User Manual

Rev 1.0

4/24/18

|  |  |  |
| --- | --- | --- |
| Date | Change | Changes made by |
| 4/24/18 | Document created | Matthew Michaels, Reagan Craddock, Milton Griffin, Michael Farden, Matthew Strenk |

[**Product Details**](#_lqpf45cq2nl8) **3**

[Product Requirements](#_hg2ssrwf2768) 3

[Product Features](#_kfdcl9qkjhze) 3

[**Code and Building**](#_nqh8r0xq2vv8) **4**

[How to Access Source Code](#_wg0zqvz8exb1) 4

[How to Build Source Code](#_wg0zqvz8exb1) 4

[**List of Deliverables**](#_d30h3tz3vpy6) **4**

[**List of Known Problems**](#_d30h3tz3vpy6) **4**

[**Contact Info and Bug Reporting**](#_isf5922dbgq6) **4**

[Contact Info](#_8zd8n8boca40) 4

[Bug Reporting](#_88re93y6bqz) 4

[**Original Memsim Info**](#_nlbwdm2bqd8p) **4**

## Product Details

### Product Requirements

This product is a Multi-Core Memory Simulator that runs a scratchpad and cache simulation based on specific benchmarks and parameters. The product requires the following parameters in order to work as intended:

* Windows OS
* Full sim project
* Correct Trace file(trace.txt)
* Correct cache memory configuration file(memory.txt)
* Correct Scratchpad memory configuration file(spm\_memory.txt)
* CPU with more than 1 core

### Product Features

This product has several memory simulation features. The user has the option to run four types of simulation benchmarks:

1. Hash: a benchmark that generates random 4-byte memory accesses.
2. Heap: a benchmark that performs random insertions and deletions of 4-byte items in a full binary heap.
3. Stride: a benchmark that performs strided 4-byte memory accesses
4. Trace: a benchmark that executes a memory access trace based on a default or user defined trace file.

The user must enter a 1, 2, 3, or 4 respectively to select one of these options. The trace file contains sequences of memory access actions: reads (R), writes (W), and idle time (I). The size is measure in clock cycles. The trace file format is shown below in Fig.1. Fig. 2 shows a screenshot of the command prompt window when the program is started. It provides the user with all of the simulation option 1-4. Option 5 allows the user to exit the simulator.

R<address>:<size>

W<address>:<size>

I<cycles>

Fig. 1: Trace file memory action format.

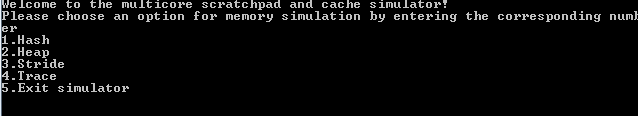


Fig. 2: Command prompt simulation start-up. Provides the user all simulation options.

## Code and Building

### How to Access Source Code

The project was built so that it only includes the executable file for the ADA simulator code. This means that the base code for the simulator cannot be edited. The only files that the user can change are the python wrapper, the batch script, the memory description file for cache and scratchpad, and the trace file. All of these files are stored in the src directory of the project. The rest of the files in the src directory are the files that allow any Windows computer to run the program without having Python installed.

To run the project, navigate to where the project is stored and open the src folder. Next, open a command prompt window from that file location and type “fullSim.exe”. This should run the program, asking for the user to choose the type of benchmark to run. Once program finishes running, it will output the simulation findings to the command prompt window and will also output the results to the simulation’s respective output file. These files are located in the results folder.

### How to Build Source Code

The program is released to the user in a built state, it would only need to be rebuilt, if the user were to make changes to the code. To build the project to run on any Windows computer without installing Python, the user will need to follow these steps:

1. Install Python on your computer
2. Open a command prompt window and type “pip install pyinstaller”
3. Once the installation is complete, navigate to the location where the project .py file is stored
4. Then type “pyinstaller fullSim.py” into the command prompt window
5. This will create all of the files needed to run the program on a different computer that does not have python installed.
6. Make sure that the memory description files, trace file, batch file, and the exe file from the ADA code are included in the newly compiled project.

## List of Deliverables

* Design document
* Project plan
* Project requirement
* Test plan
* Test report
* User manual

## List of Known Problems

* One error received is “'.' is not recognized as an internal or external command, operable program or batch file.”, fortunately though this does not affect the results and is a legacy bug and not an issue with our code
* Mm simulations output error when run, so it was removed as an option for the users.

## Contact Info and Bug Reporting

### Contact Info

* Matthew Michaels, www.linkedin.com/in/matthew-michaels-25951b107
* Matthew Strenk, strenkml, on Github.com
* Michael Farden, https://www.linkedin.com/in/michael-farden-62255ab9/

### Bug Reporting

To report bugs with the application please message Matthew Strenk on his Github account. Matthew’s username on Github that you can use to find and contact him is strenkml

## Original Memsim Info

If you’re interested in seeing more detailed information about the memory simulator, the original Github repo can be found at the following link: <https://github.com/joewing/memsim>. The application can also be found by searching memsim on github.com and looking for joewing/memsim. Inside of the repo is a readme file that has lots of information about how to use the simulator on its own and how to simulate more types of memory than what was provided in our application.