CS221 Assembly Language

Lab 05: Addressing and Looping

Objectives

- Understanding and using the various addressing modes.
- Using the `loop` instruction to create loops and nested loops.
- Using the `jmp` instruction.

Practice

Use your chapter 5 slides to:

1. Declare an array (called `my_ints`) of type `word` and initialize it with *5* integer values.
2. Create a loop (using the `loop` instruction) for each of the following tasks to sum the values of this array. (slide 49)
   a) Use register indirect addressing mode (use `ebx` as base address register). Store the result in `total_register`. (slides 37 - 39)
   b) Use index addressing (use `esi` as index register). Store the result in `total_index`. (slide 40)
   c) Use index addressing with scaling (use `esi` as index register). Store the result in `total_sindex`. (slide 41)
   d) Use base-index-with scale addressing mode. Store the result in `total_biscale`. (slide 46)

Exercises

3. Printing even and odd numbers

    a) Create an infinite loop using `jmp` to print even numbers. (slide 48)

    b) Create an infinite loop using `jmp` to print odd numbers. (slide 48)

4. Copying a string. (slide 53)

    a) Declare two strings. One has the value “Copy this string” and the other is an empty string with the size of the first string.

    b) Using a loop, copy the first string to the second string using register indirect addressing. Use the registers `ebx` and `ebp`.

    c) Print the two stings.

5. Use a nested loop (slide 51) to print times table starting with 1 and ending with 10. The output should be:

```
1 2 3 4 5 6 7 8 9 10
2 4 6 8 10 12 14 16 18 20
3 6 9 12 15 18 21 24 27 30
4 8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
6 12 18 24 30 36 42 48 54 60
7 14 21 28 35 42 49 56 63 70
8 16 24 32 40 48 56 64 72 80
9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
```