Benjamin Linam

Bml42@students.uwf.edu

EEL 4744L: Microprocessor Applications Laboratory

Lab 4: Writing and Testing a Simple Program

2/19/2018

Objective

Introduce students to writing and testing a program in HC11 assembly language and using the BUFFALO I/O routines to display/verify the results.

Introduction/Background/Theory

This lab required construction of an assembly program which would count the number of entries in an N-byte array to determine if each number was positive, negative, even, and/or odd. Provided for the lab was a list of numbers to be stored in memory starting at location $100. The value "N" was the number of entries in the array and was utilized by the program to determine how many values it should test.

Procedure

1. Starting at memory location $00, the assembly language code allocates memory to store how many even/odd and positive/negative numbers are in the N-byte array.
2. The array of numbers provided during lab for testing is then saved into memory starting at $100.
3. Program execution begins at $B600 with setting each count variable to zero as to prevent any residual values stored in memory from skewing the results.
4. Accumulator X is then loaded with the location of the first value in the array and added to the value of accumulator B to increment through the array.
5. Using BRCLR and BRSET, each number is tested to determine if it is positive or negative or even or odd. If the least significant bit (LSB) is 1, the number is odd, while if it is 0, the number is considered even. Similarly, if the most significant bit (MSB) is 1, the number is negative, while if it is 0, it is positive.

6. After testing each number, the individual counters are incremented and the loop is repeated with the next value in the array. If the last element has been evaluated, the program escapes the loop and terminates.

```
N       equ     10

        org     $00
negCnt  rmb     1                       ;1 byte saved for negative count
posCnt  rmb     1                       ;1 byte saved for positive count
evenCnt rmb     1                       ;1 byte saved for even count
oddCnt  rmb     1                       ;1 byte saved for odd count

        org     $100
array   fcb     $80,$A4,$F6,$90,$E8,$C2,$74,$53,$11,$67 ;array of 1 byte numbers to be evaluated

        org     $B600
        ldab    #$00                    ;[b] must start as 0
        stab    negCnt                  ;each of these values must also be init to 0
        stab    posCnt
        stab    evenCnt
        stab    oddCnt

loop    ldx     #array                  ;start at beginning of array

        abx                             ;move to array + [b] : [b] will be counter variable
                                        ;X stores mem(array)
        brset   0,X %10000000 isNeg     ;tests for neg
        brclr   0,X %10000000 isPos     ;tests for pos
eve     brclr   0,X %00000001 isEve     ;tests for even
        brset   0,X %00000001 isOdd     ;tests for odd
        bra     done                    ;move to end of current loop, should never reach this command, but is here as safegaurd

isNeg   inc     negCnt                  ;negative value located, negCnt+=1, cannot also be positive, move to eve/odd
        bra eve
isPos   inc     posCnt                  ;positive value located, posCnt+=1, move to eve/odd
        bra eve
isEve   inc     evenCnt                 ;even value located, eveCnt+=1, cannot also be odd, move to done
        bra done
isOdd   inc     oddCnt                  ;odd value located, oddCnt+=1, move to done
        bra done                        ;no more comparisons, move to end of current loop

done    incb                            ;[B] will increment until it reaches N (end of array)
        cmpb    #N
        bhs exit                        ;if [B] is higher or same as N, exit from loop
        bra loop                        ;move to next element in array

exit    swi                             ;end of code
```

Figure 1: Assembly language code designed to evaluate the total number of positive/negative and even/odd elements in an N-byte array.

7. After running the program using the BUFFALO I/O command “G B600” (B600 being the location of the starting byte of the program’s code), the individual counts will be saved from memory locations $00-$03. The values will be saved in the following order: negative count, positive count, even count, and odd count. Entering the command “MD 00 1” displays memory locations $00-$0F to enable all four values to be viewed at once.

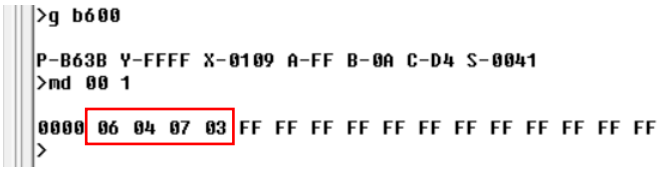

Figure 2: Memory locations $00-$0F after evaluating elements stored in array.

Conclusions

The program and lab were completed quickly and efficiently. The only problem encountered while coding the program, was incorrect usage of a 4-bit mask for BRCLR and BRSET instead of 8-bits which must be included to test each number.