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

SCIBDH      EQU         $00C8       ; Serial port (SCI) Baud 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

CRGFLG      EQU         $0037       ; Clock and Reset Generator Flags
CRGINT      EQU         $0038       ; Clock and Reset Generator Interrupts
RTICTL      EQU         $003B       ; Real Time Interrupt Control

TIOS        EQU         $0040   ; Timer Input Capture (IC) or Output Compare (OC) select
TIE         EQU         $004C   ; Timer interrupt enable register
TCNTH       EQU         $0044   ; Timer free runing main counter
TSCR1       EQU         $0046   ; Timer system control 1
TSCR2       EQU         $004D   ; Timer system control 2
TFLG1       EQU         $004E   ; Timer interrupt flag 1
TC6H        EQU         $005C   ; Timer channel 2 register

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

; Init

ldaa   #$0C         ; Enable SCI port Tx and Rx units
staa   SCICR2       ; disable SCI interrupts
ldd    #$0001       ; Set SCI Baud Register = $0001 => 1.5M baud at 24MHz (for simulation)
;ldd    #$0002      ; Set SCI Baud Register = $0002 => 750K baud at 24MHz
;ldd    #$000D      ; Set SCI Baud Register = $000D => 115200 baud at 24MHz
;ldd    #$009C      ; Set SCI Baud Register = $009C => 9600 baud at 24MHz
std    SCIBDH       ; SCI port baud rate change

bset   RTICTL,%00011001 ; set RTI: dev=10*(2**10)=2.555msec for C128 board
                        ;      4MHz quartz oscillator clock
bset   CRGINT,%10000000 ; enable RTI interrupt
bset   CRGFLG,%10000000 ; clear RTI IF (Interrupt Flag)

StartTimer6oc
          PSHD
          LDAA   #%01000000
          STAA   TIOS             ; set CH6 Output Compare
          STAA   TIE              ; set CH6 interrupt Enable
          LDAA   #%10000000       ; enable timer, Fast Flag Clear not set
```

```
        STAA   TSCR1
        LDAA   #%00000000       ; TOI Off, TCRE Off, TCLK = BCLK/1
        STAA   TSCR2            ;   not needed if started from reset

        LDD    #3000            ; 125usec with (24MHz/1 clock)
        ADDD   TCNTH            ;    for first interrupt
        STD    TC6H             ;

        BSET   TFLG1,%01000000   ; initial Timer CH6 interrupt flag Clear, not needed if fast clear set
        LDAA   #%01000000
        STAA   TIE              ; set CH6 interrupt Enable
        PULD
        RTS

; 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


; Vector
; interrupt vector section
        ORG    $FFF0            ; RTI interrupt vector setup for the simulator
;        ORG    $3FF0            ; RTI interrupt vector setup for the CSM-12C128 board
        DC.W   rtiisr


; interrupt vector section

        ORG     $FFE2       ; Timer channel 6 interrupt vector setup, on simulator
        DC.W    oc6isr


; Subroutine

;***********RTI interrupt service routine***************
rtiisr     bset   CRGFLG,%10000000 ; clear RTI Interrupt Flag - for the next one
        ldx    ctr2p5m          ; every time the RTI occur, increase
        inx                 ;    the 16bit interrupt count
        stx    ctr2p5m
rtidone     RTI
;***********end of RTI interrupt service routine********

;***********Timer OC6 interrupt service routine***************
oc6isr
        ldd   #3000             ; 125usec with (24MHz/1 clock)
        addd  TC6H              ;    for next interrupt
        std   TC6H              ;
        bset  TFLG1,%01000000    ; clear timer CH6 interrupt flag, not needed if fast clear enabled
        ldd   ctr125u
        ldx   ctr125u
        inx                 ; update OC6 (125usec) interrupt counter
```

```
        stx   ctr125u
        clra                ;   print ctr125u, only the last byte
        jsr   pnum10        ;   to make the file RxData3.txt with exactly 1024 data
oc2done     RTI
;***********end of Timer OC6 interrupt service routine********

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

;***********pnum10****************************
pnum10      pshd            ;Save registers
            pshx
            pshy
            clr     CTR         ; clear character count of an 8 bit number

            ldy     #BUF
pnum10p1    ldx     #10
            idiv
            beq     pnum10p2
            stab    1,y+
            inc     CTR
            tfr     x,d
            bra     pnum10p1

pnum10p2    stab    1,y+
            inc     CTR
;-------------------------------------

pnum10p3    ldaa    #$30
            adda    1,-y
            jsr     putchar
            dec     CTR
            bne     pnum10p3
            jsr     nextline
            puly
```

```
            pulx
            puld
            rts
;***********end of pnum10*******************

;***********printmsg**************************
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?
            bsr     putchar        ;if not, print character and do next
            bra     printmsgloop
printmsgdone    pulx
            pula
            rts
;***********end of printmsg********************

;***************putchar************************
putchar     brclr SCISR1,#%10000000,putchar   ; wait for transmit buffer empty
          staa  SCIDRL                  ; send a character
          rts
;***************end of putchar*****************

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

;****************delay1ms**********************
delay1ms:   pshx
          ldx   #$1000          ; count down X, $8FFF may be more than 10ms
d1msloop    nop                 ;   X <= X - 1
          dex                 ; simple loop
          bne   d1msloop
          pulx
          rts
;****************end of delay1ms***************
```