Introduction: The Dragon12 EVB provides a number of input devices connected to ATD0 input channels. This lab demonstrates a general-purpose A/D conversion routine along with a number of support routines to acquire and display A/D results on user-selected input channels.

Procedure:
1. Prepare, assemble, download and run ADC2.asm as shown in listing 1.

   Program Description:
   This program allows the user to select ATD channels (0..7) using the bottom 3 positions of DIP switch SW1 and performs 10-bit A/D conversions on the selected channel. The bottom 8 bits of the A/D result are displayed on the PortB LED bank and the decimal equivalent of the 10-bit result is output to the terminal when SW2 is pressed.

2. For each of the ATD0 input channels shown in the table below, set the DIP switch to the binary equivalent of the channel and observe the conversion results on the LED bank. Also press SW2 to display the decimal A/D result on the terminal. For each channel as shown, try the “stimulus” action to test different results.

3. Review the program listing to determine how this program works.

   Q1. What logic operation is used to ignore all but the bottom 3 positions of the DIP switch input? ______

   Q2. What mathematical operation is used to efficiently convert from binary to decimal? ______

   Q3. Where is the actual temperature stored when converting channel 5? __________

4. With the program still running, select the temperature sensor. Stop the program, examine the variable where the converted temperature is stored and verify for sensible value.

5. Demo this lab to receive signoff below and 5 bonus lab points!

6. When finished, press any terminal key to exit program. Remember to return SW7 in “EVB” mode for the next reset / power-up!

Instructor Signoff: ______________
Listing 1: Dragon12 ATD demo program

* ADC.asm: demonstrate analog input using A/D converter

***************************************************************************
*         <<< EQUATE SECTION >>>
***************************************************************************
* constants
CR        EQU       $0D       ;ASCII Carriage Return
LF        EQU       $0A       ;ASCII Line Feed
SPC       EQU       ' '       ;ASCII Space
***************************************************************************
*         <<< DATA SECTION >>>
***************************************************************************
* variable data allocations
ORG       DATA
Channel   RMB       1         ;ATD input channel to convert:
Result    RMB       2
Temp      RMB       2
StrBuf    RMB       6
***************************************************************************
*         <<< MAIN PROGRAM LOOP >>>
***************************************************************************
* START movb #$80,ATD0CTL2  ;turn on ATD0
   movb #$2,DDRJ        ;enable PORTB LEDs
   ldaa #$0F          ;turn off numeric displays
   staa PTP
   staa DDRP
   movb #$FF,DDRB     ;make PORTB all outputs
   ldx #Msg1           ;point X to initial message
   jsr PUTS           ;output it
MainLp    ldaa PTH           ;read DIP switch
   anda #7            ;keep only bottom 3 bits
   staa Channel       ;update current channel selection
   jsr GetADCr        ;go perform A/D process
   std Result         ;save result of A/D
   stab PTH,8,CkKey   ;skip output if SW2 is released
   ldx #Msg2           ;else output new status line
   jsr PUTS
   ldab Channel       ;with current channel number
   addb #'0'          ;make ASCII
   jsr PUTCHAR
   ldab ':'
   jsr PUTCHAR
   ldab #SPC

Bonus Lab: Dragon12 ATD - Page 2
jsr  PUTCHAR
ldd  Result ;get current ADC result
ldx  #$StrBuf ;point X to ASCII string buffer
ejr  ItoA ;convert int to ASCII
jsr  FUTS ;output ASCII string
ldab  #$CR
jsr  PUTCHAR
CkKey  brcrl  SCI0SR1,$20,MainLp ;continue until RDRF
swi                     ;else exit program

*** Subroutines ***

* GetADC: perform A/D conversion and return result
*   in: A = channel number to convert
*   out: D = result
GetADCr  oraa  #$80  ;enable right-justification results
GetADCl  staa  ATD0CTL5 ;start A/D conversion
waitSCF   tst       ATD0STAT0 ;poll for SCF on bit 7
bpl       waitSCF       ;A/D not done yet
ldd       ATD0DR0 ;fetch result of A/D conversion
rts                     ;and return it

* ItoA: convert 16-bit binary number to null-terminated ASCII string
*   in: D = number to convert
*       X = address of ASCII string buffer (need 6 bytes!)
*   out: X = address of 1st non-blank digit
ItoA      pshd                    ;save used registers
pshy
pshx                    ;TOS=buffer address
leay      5,X           ;point Y to end of buffer
movb      #0,0,Y        ;place null byte at end of string
ialoop    ldx       #10           ;to base 10
idiv                    ;get least sig digit
addb      #'0'          ;make it ASCII
stab      1,-Y          ;store in buffer R-to-L
trf       X,D           ;move quotient to D
cmpeq      0,SP          ;is Y back to begin of buffer?
bne       ialoop        ;if not
pulx                    ;pull original buffer pointer
supress   ldaa      0,X           ;L-to-R, get char from buffer
cmpa      #'0'          ;is it a zero?
bne       iaexit        ;exit if not
movb      #SPC,1,X+     ;else replace with space
cmpeq      0,X           ;hit null byte?
bne       supress       ;continue if not
movb      #48,1,-X      ;else force least sig digit to '0'
iaexit    puly                    ;restore other registers
puld
rts

* string output routine, X points to null-terminated string
FUTS      ldab  0,x       ;get 1 char from string
beq       PSDONE    ;done if terminating null
jsr  PUTCHAR ;output this char
inx                 ;advance X to next char
bra       FUTS      ;continue
PSDONE    rts                 ;done, return to caller

* single character (in B) output routine
PUTCHAR    pshx                ;save caller's X
ldx       VFPUTCHAR ;fetch address of monitor rtn
jsr  0,X       ;call monitor rtn.
pulx                    ;restore caller's X
rts

* Constant data
Msg1      FCC       "ADC test/demo program, press SW2 to output result, press any key to exit"
FCB       CR,0
Msg2      FCC       "channel 
FCB       0

***************************************************************************
ENDER