```
;*************************************************************
;* This program for CMPEN 472, Flash Memory Writing          *
;* By Kyusun Choi,  ID=0000                                  *
;* Date: 04/09/2018                                          *
;* Freescale CodeWarrior,  for the HCS12C128 Program         *
;* Target: Axiom's  CSMBC128 Board, ASCII Monitor Mode       *
;*************************************************************
; parameter declearation section

; export symbols
            XDEF        Entry           ; export 'Entry' symbol
            ABSENTRY    Entry           ; for assembly entry point

; include derivative specific macros
PORTA       EQU     $0000
PORTB       EQU     $0001
DDRA        EQU     $0002
DDRB        EQU     $0003

PPAGE       EQU     $0030           ; Flash page register
FCLKDIV     EQU     $0100           ; Flash clock divider register
FSTAT       EQU     $0105           ; Flash status register
FCMD        EQU     $0106           ; Flash command register

SCIBDH      EQU     $00c8           ; Serial port (SCI) Baud Rate Register H
SCIBDL      EQU     $00c9           ; Serial port (SCI) BAUD Rate Register L
SCICR1      EQU     $00ca           ; Serial port (SCI) Control Register 1
SCICR2      EQU     $00cb           ; Serial port (SCI) Control Register 2
SCISR1      EQU     $00cc           ; Serial port (SCI) Status Register 1
SCIDRL      EQU     $00cf           ; Serial port (SCI) Data Register

CR          equ      $0d            ; carriage return, ASCII 'Return' key
LF          equ      $0a            ; line feed, ASCII 'next line' character


;*****************************************************
; variable/data section
                    ORG       $3000    ; RAMStart defined as $3000

;                   flash data block

flashd:
               DC.B     $30,$30,$30,$30,$CF,$31,$00,$CE
               DC.B     $48,$00,$A6,$30,$07,$13,$A6,$30    ;for HCS12C128 board
               DC.B     $07,$0F,$A6,$30,$07,$0B,$A6,$30
               DC.B     $07,$07,$A7,$A7,$A7,$A7,$A7,$20
               DC.B     $F9,$4F,$CC,$80,$FC,$5A,$CF,$3D


flashc          DC.B     $28                ; data byte count in hex, $28 = 40 bytes
dly1s_r1        DC.W     $0400
dly1s_r2        DC.W     $0400

counter1        DS.B     1

msg1            DC.B     'Well', $00
msg2            DC.B     'Flash memory writing, 40 bytes at $4800', $00
msg3            DC.B     'Done.  Reset and type go 4804 to check', $00   ;for HCS12C128 board
```

```
;*****************************************************
; code section
                ORG  $3100
Entry
                LDS         #Entry              ; initialize the stack pointer

                ldx         #msg1               ; print the first message, 'Well'
                jsr         printmsg
                jsr         nextline
                ldx         #msg2               ; print the second message
                jsr         printmsg
                jsr         nextline

                ldaa        #$3E                ; PPAGE for HCS12C128 board
                staa        PPAGE               ; needed if $8000, not needed for $4800

                jsr         fclkset             ; set flash memory clock
                jsr         dly1s
                jsr         ferase              ; erase 1024 byte sector from $4800
                jsr         dly1s
                jsr         fwr40b              ; write 40 bytes to flash, starting at $4800

                ldx         #msg3               ; print the third message
                jsr         printmsg
                jsr         nextline

                jmp         $4804

looop
                nop
                bra         looop




;*****************************************************
; subroutine section
;
;***********fclkset************************************
; fclkset: flash memory clock setting subroutine
; >>>>> Must use OSCCLK and NOT BUSCLK <<<<<
;    oscillator clock (OSCCLK) = 4MHz for HCS12C128 board
;    bus clock (BUSCLK) = 24MHz
;    prescale by 8: yes, 4MHz/8 = 500KHz for HCS12C128 board
;    divider n: $03, 500KHz/3 = 166.6KHz for HCS12C128 board
; the fclock will run at 166.6KHz for HCS12C128 board
; fclock must be:  150KHz < fclock < 200KHz

fclkset:
                ldaa       FCLKDIV
                anda       #%10000000
                bne        fclksetend
                ldaa       #$43          ; for HCS12C128 board
                staa       FCLKDIV
fclksetend:
                rts

;***********end of fclkset*****************************
;
```

```
;***********ferase*************************************
; ferase: flash memory sector erase subroutine
;    assume FCLKDIV register is properly set
;    Sector size is 1024 byte, HOMEWORK FOR STUDENT.
ferase:









ferase_p2:
                nop
                ldaa        FSTAT
                anda        #%01000000       ; check for CCIF bit, is it finished?
                beq         ferase_p2        ; wait until flash sector clear done
                rts

;***********end of ferase*******************************
;
```

```
;***********fwr40b*************************************
; fwr40b: flash memory word write subroutine
;    assume FCLKDIV register is properly set
;    Sector size of 1024 byte is already erased
fwr40b:
              ldaa         FSTAT
              anda         #%10000000       ; check for CBEIF bit
              beq          fwr40b           ; wait until set, registers empty to load
              ldaa         FSTAT
              anda         #%00110000       ; check for ACCERR and PVIOL
              beq          fwr40b_p1        ; clear if set
              ldaa         #%00110000
              staa         FSTAT

fwr40b_p1:
             ldx           #$4800           ; set flash memory address pointer
             ldy           #flashd          ; set data pointer
             ldaa          flashc           ; load data byte counter
             lsra                           ;   change it to word counter
             staa          counter1         ; set flash word counter

fwr40b_p2:
             ldd           2,Y+             ; load flash data word to write
             std           2,X+             ; SET flash memory write word address
             ldaa          #$20             ; set command as WRITE(Program), $20
             staa          FCMD
             ldaa          #%10000000       ; clear CBEIF, by writing 1
             staa          FSTAT            ; start flash word write (program) operation

fwr40b_p3:
             nop
             ldaa          FSTAT
             anda          #%10000000       ; check for CBEIF bit, is it ready for next command?
             beq           fwr40b_p3        ; wait until flash command ready

             dec           counter1
             bne           fwr40b_p2

fwr40b_p4:
             nop
             ldaa          FSTAT
             anda          #%01000000       ; check for CCIF bit, is it finished?
             beq           fwr40b_p4        ; wait until flash WRITE all done
             rts

;***********end of fwr40b*************************
;
```

```
;***********printmsg**************************
;* Program: Output character string to SCI port, print message
;* Input:   Register X points to ASCII characters in memory
;* Output:  message printed on the terminal connected to SCI port
;* C
;* Registers modified: CCR
;* Algorithm:
;     Pick up 1 byte from memory where X register is pointing
;     Send it out to SCI port
;     Update X register to point to the next byte
;     Repeat until the byte data $00 is encountered
;       (String is terminated with NULL=$00)
;*******************************************
NULL              equ        $00
printmsg          psha                       ;Save registers
                  pshx
printmsgloop
                  ldaa       1,X+            ;pick up an ASCII character from string
                                             ;   pointed by X register
                                             ;then update the X register to point to
                                             ;   the next byte
                  cmpa       #NULL
                  beq        printmsgdone    ;end of strint yet?
                  jsr        putchar         ;if not, print character and do next
                  bra        printmsgloop
printmsgdone
                  pulx
                  pula
                  rts
;***********end of printmsg*******************

;***************putchar************************
;* Program: Send one character to SCI port, terminal
;* Input:   Accumulator A contains an ASCII character, 8bit
;* Output:  Send one character to SCI port, terminal
;* Registers modified: CCR
;* Algorithm:
;    Wait for transmit buffer become empty
;      Transmit buffer empty is indicated by TDRE bit
;      TDRE = 1 : empty - Transmit Data Register Empty, ready to transmit
;      TDRE = 0 : not empty, transmission in progress
;*******************************************
putchar           brclr    SCISR1,#%10000000,putchar     ; wait for transmit buffer empty
                  staa     SCIDRL                        ; send a character
                  rts
;***************end of putchar*****************

;***************getcharw*********************
;* Program: Input one character from SCI port, terminal/keyboard
;* Input:   none
;* Output:  Accumulator A containing the received ASCII character
;* Registers modified: CCR
;* Algorithm:
;    Wait for receive buffer become full
;      Receive buffer full is indicated by RDRF bit
;      RDRF = 1 : full - Receive Data Register Full, 1 byte received
;      RDRF = 0 : not full, 0 byte received
;*******************************************

getcharw          brclr    SCISR1,#%00100000,getcharw
                  ldaa     SCIDRL
                  rts
;***************end of getchar***************
;
;***************nextline*********************
nextline          ldaa      #CR                      ; move the cursor to beginning of the line
                  jsr       putchar                  ;   Cariage Return/Enter key
                  ldaa      #LF                      ; move the cursor to next line, Line Feed
                  jsr       putchar
                  rts
;***************end of nextline***************
;
```

```
;************************************************************
; delay1sec subroutine
;
dly1s
                      LDY      dly1s_r2           ; long delay by
dly1s_p1              JSR      dlyMS              ; Y * dlyMS
                      DEY
                      BNE      dly1s_p1
                      RTS

;************************************************************
; dlyMS subroutine
;
; This subroutine cause few msec. delay
;
; Input:  a 16bit count number in 'dly1s_r1'
; Output: time delay, cpu cycle waisted
; Registers in use: X register, as counter
; Memory locations in use: a 16bit input number in 'dly1s_r1'
;
; Comments: one can add more NOP instructions to lengthen
;           the delay time.
dlyMS
                       LDX      dly1s_r1          ; short delay
dlyMS_p1               NOP                        ; X * NOP
                       DEX
                       BNE      dlyMS_p1
                       RTS
```

```
;***********Flash ROM printmsg*************************
;* Program: Output character string to SCI port, print 4 byte message
;* Input:   4 ascii bytes at $4800.
;* Register X points to ASCII characters in memory
;* Output:  message printed on the terminal connected to SCI port
;* C
;* Registers modified: X, A, CCR
;* Algorithm:
;     Pick up 1 byte from memory where X register is pointing
;     Send it out to SCI port
;     Update X register to point to the next byte
;     Repeat until the 4 byte data is finished.
;* this is NOT a subroutine, it is infinite loop
;* this code will be programmed into Flash ROM starting at $4804
;*****************************************

                            nop
                            nop
                            nop
                            nop
                            nop
                            nop
                            nop
                            nop

hw11code
                            LDS           #$3100         ; initialize the stack pointer
                            ldx           #$4800         ; for HCS12C128 board
                            ldaa          1,X+
                            bsr           fputchar
                            ldaa          1,X+
                            bsr           fputchar
                            ldaa          1,X+
                            bsr           fputchar
                            ldaa          1,X+
                            bsr           fputchar
fprintmsgloop
                            nop
                            nop
                            nop
                            nop
                            nop
                            bra            fprintmsgloop

fputchar                    brclr          SCISR1,#%10000000,fputchar   ; wait for transmit buffer empty
                            staa           SCIDRL                       ; send a character
                            rts



;* the machine code for the above program is as follows:
;* the first 4 byte is the data to print (student's id, last 4 digits in ascii)
;
;            DC.B  $30,$30,$30,$30,$CF,$31,$00,$CE
;            DC.B  $48,$00,$A6,$30,$07,$13,$A6,$30      ;for HCS12C128 board
;            DC.B  $07,$0F,$A6,$30,$07,$0B,$A6,$30
;            DC.B  $07,$07,$A7,$A7,$A7,$A7,$A7,$20
;            DC.B  $F9,$4F,$CC,$80,$FC,$5A,$CF,$3D
;
;* after flash programming, when the program at $4804 is ran,
;* it will print 0000 on the SCI port terminal.
;
              END                                       ; this is end of assembly source file
                                                        ; lines below are ignored - not assembled/compiled
```