```
;*****************************************************
;* CMPEN 472, HW11 sample 1 program,  converting analog
;*            signal to digital data, MC9S12C128 Program
;*
;* Date: 11/02/2018     Kyusun Choi
;* Date: 04/08/2020     Kyusun Choi   Updated for CodeWarrior Debug/Simulator
;* Date: 11/06/2020     Kyusun Choi
;* Date: 04/14/2020     Kyusun Choi
;* Date: 04/04/2022     Kyusun Choi
;* Date: 11/16/2022     Kyusun Choi
;*
;* Term ATD = ADC = Analog-to-Digital Converter
;*
;*   Sample program for ADC Testing is
;*   non-interrupt, busy wait method (different from Homework 11)
;*
;* For SIMULATOR:
;*   Serial communication at fast rate (1.5M baud).
;*   Typewriter program, but when 'enter' key hit, ADC acquired
;*   number printed on the terminal window, as a hexadecimal number.
;*   Single AD conversion, 8bit, right justified.
;*
;*****************************************************
; export symbols
            XDEF      Entry       ; export 'Entry' symbol
            ABSENTRY  Entry       ; for assembly entry point

; symbols/addresses
ATDCTL2     EQU  $0082            ; Analog-to-Digital Converter (ADC) registers
ATDCTL3     EQU  $0083
ATDCTL4     EQU  $0084
ATDCTL5     EQU  $0085
ATDSTAT0    EQU  $0086
ATDDR0H     EQU  $0090
ATDDR0L     EQU  $0091
ATDDR7H     EQU  $009e
ATDDR7L     EQU  $009f

SCIBDH      EQU  $00c8            ; Serial port (SCI) Baud Rate Register H
SCIBDL      EQU  $00C9            ; Serial port (SCI) Baud Register L
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
;*   CodeWarrior project MUST specify MC9S12C32 chip for the terminal simulation to work.

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
ATDdone     DS.B  1               ; ADC finish indicator, 1 = ATD finished

msg1        DC.B  'Hello, this is an ADC testing program.', $00
msg2        DC.B  'One ATD convert per each Enter Key hit.', $00
msg3        DC.B  'You may type below.', $00

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

            ldd   #$0001          ; For SIMULATION, Set SCI Baud Register = $0001 => 1.5M
```

```
baud at 24MHz
            std   SCIBDH          ; SCI port baud rate change
            ldaa  #$0C            ; Enable SCI port Tx and Rx units
            staa  SCICR2          ; disable SCI interrupts

; ATD initialization
            LDAA  #%11000000      ; Turn ON ADC, clear flags, Disable ATD interrupt
            STAA  ATDCTL2
            LDAA  #%00001000      ; Single conversion per sequence, no FIFO
            STAA  ATDCTL3
            LDAA  #%10000111      ; 8bit, ADCLK=24MHz/16=1.5MHz, sampling time=2*(1/ADCLK)
            STAA  ATDCTL4         ; for SIMULATION

; Guide user, instruction
            ldx   #msg1           ; print the first message, 'Hello'
            jsr   printmsg
            jsr   nextline
            ldx   #msg2           ; print the second message, instruction
            jsr   printmsg
            jsr   nextline
            ldx   #msg3           ; print the third message, more instruction
            jsr   printmsg
            jsr   nextline

loop1       jsr   getchar         ; type writer - what is typed on key board
            jsr   putchar         ; is displayed on the terminal window
            cmpa  #CR
            bne   loop1           ; if Enter/Return key is pressed, move the
            ldaa  #LF             ; cursor to next line
            jsr   putchar

            jsr   go2ADC

            bra   loop1


;***************************************************
; subroutine section

;***********single AD conversiton*********************
; This is a sample, non-interrupt, busy wait method
;
go2ADC
            PSHA                  ; Start ATD conversion
            LDAA  #%10000111      ; right justified, unsigned, single conversion,
            STAA  ATDCTL5         ; single channel, CHANNEL 7, start the conversion

adcwait     ldaa  ATDSTAT0        ; Wait until ATD conversion finish
            anda  #%10000000      ; check SCF bit, wait for ATD conversion to finish
            beq   adcwait

            ldaa  #'$'            ; print the ATD result, in hex
            jsr   putchar

            ldaa  ATDDR0L         ; for SIMULATOR, pick up the lower 8bit result
            jsr   printHx         ; print the ATD result
            jsr   nextline

            PULA
            RTS
;***********end of AD conversiton**************

;***********printHx*****************************
; prinHx: print the content of accumulator A in Hex on SCI port
printHx     psha
```

```
             lsra
             lsra
             lsra
             lsra
             cmpa   #$09
             bhi    alpha1
             adda   #$30
             jsr    putchar
             bra    low4bits
 alpha1      adda   #$37
             jsr    putchar
 low4bits    pula
             anda   #$0f
             cmpa   #$09
             bhi    alpha2
             adda   #$30
             jsr    putchar
             rts
 alpha2      adda   #$37
             jsr    putchar
             rts
 ;************end of printhx****************************

 ;************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****************

;****************getchar***********************
;* Program: Input one character from SCI port (terminal/keyboard)
;*             if a character is received, other wise return NULL
;* Input:   none
;* Output:  Accumulator A containing the received ASCII character
;*          if a character is received.
;*          Otherwise Accumulator A will contain a NULL character, $00.
;* Registers modified: CCR
;* Algorithm:
;    Check 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
;***********************************************

getchar     brclr SCISR1,#%00100000,getchar7
            ldaa  SCIDRL
            rts
getchar7    clra
            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***************

*
* Add any subroutines here
*



            END               ; this is end of assembly source file
                              ; lines below are ignored - not assembled/compiled
```