

ELEC 2220 Computer Systems
Homework #12
Due: Wednesday, June 30

Program 1 (parameters passed in registers)

Write a subroutine that finds and returns the smallest value in an array of signed 16-bit numbers. *The subroutine should be capable of searching an array of any size (up to 256 words), and starting at any address in memory.* Pass the starting address of the array to the subroutine in the X register, and the size of the array in accumulator B. Return the smallest value in the array to the calling program in register Y.

To test the subroutine, define the following variables in the data area, and write a single "main" program that calls the subroutine two times, once to find the smallest value in Array1, storing it in Value1, and once to find and store the smallest value from Array2 in Value2.

```
Array1:    dc.w  30,-5500,129,-1080,25000,-20000,200,-4,-220,0
Array2:    dc.w  -200,-1000,50,12800,70
Value1:    ds.w  1
Value2:    ds.w  1
```

Main Program

```
Example:   ...set up first call
           ...call subroutine
           ...store first result
           ...set up second call
           ...call subroutine
           ...store second result
```

Turn in a printout of your source program and the *CodeWarrior* debug window, showing the two arrays and the two "values" in memory after the program has executed.

Program 2(parameters passed on the stack)

(See next page)

Program 2(parameters passed on the stack)

Variable-length strings of one-to-five ASCII characters, representing decimal digits as might be entered from a keyboard, are stored in consecutive bytes of memory. The ASCII “null character” is stored immediately after the last byte of each string to mark the end of the string. (See examples below.) Each string represents an unsigned decimal number in the range [0...65535]. Assume, for this exercise, that there are no “illegal” entries (i.e. non-digits, or numbers outside of this range).

Write a **subroutine** “ASC2BIN” to convert a specified ASCII character string to a **16-bit binary number** (NOT BCD) and store it in a designated word variable in memory.

- The starting address of the string should be passed to the subroutine on the stack.
- The address of the word at which the result is to be stored should likewise be passed to the subroutine on the stack.
- The subroutine may contain no direct memory addresses, nor is the size of any string known in advance.
- Upon return from the subroutine, all registers should be exactly as they were when the subroutine was entered, and all stack space used for parameters should be reclaimed.
- The subroutine cannot make any assumptions about values contained in the registers, other than PC and SP.

To test this subroutine, write a main program that calls the subroutine twice, once for each of the following strings, with the subroutine storing the 16-bit binary numbers in the data words labeled Num1 and Num2, respectively.

```
Strng1 dc.b  "43705",0
Strng2 dc.b  "283",0
Num1 ds.w  1
Num2 ds.w  1
```

Turn in your source program, and the *CodeWarrior* variables window showing the final values of the two 16-bit numbers. Highlight these numbers in the memory display area.

The program is to be executed only one time, with the “main” program calling the subroutine twice. Therefore, only one result window printout is necessary.

HINT: Recall that 43705 represents $4 \times 10^4 + 3 \times 10^3 + 7 \times 10^2 + 0 \times 10 + 5$

$$\begin{aligned} &= (4 \times 10^3 + 3 \times 10^2 + 7 \times 10 + 0) \times 10 + 5 \\ &= ((4 \times 10^2 + 3 \times 10 + 7) \times 10 + 0) \times 10 + 5 \\ &= (((4 \times 10 + 3) \times 10 + 7) \times 10 + 0) \times 10 + 5 \\ &= (((((4) \times 10 + 3) \times 10 + 7) \times 10 + 0) \times 10 + 5 \end{aligned}$$