Sara Version 0.5.0 May 4, 2008

Sara is an Apple /// emulator for Mac OS X Macintosh Systems
The latest version can be downloaded from:
http://www.blackcatsystems.com/software/sara.html

Sara is freeware, use it and pass it around to your friends! Amaze them with a demonstration of state of the art personal computing from the early 1980's!

This is an extremely preliminary version of Sara. Use it at your own risk.

It is guaranteed to crash, bomb your system, blow out your hard drives, fry your memory, and shatter your monitor. No refund if it doesn't.

Assuming it does none of the above, you'll be presented with several windows. One for the Apple ///'s screen, one showing the status of the disk drives, and one showing the CPU status (register contents, program counter, flags, etc).

Quick Start:

1.Make sure you have a ROM file, called ROM, in the same folder as Sara.
2. Make sure you have a bootable SOS disk image file, rename it DISK1.do, or after starting Sara, select the correct disk image. A SOS utilities disk is ideal for experimentation.
3. Run Sara.

That's it!

Other Instructions:

Selecting Run from the Control Menu (also command-G for go) will start emulation. Halt (command-H) will stop it. Reset (command-R) will reset the emulated Apple ///.

To use disk images, you can rename them as follows:

DISK1.do Internal drive disk
DISK2.do First External drive disk
DISK3.do Second External drive disk
DISK4.do Third External drive disk

These are the default filenames used for image files.

You are also able to select the images from within Sara. Select the appropriate Open Disk Image... item in the File menu.

It boots the SOS 1.3 System Utilities disk! And it seems to work! You can click on the name of a disk image in the drives window, and change the disk file. If the image file is locked, the disk will be write protected. Otherwise, you can write to it. Once a disk has been written to (is "dirty") the filename is shown in red. If you want to save the changes made, select the corresponding disk from the File menu, and the disk image file will be written to from the image in memory.

Currently a 5MB proFile hard drive is supported. This will be changed to a 10MB, or maybe even larger, if SOS allows it. When you first start your copy of Sara, you won't have a valid proFile image, so just format the .PROFILE volume from within the SOS utilites. The HD data is saved in a file called Profile.dat, by the way, which is automatically updated when you quit Sara.

Right now, it boots the Business Basic disk. The keyboard does work. You can enter commands directly, and they seem to work. However, programs don't run correctly. This is probably a memory mapping problem.

I don't have all of the VIA functions implemented yet, so that's another problem. For example, the shift register and timers. Are these actually used for anything?

The character generator RAM download now works, so the fonts should look correct.

I've added buffering for the keyboard, so if you type too fast, the keystrokes are buffered until the /// software can read them in.

If you have an extended keyboard, then you can use the left option key as the open apple key, and the right option key as the closed apple key. Don't

use the apple/command keys on your Mac keyboard, as they are only used for Mac commands (such as cmd-R to reset the emulator, etc). I change the keyboard's device ID when Sara starts so that the option keys can be read. I reset it back again when you quit Sara, but if your keyboard acts flakey after running Sara, a restart will fix it. Let me know if this happens.

I don't have the alpha lock key implemented yet. I guess the best way to do this is to read the state of the caps lock key, and use that?

Control Menu Commands (most of these don't work yet in the first port to Mac OS X):

Debug:

Turns on debug mode, which can display some information to the text Debug window. Probably not useful to anyone besides myself.

Control Menu Commands:

Reset CPU
Simulates pressing the Apple ///'s Reset button

Disassemble CPU
Disassembles the next 30 instructions, starting at the current PC. Debug mode must be
turned on, or nothing will be displayed.

Init CPU
Initializes the CPU data structure. (You should never need to do this)

Single Step CPU
Executes the next instruction, displaying the instruction executed.

If you want to single step through some code, a suggested procedure is as follows:

Halt the CPU
Jump to the desired address
Start single stepping

Run CPU
Starts running the simulation.

Halt CPU
Stops running the simulation.

Jump to Monitor
Sets the PC to F901, so that the monitor routines start running.

Jump to Address
Provides a dialog box so that you can enter an address, the PC is set to this address.

Set Breakpoint
Provides a dialog box so that you can enter an address, which will become the breakpoint. If the 6502's program counter (PC) reaches this address, the emulation is halted.

Dump Trace Buffer
Provides a dump of the last 30 instructions the 6502 executed. (Note, there is a bug, it doesn't always display the correct trace information. On the stack of things to fix)

There may be some additional menu items from time to time, these are for my test purposes.

Debug
This allows you to examine and edit memory locations while the emulation is running. The format is somewhat similar to the old Apple][monitor. The prompt is an asterisk (*).

By typing an address, the contents of that address are displayed:

```
*45
0045:23
*
```

A range of addresses may be displayed by using the period, such as:

```
*200.205
0200:21 07 26 06 1F 00
*
```

Between the range 2000-9FFF, a 20 bit address may be entered, to allow access to the entire Apple /// memory map:

```
*67483
[6]7483:00
*
```

The first digit is the bank number. So in this case, address 7483 is read, with bank 6 inserted in the switched bank range.

It's also possible to write into memory, this is done using the colon:

```
*0:33
```

This writes 33 into address 0.
20 bit addressing may also be used for writes.

[Hopefully some of those more familiar with the Apple /// addressing and memory map will take some time to poke around memory, to see if I have any memory mapping problems remaining, which may cause some of the weird behavior when running certain programs, such as Business Basic.]

Comments (including those on the user interface) are appreciated to:
cps@blackcatsystems.com

Version History:

0.5.0 (5/3/08):
First version for Mac OS X

0.4.0 (8/23/01):
The ProFile works!!!
Disk formatting now works

0.3.0 (6/28/01):
Able to write disk images back.
Able to change disk images on the fly.

0.2.0 (2/22/98):
Added a real-time debugger.

0.1.0 (12/21/97):
Character generator RAM downloads now work.
Option keys simulate open/closed apple keys.
Added keyboard buffering, so if you type too fast, the keystrokes are not lost.

0.0.5 (12/15/97):
Got enough of the interrupts to work to make System Utilities and Basic (well, part of Basic) happy.
Implemented all? of the Apple /// graphics modes. They seem to work OK.
Fixed the 68K version.
Broke support for the Apple /// Confidence program. (It uses the VBL for timing, which I broke while getting interrupts kinda working) Too bad, because it shows you all the pretty graphics modes that I spent so much time getting to work...

0.0.4 (12/13/97):
Added support for the clock.
Fixed some addressing problems, dealing with 8F Extended Addressing.

Broke the 68K version. Hopefully that will be fixed by version 0.0.5.

Chris Smolinski
csmolinski@blackcatsystems.com
http://www.blackcatsystems.com/