EEEE-521-621-Lab10

1) From mycourses download on your desktop this pdf file.
2) The goal of this lab is to augment your instruction set with:
    a. 521: one SIMD instruction: ADDV.
    b. 621: two SIMD instructions: ADDV and SUBV.
3) As discussed in the lecture, if your current data width is:
    a. 12-bit then you split into three 4-bit groups, else if
    b. 14-bit then you split into two 7-bit groups.
4) Anticipated modifications:
    a. "Split" your current add/subtract operations.
    b. Add the necessary statements to decode the new instruction word(s). Their fields are identical to those of your other manipulation instructions.
5) To test your upgraded design, run the code you develop based on the specifications at the end of this document. The arrays can be the random values with which your DM or data memory segment of the MM is initialized after reset.
6) If your pipelined processor and/or cache memory are not working at this time, you can use your non-pipelined processor and/or "monolithic" memory block with the penalties shown below.
7) If time permits, show your simulation to your TA. Note: only the TA will assign a grade.
8) Archive your project.
9) Write your report and upload it along with your archived project(s) in the dropbox on mycourses, as described in the lab policy.
10) This concludes this week's lab.
11) **Grading**:
    a. 521:
        i. 10 points for the correct and complete implementation of ADDV.
        ii. 10 points for the correct and complete execution of the first code.
        iii. 10 points for the correct and complete execution of the second code.
        iv. -5p if not using the pipelined processor.
        v. -5p if not using the cache memory.
    b. 621:
        i. 5 points for the correct and complete implementation of ADDV.
        ii. 5 points for the correct and complete implementation of SUBV.
        iii. 10 points for the correct and complete execution of the first code.
        iv. 10 points for the correct and complete execution of the second code.
        v. -5p if not using the pipelined processor.
        vi. -5p if not using the cache memory.

For the assembly code required below use your own instruction set.

For 12-bit architectures:

Assume an array of 256 elements is pre-initialized in the DM (Harvard) or Data Segment of the MM (von Neumann). Each element is 12-bits wide. The 12-bits are divided into three groups of 4-bits, each representing the color channel value for R-G-B.

1) Write and test the assembly code that would create three arrays that contain each the R, G, and B color channel values for all pixels. Note that in these arrays a 12-bit word will contain one color channel value for three pixels. You can use shifts or rotates.

2) Write and test the assembly code that adds the constant 0x3 to the R channel array elements, subtracts the constant 0x1 from the G and B channel array elements. You should use the ADDV and SUBV SIMD instructions. **521** should add 0x1 to the G and B channels.

For 14-bit architectures:

Assume an array of 256 elements is pre-initialized in the DM (Harvard) or Data Segment of the MM (von Neumann). Each element is 14-bits wide. The 14-bits are divided into two groups of 7-bits, each representing the ASCII code as described in Figure 2.15 in your textbook. Starting with the first element, the array contains 2 uppercase letters, followed by 2 lowercase letters, followed by 2 decimal digits (0 through 9), followed by a comma and a space characters. The sequence repeats every 8 characters.

1) Write and test the assembly code that would create three arrays that contain in order all uppercase letters in order, all lowercase letters, and all decimal digit characters, respectively.

2) Write and test the assembly code that adds the constant 0x20 to the uppercase letters array elements (uppercase to lowercase conversion), subtracts the constant 0x20 from the lowercase letters array elements (lowercase to uppercase conversion), and subtracts the constant 0x30 from the decimal digit array elements (ASCII to integer conversion). You should use the ADDV and SUBV SIMD instructions. **521** implements only the first addition, i.e. uppercase to lowercase conversion.