EXAMPLE 6

OBJECTIVE
This simple example has the following objectives:

- Introduce *hex arithmetic with carry*
- Facilitate comparison with the decimal case by using the same numbers as in Example 5

PROGRAM
Ex6.asm program is similar to ex5.asm, but uses hex numbers representation. It performs the arithmetic operation $09 + $08 = $11. To achieve this, the program follows two paths:

Path 1: Load $09 into accA increment eight times.
- During this process, the number $09 is increased by one eight times using the INCA opcode. Sequentially, it goes $09 \rightarrow $0a \rightarrow $0b \rightarrow $0c \rightarrow $0d \rightarrow $0e \rightarrow $0f \rightarrow $10 \rightarrow $11.
- After the first six increments, the end of the hex symbols list (i.e., symbol f) has been reached. The process is at $0f.
- The next (seventh) increment is performed with a carry, i.e. a zero is placed in the units placeholder and a one is carried over into the next placeholder. At this point, the number $10 is obtained.
- The remaining and last increment moves the number from $10 to $11 by adding a one to the units placeholder.

Path 2: Load $09 into accA, $08 into accB and perform the addition $A + B \rightarrow A$. The result is $09 + 08 = 11$. (You can check this by using a Hex calculator.)

Path 1 is explicit and lets the user understand the carry process. Path 2 is shorter and is usually taken in programming. Note that the result, $11, is the hex equivalent of the decimal number 17 from Example 5.
EXECUTION
Open THRSim11. Maximize THRSim11 window. Close the Commands window. Open file Ex3.asm. Assemble file. Tile windows. Set break point at the line containing SWI. Set the display of A, B, and D registers to ‘Hexadecimal’. Reset registers D, X, Y. Your screen should look like this:

Use the ‘Step’ button to step through the program. Press the Step button once. The registers do not change.
Path 1

Step again. The numbers $09 is been loaded into accA. Your screen should look like this:

Step again. AccA has increased by one from $09 to $0a. The screen should look like this:
Continue stepping and watch accA going sequentially through $0b$, $0c$, $0d$, $0e$, $0f$. At this point, the screen should look like this:

Step again. An increment from $0f$ to $10$ takes place. The screen should look like:

Perform the final step and get the desired result at the end of Path 1, A $11$. 

Dr. Victor Giurgiutiu                  Page 4                  1/8/2003
Path 2: Place PC at $c000b (beginning of Path 2). Step once. The accA is reloaded with $09. Your screen looks like this:

Step again to load accB with $08. Now, step again to perform the addition $09 + $08 = $11. The result appears in accA. The screen looks like this:
WHAT YOU HAVE LEARNED
In this simple example, you have understood hex arithmetic with carry:

1. In Path 1, you saw addition as a repeated incrementation. The number $09 was loaded in accA and subsequently incremented.
   - After 6 increments, from $09 to $0a, $0b, $0c, $0d, $0d, $0e, and $0f, the hex symbols were exhausted.
   - The next increment (the seventh) enacted a carry such that the second placeholder (MSD) was increased from 0 to 1, while the first placeholder (LSD) was reset to 0. At this stage, the value in accA was $10. (This is by far different from the decimal 10).
   - The next and final increment (the eighth) incremented the LSD and the number in accA moved from $10 to $11.

2. In Path 2, you saw the direct application of the hex addition using accA and accB, such that $09 + $08 = $11.

3. You should note that the result $11 is the hex equivalent of the decimal result 17 from Example 5 (recall, $9 + 8 = 17$).

4. New words: hex arithmetic with carry, placeholder.