CS 382 Fall 2023 — Computer Architecture and Organization — Lab 5

1 Task: Calling Convention

In this task you will practice calling convention and creating procedures. Things can be **very** tricky in this lab so be careful!

You will write two procedures in assembly: `_uppercase()` and `_toupper()`:

- `char _uppercase(char lower);` receives a single character in lower case, convert it to upper case, and return this new character;
- `int _toupper(char* string);` receives the address of a string, and convert all characters to upper case by calling `_uppercase()`, and return the number of characters converted. Note this function will convert the characters **in place**: it will replace the lower case in the old string with upper case, instead of creating a new string.

Using the data in the starter code, the program should print the following:

```
Converted 10 characters: HELLOWORLD
```

Be Careful...

- When loading a character, or a byte, into a register, the instruction is `LDRB` or `LDRSB`, and the destination register is `Wt` not `Xt`. This is the same when you are storing a character back: the instructions are `STRB` and `STRSB`.

Requirements

- **Note** your code is a **complete** assembly program (not just a sequence of instructions). It should be able to assemble, link, and execute without error and warnings. When executed, the program must finish without problems. If your code cannot assemble, link, and/or execute, you get no credit;
- You must create procedures correctly, meaning following the calling convention discussed in the textbook and class;
- You must **not** use any C library functions other than `printf()`. When using `printf()`, you must use `outstr` defined in the starter code without changing it;
- You must **not** hard code any length-related variables;
- You do not need to write comments on every line, but you're strongly encouraged to do so;
- Put your name and honor code pledge at the top of your code in comments.

2 Grading

The task can be very easy without procedure calls, but getting familiar with procedures and calling conventions are important and therefore exactly the purpose of this lab. We take the proper creation of procedures as equally important as getting the correct output on screen.

The lab will be graded based on a total of 10 points. The following lists deductibles, and the lowest score is 0 – no negative scores:

- **-10:** the code does not assemble, or the program terminates abnormally/unsuccessfully;
- **-10:** the code is generated by compiler;
- **-10:** did not call `_uppercase()` in `_toupper()` and/or `_toupper()` in the main procedure;
- **-10:** used other external functions other than `printf()`;
- **-5:** did not call `printf()` in the main procedure to print expected result;
- **-5:** the procedure was not created properly and/or didn't follow calling convention. -5 points for each of the two procedures;
- **-5:** declared/hardcoded any data that represents the length of the string;
- **-3:** the string is incorrectly modified or not modified at all;
- **-2:** the return value is wrong;
- **-1:** no pledge and/or name.

Earlybird Extra Credit: 2% of extra credit will be given if the lab is finished by Wednesday 11:59PM EST (1 day before the lab deadline). For specific policy, see syllabus.

Attendance: check off at the end of the lab to get attendance credit.

Deliverable

Submit a single `.s` file.