```
; Marc Dominic Cabote
; CECS 460  Spring 2018
; Transmit Engine Assembly
;=================================================================
;FOR DISPLAYING CHARACTERS
;=================================================================
ascii_C     EQU     0043    ; "C"
ascii_S     EQU     0053    ; "S"
ascii_U     EQU     0055    ; "U"
ascii_L     EQU     004C    ; "L"
ascii_B     EQU     0042    ; "B"
ascii_E     EQU     0045    ; "E"
ascii_4     EQU     0034    ; "4"
ascii_6     EQU     0036    ; "6"
ascii_0     EQU     0030    ; "0"
ascii_SPC   EQU     0020    ; SPACE
ascii_DASH  EQU     002D    ; "-"
ascii_CR    EQU     000D    ; Carriage Return
ascii_LF    EQU     000A    ; Line Feed
;=================================================================
;FOR DISPLAYING LINE COUNT
;=================================================================
TEN_THSND   EQU     2710    ; 10,000
ONE_THSND   EQU     03E8    ;  1,000
ONE_HNDRD   EQU     0064    ;    100
TEN         EQU     000A    ;     10
ONE         EQU     0001    ;      1

;=================================================================
; REGISTERS
;=================================================================
CHAR_REG    EQU     R1
CHAR_COUNT  EQU     R2
LEDS        EQU     R3
LED_COUNT   EQU     R4
DELAY_COUNT EQU     R5
LINE_COUNT  EQU     R6

;TEMPORARY REGISTERS FOR BIN_TO_ASCII
RE          EQU     R7
RD          EQU     R8
RB          EQU     R9

;=================================================================
; INIT
;=================================================================
;=================================================================
; Load Registers --> "CSULB CECS 460 - [COUNT]<cr><lf>
;=================================================================
START   ENINT

        LOAD  CHAR_COUNT,  0000
        LOAD  LED_COUNT,   0000
        LOAD  DELAY_COUNT, 0000
        LOAD  LINE_COUNT,  0000

        LOAD  LEDS,        0001 ;LED INIT

        LOAD  CHAR_REG, ascii_C
        STORE CHAR_REG, 0000

        LOAD  CHAR_REG, ascii_S
        STORE CHAR_REG, 0001

        LOAD  CHAR_REG, ascii_U
        STORE CHAR_REG, 0002

        LOAD  CHAR_REG, ascii_L
        STORE CHAR_REG, 0003
```

```
70
71          LOAD  CHAR_REG, ascii_B
72          STORE CHAR_REG, 0004
73
74          LOAD  CHAR_REG, ascii_SPC
75          STORE CHAR_REG, 0005
76
77          LOAD  CHAR_REG, ascii_C
78          STORE CHAR_REG, 0006
79
80          LOAD  CHAR_REG, ascii_E
81          STORE CHAR_REG, 0007
82
83          LOAD  CHAR_REG, ascii_C
84          STORE CHAR_REG, 0008
85
86          LOAD  CHAR_REG, ascii_S
87          STORE CHAR_REG, 0009
88
89          LOAD  CHAR_REG, ascii_SPC
90          STORE CHAR_REG, 000A
91
92          LOAD  CHAR_REG, ascii_4
93          STORE CHAR_REG, 000B
94
95          LOAD  CHAR_REG, ascii_6
96          STORE CHAR_REG, 000C
97
98          LOAD  CHAR_REG, ascii_0
99          STORE CHAR_REG, 000D
100
101         LOAD  CHAR_REG, ascii_SPC
102         STORE CHAR_REG, 000E
103
104         LOAD  CHAR_REG, ascii_DASH
105         STORE CHAR_REG, 000F
106
107         LOAD  CHAR_REG, ascii_SPC
108         STORE CHAR_REG, 0010
109
110         LOAD  CHAR_REG, ascii_CR
111         STORE CHAR_REG, 0016
112
113         LOAD  CHAR_REG, ascii_LF
114         STORE CHAR_REG, 0017
115
116 ;=====================================================================
117 ;               OUTPUT a walking LED
118 ;=====================================================================
119 MAIN
120         ADD LED_COUNT, 0001
121         ADDC DELAY_COUNT, 0000
122         COMP DELAY_COUNT, 0007 ; delay
123         JUMPC DONE_LED
124
125         LOAD LED_COUNT, 0000
126         LOAD DELAY_COUNT, 0000
127         RL LEDS ; rotate LEDS
128
129 DONE_LED
130         OUTPUT LEDS, 0001
131         JUMP MAIN
132 ;=====================================================================
133 ;   ADDRESS FOR BIN_TO_ASCII 0100
134 ;=====================================================================
135         ADDRESS 0100
136
137 BIN_TO_ASCII
138         LOAD RE, LINE_COUNT ;RE<-R2
```

```
        LOAD RD, TEN_THSND ;RD<-2710
        CALL FIND_IT       ;FIND_IT
        ADD RB, 0030       ;convert to ascii_hex
        STORE RB, 0011     ;store to memory

        LOAD RD, ONE_THSND ;RD<-03E8
        CALL FIND_IT
        ADD RB, 0030
        STORE RB, 0012

        LOAD RD, ONE_HNDRD ;RD<-0064
        CALL FIND_IT
        ADD RB, 0030
        STORE RB, 0013

        LOAD RD, TEN       ;RD<-000A
        CALL FIND_IT
        ADD RB, 0030
        STORE RB, 0014

        ADD RE, 0030       ;least significant
        STORE RE, 0015

        RETURN             ;return ISR routine

;=============================================================================
;   ADDRESS FOR BIN_TO_ASCII 0200
;=============================================================================
        ADDRESS 0200

FIND_IT
        LOAD RB, 0000 ; RB<-0000
NOT_DONE
        SUB RE, RD    ; RE<-RE-RD
        JUMPC RESTORE ; if there is a carry restore RE
        ADD RB, 0001  ; increment RB
        JUMP NOT_DONE ; keep subtracting
RESTORE
        ADD RE, RD    ; restore last value
        RETURN        ; return bin to ascii routine

;=============================================================================
;               ADDRESS FOR ISR 0300
;=============================================================================
        ADDRESS 0300

ISR
        COMP   CHAR_COUNT, 0011     ;check MSB address
        CALLZ  BIN_TO_ASCII

        FETCH  CHAR_REG, CHAR_COUNT ;fetch mem data
        OUTPUT CHAR_REG, 0000       ;output mem data
        ADD    CHAR_COUNT, 0001     ;traverse memory

        COMP   CHAR_COUNT, 0018
        JUMPZ  LINE_OUT
        RETEN

LINE_OUT
        LOAD CHAR_COUNT, 0000 ;reinitialize
        ADD  LINE_COUNT, 0001 ;increment line counter
        RETEN

;=============================================================================
;               ISR VECTORED THROUGH 0FFE
;=============================================================================
        ADDRESS 0FFE
ENDIT
        JUMP ISR
```

```
208           END
209
210   ;=============================================================================
211   ;                           END
212   ;=============================================================================
```