ECE 5362 Machine Problem 2
Due 10:20am Oct 23 (Carmen Online Submission)

This is the second machine problem in this semester. In the first machine problem, we have written microinstructions to implement two simple instructions: **NEG AC** and **EXG AC, X**. In this machine problem, we need to generalize the implementation of **NEG** and **EXG** to work 1) with any of the four registers visible to the programmers (i.e., AC, X, SP, and PC), and 2) for four different addressing modes: register, register indirect, autoincrement, and autodecrement, as listed below. Note that **HALT** must also work. **Please note that the condition codes CVZN must work properly for this machine problem.**

Mode	Name	Assembler Syntax	Operand
0	Register	Rk	[Rk]
1	Register indirect	(Rk)	[[Rk]]
2	Autoincrement	(Rk)+	[[Rk]], then [Rk]+1 → Rk
3	Autodecrement	-(Rk)	[Rk]-1→Rk, then [[Rk]]

Your implementation should be based on what you did for Machine Problem 1. Your tasks are:

1. Generalize the implementation of **NEG** and **EXG** to work with other registers visible to the programmers (i.e., X, SP, and PC). To be specific, in addition to **EXG AC, X**, you should make the machine work for **EXG AC, SP** or **EXG (PC), AC** or **EXG X, -(PC)**, where the source and destination can be any of the 4 registers visible to the programmers. Similarly, in addition to **NEG AC**, you should make the machine work for **NEG X, NEG (SP)** and **NEG (PC)+**, where the destination can be any of the 4 registers visible to the programmers. You should think about how to make your program efficient such that you do not have to handle all the register combinations separately. A hint is that the RAC/WAC (read/write access) interface of OSIAC for the four registers allows us to directly select the register specified in the instruction stored in IR. This had been emphasized in class (rac=2 or 3).
2. Implement the four addressing modes for the **NEG** and **EXG** instructions. For the four addressing modes, the instruction is one memory word, which includes only the opcode word.
3. Write a test program to test your completed microcode file thoroughly. **HW4 provides several test cases for the addressing modes you need to implement**, but you may need to revise the test program a little by removing those addressing modes not required for this MP. When you revise, make sure you change the memory addresses used in those instructions accordingly such that the test cases still work correctly. Based on your analysis of those test cases, you should make sure your implementation can generate the same outputs for the 4 programmer-visible registers, as well as any memory writes and CVZN.

You are suggested to take the following steps: 1) Place the **src** operand in a designated temporary register (e.g., t2) for all the addressing modes; 2) Place the effective address of the **src** in another designated temporary register; 3) Place the **dst** operand and its effective address in designated temporary registers, as you do for the **src** operand; 4) Perform the desired operation with those temporary registers and write the result back to either the dst register or the desired memory location. Note that OSIAC is simplified from 68000. So, as introduced in class, for 68000, **one of the two operands (src or dst) of a double-operand instruction must use the register addressing mode or the source operand is an immediate number**. This rule applies to OSIAC as well and simplifies your designs, because you do not need to consider over-complex instructions like EXG (AC)+, (AC)+.

When you are done, make a copy of your microcode file and call it **mp2-specs.dat**. Be sure that your name is in the opening comments in the file. Then submit your mp2-specs.dat file through Carmen's online submission. Site will be closed exactly at the due time, sharp!

Do NOT submit the **details.1 or summary.1** files, or any test files. Tests will be run on your **mp2-specs.dat** file to verify its correct operation. Keep a copy of the microcode file as you submit it. Be sure to double check and ensure you are submitting the right files. You do NOT get to replace your files after the due time. You also do NOT get your work re-graded just because you submit any wrong files. So, every time after you submit your files, please double check by opening them on Carmen and make sure they are the right files. You can always replace your files before the due time. The new Carmen Canvas system may automatically add -1, -2, -3 to your submitted file every time you submit. **NO worries. Our grader(s) will always use the latest submission for grading.**

You must do your own work without collaboration with any other students. The grader will use some software to check the submissions in order to find any similarities among the submissions from different students.