ECE 375 LAB 3 (CHALLENGE)

Simulation Challenge

Lab Time: Thursday 1000-1200

Eric Prather

CHALLENGE QUESTIONS

1. What type of operation does the FUNCTION subroutine perform on its two 16-bit inputs? How can you tell? Give a detailed description of the operation being performed by the FUNCTION subroutine.

FUNCTION performs the following C pseudocode:

```
// X = $0100, Y = $0102, Z = $0104
FUNCTION(*X, *Y, out Z):
        A = *X
        X++ // $0102
        B = *Y
        Y++ // $0104
        B = B + A
        *Z = B
        Z++ // $0106
        A = X
        B = Y
        B = B + A (carry from last sum included)
        *Z = B
        Z++ // $0108
        If(no carryover bit)
                Return;
        Else
                *Z = XH
```

In plain English, this can be described as follows:

FUNCTION sums the values at X and Y in memory and stores them at Z in memory, where X, Y, and Z are contiguous 16-bit values from $0100 to $0105. **Next**, it adds ***Z**, ***Y**, and the carryover- storing the *new* carryover in a special register pin. This second sum is stored in the byte following Z's original address (because z was earlier post-incremented). **Lastly**, if there is a carryover bit, it writes the *next* 8 bits following where the previous sum was stored with the most significant 8 bits of X.

In even simpler terms:

Sums two inputs, stores the result, then sums the result and the second input. Writes to designated output area. All bytes are in reverse endian (rightmost bytes are more significant than leftmost bytes).

Mathematically:

OUTPUT = INPUT 1 + 2* INPUT2

2. Currently, the two 16-bit inputs used in the sample code cause the "brcc EXIT" branch to be taken. Come up with two 16-bit values that would cause the branch NOT to be taken, therefore causing the "st Z, XH" instruction to be executed before the subroutine returns.

Adding 0xFF and 0xFF would do the trick, because it would overflow in the addition operation. If we want the extra statement to execute, that would work nicely.

3. What is the purpose of the conditionally-executed instruction "st Z, XH"?

It is possible that the sum results in a number that is **too big to be stored in 2 bytes**. In order to output the whole number, which may take three bytes, the new most significant byte is tacked onto the "beginning" of the sum if necessary.