Erase_M}

```

```

;-----
;-----

```

```

;-----
;-----
*****
* PAGE_ERASE - Rutina de borrado de Página en los HC908APxx          *
* Toda la rutina está en FLASH y luego se copia a RAM p/ ejecutarla *
* desde allí. (FBUS = 2,4576 MHZ)                                   *
*                                                                     *
* VARIABLES INVOLUCRADAS:                                          *
*                                                                     *
* V_FLASH_ADDR -> Dirección de czo. página a borrar (512 bytes)   *
*                                                                     *
* ATENCION: Reservar 60 BYTES en la RAM a partir de "Q_RAM_Blz_Erase" *
*****

```

PAGE_ERASE

```

        lda #$FE
        sta FLBPR          ; protect vector table only

```

*--- erase block -----

erase1:

```

        pshh              ; "H:X" content will be corrupted in the jump
        pshx              ;
        jsr FLASH_ERASE

        pulx              ;
        pulh              ;

        RTS              ;

```

```

*****
* Block Erase          *
*****

```

FLASH_ERASE:

```

        bsr BlkErase2RAM    ; copy block erase routine to RAM
        ldhx V_FLASH_ADDR  ; load H:X with the block address
        lda #b_ERASE       ; MUST load Acc with b_ERASE
        jsr Q_RAM_Blz_Erase ; execute block erase in RAM
        rts

```

*-----

BlkErase2RAM:

```

        ldhx #Blk_Erase_Len ; get blk erase routine length
                                ; NB: Assume "Blk_Erase_Len" is one byte long

```

BE2RAM1:

```

        lda (Block_Erase-1),x ; load from FLASH
        sta {Q_RAM_Blz_Erase-1},x ; copy to RAM
        dbnzx BE2RAM1         ; NB: Assume "Blk_Erase_Len" is one byte long
                                ; need modification if length over 1 byte
        rts

```

```

Block_Erase:
    sta FLCR          ; set ERASE bit
    sta ,x           ; write any data to block
    bsr Dly_5us
    lda FLCR
    ora #b_HVEN
    sta FLCR          ; set HVEN bit

```

* ----- *

```

Blk_Erase_Time:
    ldx #20           ;(2)
    bsr Dly_1ms      ;[14]
    dbnzx Blk_Erase_Time ;(3)
    ldhx #FLCR
    lda #%00001000    ; clear ERASE bit
    sta ,x
    bsr Dly_5us
    clr ,x            ; clear HVEN bit
    rts

```

Blk_Erase_Exit:

* ----- *

```

Dly_5us:
    ;For 2,5 MHz bus
    ; [4] cycles
    lda #2           ; [2]
    dbnza $          ; [3]
    rts              ; [4]

Dly_1ms:
    ; [4] cycles
    lda #128        ; [2]
    mov PORTB,PORTB ; dummy for 15 bus clk
    mov PORTB,PORTB ;
    mov PORTB,PORTB ;
    dbnza $         ; [3]
    rts             ; [4]

```

Dly_5us_Exit:

Blk_Erase_Len equ {Dly_5us_Exit - Block_Erase}

Bien, ahora veremos en un par ejemplos como utilizarlas

```

*****
* Borrado de una página cualquier de la Flash *
* Cada página está formada por 512 bytes      *
* Para borrar una página se debe “apuntar” al  *
* comienzo de la misma. Por ejemplo:          *
* $0860 , $0A60, $0C60, $xx60.....           *
*****

```

```

CLEAN_PAGE!    STHX TEMPHX                ;Resguardo valor H:X en RAM
                LDHX #PAGINA_A_BORRAR    ;H:X – Dir. Czo de la página a borrar!!
                STHX V_FLASH_ADDR        ;la que usted elija .....

                SEI                        ;Deshabilito INT'S antes de borrar !!

                JSR PAGE_ERASE            ;Voy a la rutina de borrado página (512 bytes)

                CLI                        ;Habilito las INT'S luego de borrar !!

                END

```

```

*****
* Borrado gral. (Mass Erase) de la memoria Flash *
* Aquí para borrar toda la memoria Flash solo se debe *
* apuntar al final de la misma y listo!!          *
*****

```

```

CLEAN_ALL!    LDHX #$FFFF                ; V_FLASH_ADDR <-- $FFFF
                STHX V_FLASH_ADDR        ;

                JSR MASS_ERASE            ; Rutina MASS ERASE p/ AP32

```

En los próximos artículos, veremos más cositas útiles sobre la familia HC908.

Hasta la próxima ¡!!