

## LAB #1: Simulator Lab

**Partially due:** Friday October 2nd, in class (preliminary documentation only).

**DUE:** Wednesday October 14th, in class.

---

Using the machine definition and instruction set descriptions given in class, design, code, test, and document a program to simulate the execution of the abstract machine.

### Object Files

Your input will be a file of records as follows:

1. A Header Record

record position 1: H

record positions 2-7: a 6 character segment name

record positions 8-11: a 4 Hex character value denoting the initial program load address

record positions 12-15: a 4 Hex character value denoting the length of the segment

2. Text Records

record position 1: T

record positions 2-5: a 4 Hex character address at which the information is to be stored

record positions 6-9: Initial value of that address, as a 4 Hex character string

3. End Record

record position 1: E

record positions 2-5: a 4 Hex character address at which execution is to begin

Figure 1 shows an example input file. Like all (correct) input files, it starts with a single header record, followed by a sequence of text records, and finishes with a single end record.

Do not assume that initial contents will be given to you for each of the memory addresses in the segment. Thus, the length of the segment given in the header record can be larger than the number of text records in the input file (as in the sample above).

### Requirements

Your program should have three major components: one (the “loader”) to put the input information into the data structures that represent the memory and registers of the machine, one (the

---

```
HSAMPLE30000104
T3000E300
T300156E0
T300254A0
T300314A4
T3004040A
T30056840
T300616C4
T30071261
T300814BF
T30090E04
T300A5020
T300B1003
T300CF031
T300DF025
T31000001
T31010001
T31020001
T31030001
E3000
```

---

Figure 1: Example Input File

“interpreter”) to simulate the operation of the machine as it executes each instruction, and one (the “simulator”) to act as the simulator’s user interface, displaying the state of the machine as appropriate and controlling execution.

The loader must detect and provide appropriate messages under the following error condition:

- invalid contents (i.e., illegal characters in an input record)
- other errors you identify

The instruction interpreter must be capable of executing each of the machine’s instructions as specified. The interpreter should not be capable of stopping execution, unless the HALT instruction is issued.

The simulator must halt instruction interpretation when a user-specified time limit has been exceeded. This time limit should be specified in terms of number of instructions executed. Some reasonable default for this time limit should be provided. In addition, you may also elect to have some I/O errors be fatal.

Your simulator should have at least 3 modes: quiet, trace, and step. In quiet mode, it simulates the execution of the loaded program without interruption (unless the time limit is exceeded, of course). In trace mode, it should generate a trace of execution including:

1. The state of the machine (memory page and registers) immediately after loading but before execution.

2. Each executed instruction, including the memory locations and registers affected or used.
3. The state of the machine (memory page and registers) after execution.

Step mode should be similar to trace mode, except that the user is prompted before each step is taken. You may add other modes as you feel necessary.

## Mandatory Design Review

Each group is required to schedule an approximately 25 minute meeting with me and the grading assistants to discuss your design and to answer questions. All members of the group are expected to be present for the entire design review. The schedule for these reviews will be set during the second week of the quarter. However, please don't hesitate to ask questions earlier than that in class or during office hours, especially if there is a point that appears vague or unintelligible to you.

**A draft of your documentation, in particular the programmer's guide** is due *prior* to your scheduled design review meeting. See the syllabus for the due date of this preliminary documentation.

## Breakdown for the lab 1 grade

Group part:

- User's guide: 20%
- Programmer's guide: 20%
- Coding: 10%
- Your testing: 10%
- Our testing: 20%
- Meeting minutes: 10%

Individual part:

- Peer review: 10%

The documents will be graded based on both their technical quality and their English.