4/16/95

Ok, so now you have it. Version 0.88r, until recently known as 0.872d, which has pretty much been sitting around on my hard drive for six months. And now it's on yours, ha ha!

In case this was a random acquitision, STM is Apple II+ emulator.

STM's essential theory of operation has changed somewhat since the last release. STM itself is not much more than a process scheduler (well, a little more, but I'm working on cutting it down). Most of its "Apple II nature" comes from modules loaded at runtime.

As soon as the interface is documented (see, I say that like maybe somebody else is gonna do it for me), you'll be able to write your own modules to emulate your favorite peripherals.

I'd like to say, right up front, that though I (Kevin) am writing this up, Jim Nitchals has made invaluable contributions to STM. All copyright violations were made by me, before Jim even started working on it.

Running STM

STM requires a mac running system 7 or higher, with some version of QuickTime installed (speed regulation will suffer without QuickTime). At least 3 megs of free ram are required, too. Hey, maybe I should work for Microsoft! Um, there may be other stuff it needs too that I've forgotten, and which I don't check for, and it'll crash if you don't have

them. But basically, if you're living in the '90s, you should be fine.

Before you start STM for the first time (like I really think you're reading this first…), make sure that you have the following things all in the same folder:

- STM application (duh)
- A text file named 'STM config'
- A file named 'prefs' (yeah, someday I'll put it where it belongs)
- A bunch of little files with cool names and spiffy icons (the full suite is: 6502, speaker, video, disk II, keyboard, language card, and paddles)

Now, when you start STM, you'll get just a video window (or, if you don't have the 'prefs' file handy, you'll get no windows). STM has a bunch of potential windows, and unless you have a big monitor you have to pick which ones you want. They're under the Windows menu. When you have them set up like you want them, you can fix them in place by choosing 'Save prefs' under the File menu. Incidentally, the windows do take time to update when they're open, so if you're on a slow machine (030 or lower), be careful how many you keep up.

Some of the modules have configuration options; if so, they'll appear in the sub-menus under the Modules menu. The default option settings are usually pretty reasonable, as behoove default option settings.

If you get a blank video window, and you're not in 256-color mode, switch (unless you don't mind blank video). In general, STM is set up to run in 256-color mode, and you may sacrifice a lot of performance if you're not there.

STM starts up in a paused mode; to start the emulation up, choose Run from the File menu (or command-R).

If you run into something on a menu that isn't mentioned in the docs, feel free to mess with it - everything should be stable enough to not take your hard drive down - but don't be disappointed if it doesn't work quite right; some features aren't terribly stable yet.

STM takes a lot of memory; you can get away with a little less if you don't use the second drive. A tip: if you're short on memory, it takes a lot less to load a disk if you do it through the pop-up menu on the drive than if you use the Open… command on the File menu.

Copyright and Distribution

Now, I'm no lawyer, nor can I afford one. But here are my wishes: you're free to copy STM, distribute it, feed it to your dog, and you owe me

nothing. But don't charge more than a nominal fee for it (e.g., cost of a disk in a users' group), and please keep the whole distribution together, including this documentation.

If you want to put STM on something you're selling (big software collection or something like that), please contact me first. I don't expect I'd have a problem with it, but see the section on copyright issues…

Oh yeah, STM is copyright me, Kevin Lund, 1995. And Jim Nitchals too. Portions are copyright Symantec, Inc., but as soon as I can afford to buy Codewarrior, they won't be.

Common questions:

1) what does STM mean?
Once again, the name has absolutely nothing to do with that nutty diet lady. It's something I saw written on a bathroom wall on my way to MacWorld '87 or '88. Anyway, the name has no relevance to the Apple II world. The full quote, for the record, was "Stop the madness, rock and roll your balls." It probably had some exclaimation points too, but I figure with a sentence like that they're pretty much redundant anyway.

2) Question: why?
I wrote it so I could play “Odyssey, the Complete Apventure” in peace. You see, I have two monitors hooked up to my II+: a Monitor ///, which gives a very nice green picture, and some other composite color monitor, which gives a somewhat shaky color picture. Neither is satisfatory on its own, and it's a drag to look back and forth. Well, Odyssey is mainly in color, for which the color monitor does a fine job, but when you go into a bazaar, it goes into text mode and the screen is almost all white, which makes the color picture skew way to the side. So then I have to look at the Monitor /// (the green one) until I leave the bazaar, Ack!

Configuration

Well, this part really sucks. It's one of the things which I keep meaning to fix, but never do, and that's why this version took so long to get out. So to get it out I set up a major kludge…to be replaced…someday.

If you don't feel like screwing around with STM's configuration, feel free to skip this part (but what's the point of an Apple II that you can't screw around with?).

STM has a number of internal positions for external modules (compiled-in limit of fifty, last time I checked). Positions 0-7 correspond to the slots in a real Apple II; the rest are just to hold other stuff down.

One of these positions must be designated the speed master; the other modules will have their speed regulated according to how well it's moving. It's generally a good idea to make the (a?) CPU module the speed master; most of the others wouldn't even know how.

When STM starts up, it reads its configuration from a file called "STM config". If the file isn't there, it does its best to form a reasonable configuration, but you're taking your chances.

Here's an example configuration file:

```
# Sample configuration file for STM
#
# fields are separated by colons; an asterisk at the end of
# a module name indicates that it will be considered as the
# master speed regulator...
#
language card:0
disk II:6
disk II:5
6502*:9
keyboard:10
speaker:11
paddles:12
video:13
```

Any line beginning with “#” is ignored as a comment. Other lines are scanned for a module name and position number, separated by a colon. The speed master module is designated by putting an asterisk (“*”) at the end of its name (see, I told you it was a major kludge).

So, the above file puts a language card into slot zero and two drive controllers (each with two free drives attached, which is unavoidable) into slots five and six. On top of that, it loads up modules to handle processing (6502, note the asterisk), keyboard input, sound, paddles, and video. The position numbers on these are pretty much arbitrary.

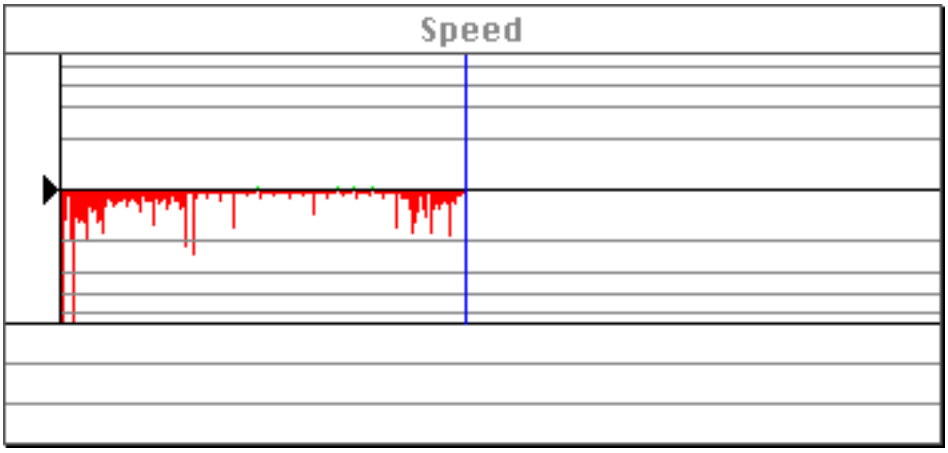
STM's memory requirements will vary according to what modules you're loading; most notably, configuring a zillion floppy drives and loading them up will take its toll.

You can edit this file with any text editor (TeachText or SimpleText should do fine); if you use a word processor, be sure to save the file as text only.

Internal Windows

These windows can give you a little insight as to what's going on in STM. Since STM is really just a task switcher, these windows pertain to timing and scheduling issues.

The Speed Window



This gives you a reading of the absolute performance of STM; more precisely, it tracks the performance of whichever module you've designated as the speed master module (probably the 6502 module).

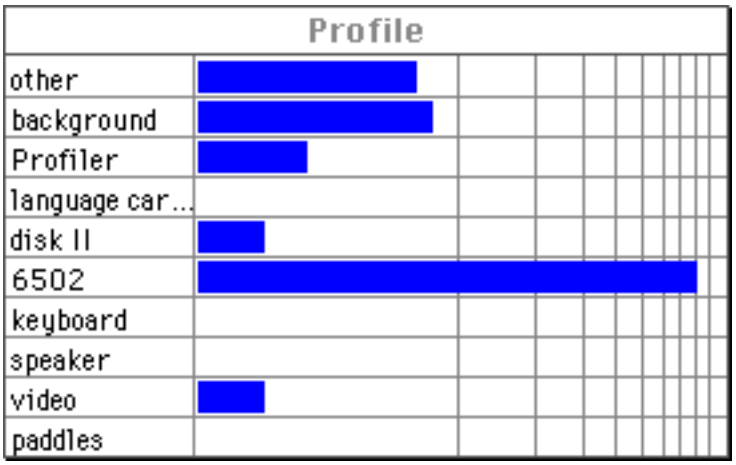
The horizontal lines represent speed on a log scale; each line represents 1.023 Mhz, with the middle (black) line being the 1.0 reference (i.e., if regulation is sticking to that line, you're running at a 1.023 Mhz equivalence, while if you were at the next line down you'd be running at 0.5115 Mhz; the first two lines up are 2.048Mhz and 3.069 Mhz, respectively, and I hope that you get the picture by now).

The black triangle on the left represents the target speed for the emulation. It starts out on the midline, for a default 1 Mhz emulation, but you can move it to wherever you want. The colored lines indicate actual performance; points below your selected regulation speed are colored read and those above are colored green. My computer is a IIsi; as you can see, I don't get a lot of green.

Often, you'll see a regular pattern of small downward spikes; these seem to be caused by background applications wanting little chunks of time every now and then.

By choosing 'Speed messages' on the FIle menu, you can see whatever performance statistics the modules are reporting back, displayed in the lines under the graph. This can really slow things down, as the modules tend to report back often.

The Profile Window

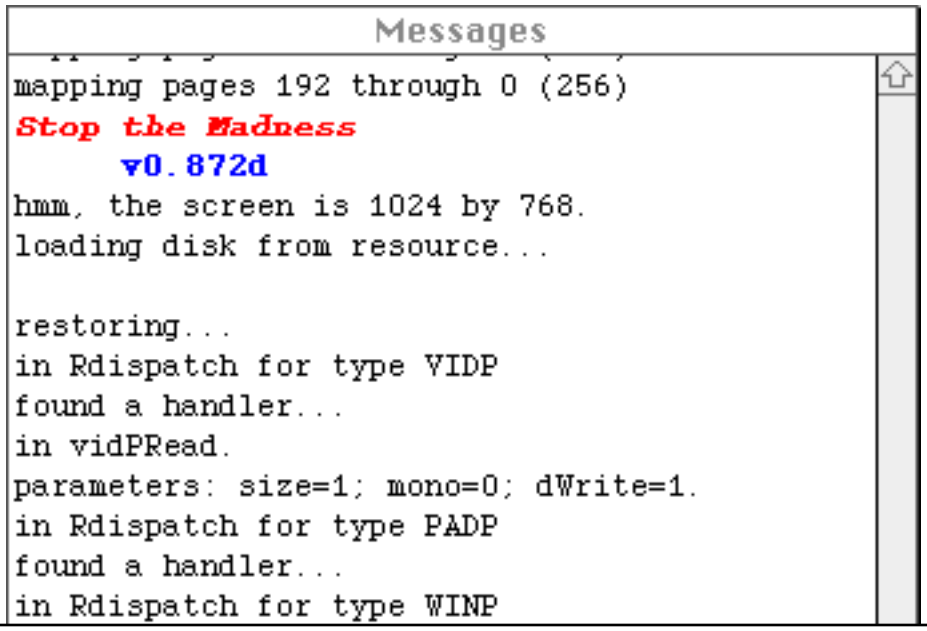


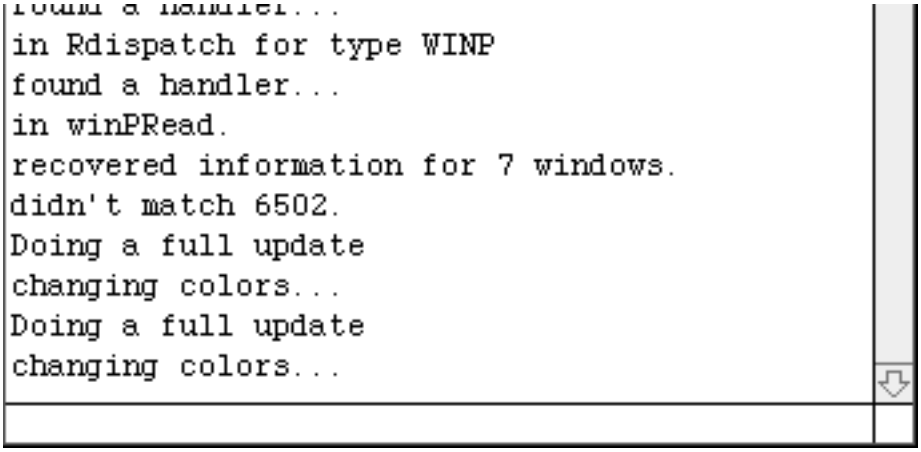
The speed window tells you how fast the emulator is running; the profile window shows how it's spending its time. Each horizontal segement represents some chunk of the whole emulation scheme; again, performance is plotted on a log scale. For this graph, each vertical line represents 10% of STM's time. Background tasks, in this example, are taking about 9% of available time, and the 6502 is getting about 80%.

Three of these categories are internal to STM. 'Other' just represents all of the time which isn't accounted for by all of the other categories. 'Background' represents time given to applications running in the background. 'Profiler' represents the time needed to compute and draw the stuff in this window; close the window, and this drops to zero (but you'll just have to take my word for it).

All of the other categories represent external modules. At the time that this window was captured, the 6502 module was the big winner, which should come as no surprise.

The Messages Window



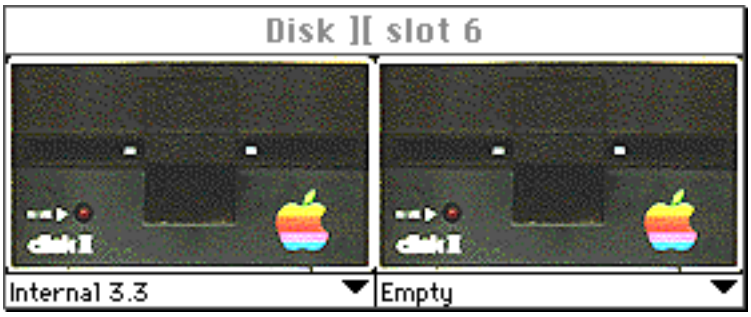


This is just a sort of catch-all messaging system; all sorts of (usually) trivial messages come up here. You can control the types of messages to some degree via the 'Messages' menu.

External Modules

STM comes with a number of external modules which, when used together, enable a fair emulation of an Apple II+. Here's a quick rundown on them:

Disk II



This, of course, is a floppy drive emulation module. It emulates the standard Disk II controller card, with two drives attatched. The lights work, but no sound effects yet.

Each drive displays the name of the currently loaded disk image; above, the drives are in their default state with drive 1 containing the built-in image of the 3.3 system master, and drive 2 empty. The first (highest slot #) disk II module will automatically load the 3.3 system master image into drive 1 at startup.

Each drive has a pop-up menu, accessed by clicking in the lower area of the window (where the disk titles are). This menu will let you load and save disk images, as well as exchange images between drives and empty drives.

If the Disk II module gets passed a disk image from STM as the result of choosing 'Open…' from the File menu, you'll get a dialog box identifying the module by slot number, and giving you a choice between loading the image into drive 1 or drive 2. A third choice is to skip the controller, in which case STM will continue to search for another module to handle the image. This is useful if you have multiple drive controllers configured.

The disk II module supports two types of disk images. The first is the more-or-less standard '.DSK' format, which is just a sector-by-sector dump of a 5.25" disk. The second format, which is the only format the module will save images into, is a pre-nibbleized format. This format is bulkier than the '.DSK' format, and has no cross-platform compatibility, but is more flexible, allowing non-standard formats and 1/4 track resolution.

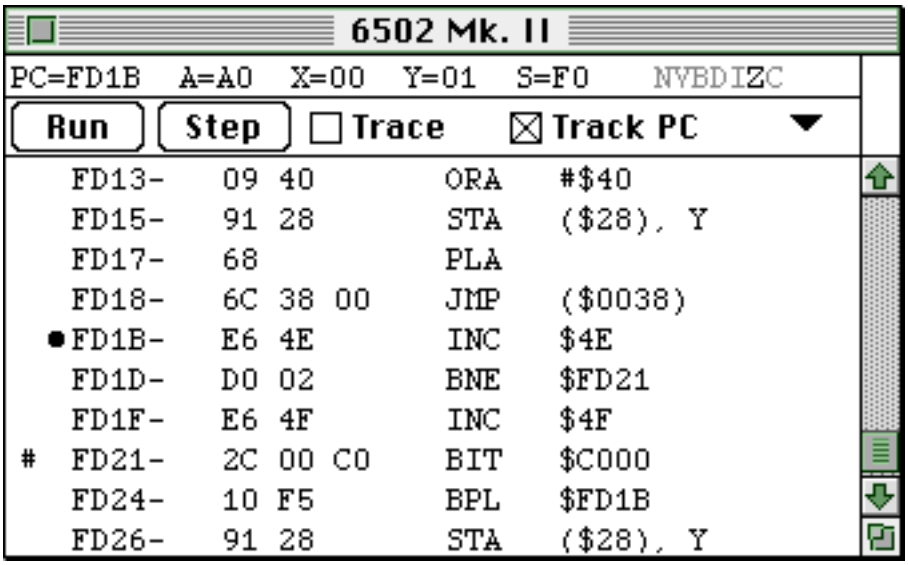
By the way, the picture looks best with your screen brightness way down and contrast way up, just the way I like it.

Language card

This is a standard 16K memory card. It consists of four 4K banks mapped into the top 12K of STM's memory map. The window shows which bits of memory are enabled. The left column of boxes represents pre-existing memory (external to the language card module; in this case, the Apple II ROMs), and the other two columns represent the four banks on the language card. Boxes are red when read-enabled, green when write-enabled (for all the good it'll do you to write to ROM), and yellow when they're both read and write enabled.

In the window above, the module is enabled, read-only.

6502



Ah,the big cheese itself! This emulates a 6502 (as opposed to, say, a 65c02). The window gives a bit of state information about the processor and shows a disassembly of memory.

There are a few controls. The 'Run' button just mimics the 'Run' option on the file menu. When the processor is running and this window is visible, it's updated a few times per second with the latest information. If the 'Track PC' box is checked, the disassembly portion of the window is scrolled to keep up with the program counter (but only at these periodic updates). If the 'Trace' button is selected, the window is updated after every instruction* is executed; this carries a stiff (huh huh) performance penalty. The 'Step' button just single-steps the processor.

The black dot shows the current location of the program counter; the an octothorpe next to aline means that it has a breakpoint set.

A few comments about breakpoints: breakpoints are implemented by inserting a BRK instruction into the instruction sequence. When a BRK is hit, the emulator looks up the

location to see if it has a breakpoint set; if not, the BRK is processed normally.

If the BRK coincides with a breakpoint, execution halts. When it resumes, the byte replaced by the BRK is temporarily patched back in so that execution continues as though the instruction were intact.

The disassembler in the emulator knows about this tomfoolery, and will show you a disassembly of the code as it will execute (that is, you don't see the BRK). The BRK is there, though, and will show up if you look for it via the Apple II monitor, or whatever. If you really want to see the breakpoint BRKs, there's an option on the menu ('Conceal breakpoints') which you can un-check.

All of the registers on the top line can be edited by clicking on them. When you start to edit one, execution halts. If you press the enter key (as opposed to the return key) after entering a new value, execution resumes (or starts, whatever). You can also edit the hex digits to the left of the opcodes.

The black triangle marks a pop-up menu which is identical to the 6502 submenu of the Modules menu.

(*) As a nifty speed optimization, sometimes the emulation executes two instructions at a time; these sequences are stepped/traced as though they were single instructions. Breakpoints can still be set on either instruction of the pair, as the BRK instruction prevents the double-execution.

Paddles

You have two basic options on the paddles: keypad or mouse.

Keypad paddles use the intuitive numerical square-directional setup; the '5' key centers the paddles if you don't have auto-centering turned on. This may not work if you don't have a standard extended keyboard.

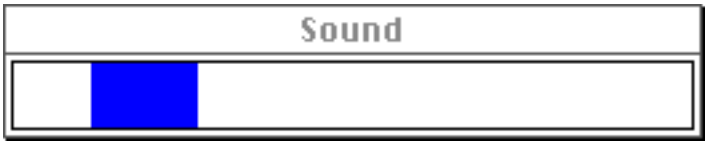
There are a couple of extra options for the mouse paddles; you can either have the paddle positions computed from the mouse's position on the whole screen or from its position within the frontmost window.

Paddle button 0 is tied to either the command key or the mouse button; your option. Button 1 is the option key, and button 2 is the shift key.

Keyboard

There's not a whole lot to say about this one. The delete key (again, your mileage may vary if you don't have an extended keyboard) doubles as a reset key; i.e., control-delete resets the emulator. Command-control delete causes a reboot. Just like a //e, whee!

Speaker



This window is definately more of a debugging aid than anything else, but it can be a little bit amusing if it's a slow day. The outlined bar in the window represents the sound module's buffer; the blue portion shows areas which are full of buffered sound.

Two options here deserve some explaination:

Normally, sound is stopped whenever there's no sound actually being produced; this saves the mac thr trouble of playing a silent audio stream. It also can cause a bit of a popping sound whenever sound begins to play, so if that bothers you, you can choose 'Sound always runs' to keep the channel going.

Another problem that can pop up is that sound can be a little choppy; this is especially a problem on slower machines, and is a result of the fact that the speaker module isn't actually running at the same time as the 6502; they take turns. If you're watching the sound buffer window, you can see the problem occur when the trailing end of the blue bar briefly catches up with the leading end. If you turn on the 'Hold sound momentarily', the module will wait until about 1/2 second of audio is "in the pipe" before it starts to play it, thus giving that leading edge a bit of a head start. This gives the system a little leeway, and can help keep short sounds together (at the cost of putting sound a bit out of sync with the rest of the emulation).

Video

The video module reuires that your monitor be in 8-bit color mode; if you're not, everything will work fine, but you won't get a picture.

The only option on the video which may need exaplaination is 'Bypass QD'. On some (most) machines, choosing this option will speed up video displays, giving you a faster emulation. It's possible, though, that this option will be incompatible with a system, or may actually slow things down.

An important note on bypassing Quickdraw: it only works when the video window is frontmost & selected; otherwise, copybits is used to get the image onto the screen.

Making disk images

The following is an chapter 5 excerpted from the "COMP.EMULATORS.APPLE2: FAQ][v2.5", quoted with permission (hey, that's a change).

****

5) Disk-image creation, formats & conversion, modification

Making Disk-Images

There are three primary methods (and associated software) which can beused to create and transfer disk-images from an Apple][(of any persuasion) to the required platform (Amiga, Macintosh or IBM-PC). To transfer ROMs or text files (or indeed any file from an Apple][), just substitute those files in place of the disk-images in the following examples.The first method is primarily for Macintosh users, or people with access toany type of Macintosh.

(i) A must for anyone creating disk-images is "dsk2file.zip" by Ron Kneusel(rkneusel@carroll1.cc.edu), which contains a small Applesoft Basic/Binary program which MUST be saved/compiled under ProDOS. When run on an Apple][the program will read the contents of a 5.25" (unprotected) disk and write them to a specified path (this can be into your Apple's RAM, or onto an 800K ProDOS 3.5" disk) as a disk-image. From there, take the ProDOS disk and insert it into a Macintosh running either the "ProDOS File System" extension, System 7.5 (with standard DOS-mounting software) or use Apple File Exchange. Then, it is a relatively simple matter to either transfer the file to the PC (see below) or set the File Type and Creator information for the Macintosh emulator (DSK5/A2EM).

Alternatively, use a Macintosh LC which has the IIE card attached (onceagain using the "ProDOS File System" extension). Place your 5.25" disk intothe attached drive and use the "dsk2file" program again, this time specifyingeither the Macintosh hard-drive, RAM, or the 3.5" floppy as the path. Onceagain, from there the disk-image can be transferred to the PC or used on theMacintosh.

It is a great deal easier to transfer disk-images from the Macintosh tothe PC (via 720K or 1.44Mb DOS disks) or to the Amiga (via 720K DOS disk) than it is from the Apple][to either PC or Amiga. On the Macintosh, useeither PC Exchange software (which allows PC 1.44Mb disks to be inserted into Mac HD disk-drives), Apple File Exchange, or System 7.5 (with standard DOS-mounting software). On the PC, use Macsee or Mac-Ette (both Shareware) or Macindos, which all read 1.44Mb Macintosh disks. On the Amiga, use Dos2Dos (KS 1.2/1.3) or CrossDOS (inbuilt KS/WB2.1) to read 720K DOS disks in the Amiga 3.5" drive. NOTE: Macintosh 400/800K disks are NOT compatible with either Amiga or PC drives. Always ensure Binary/Data translation (not Macbinary)!

PROGRAM LOCATIONS:
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/dsk2file.zip
ftp://ftp.apple.com/software/aii/lc.iiecard/iie-startup-disk-image.hqx
(the "ProDOS File System" is part of the disk-image - use Disk Copy to extract).

(ii) By far the most common way of creating/transferring disk-images is toconnect your Apple to either a Macintosh or PC via null modem cable. Thefollowing files: "a2pctr11.zip" by Nye Liu (nyet@halycon.com), "senddisk.zip"by Rich Williamson (glitch@eskimo.com) and "adt120.zip" by Paul Guertin(guertinp@iro.umontreal.ca) contain Apple programs which read 5.25" disksand then transmit that information via null modem cable or network. If you area Macintosh user, try "MacADT120a1s.sit.hqx" and "MacADT1.2.0d1.sit.hqx"(upgraded MacADT executable) by Hideki Naito (PBCO3243@niftyserve.or.jp) -this gives a better Macintosh interface but still requires the Apple][software that is included with "adt120.zip". On the PC or the Macintosh, use aterminal program to capture the disk-images. The precise setup of the cabledepends on your computers - older Apple][s require a serial card, whereas the//c and //GS have an inbuilt serial port. Consult the manual that came withyour Apple and determine the serial connection required, as well as the serialconnection on the computer you are transferring the data to.

If you don't have a serial card on your Apple, try "ap2222pc.zip" by ClayChang. This program will transfer disk and sequential text files between thePC and Apple using the Apple game I/O port and the PC parallel port.

PROGRAM LOCATIONS:
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/adt120.zip
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/a2pctr11.zip
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/senddisk.zip
ftp://cassandra.ucr.edu/pub/apple2/incoming/MacADT120a1s.sit.hqx
ftp://cassandra.ucr.edu/pub/apple2/incoming/MacADT1.2.0d1.sit.hqx
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/incoming/ap2222pc.zip

(iii) The third method of transferring disk-images from the Apple][requiresa serial card and a modem on the Apple][, and a modem on the platform you aresending to (this method can be used for long-distance transfers). The program"diskread.bsc" contains a BASIC program by Damon J. Rand(D.Rand@cantva.canterbury.ac.nu) that will create disk-images and allow thefile to be segmented if you don't have a 3.5" drive. Alternatively, "RTRK" byAndrew Kingdom (agk@ausom.oz.au) will write disk-images from 5.25" disks tolarger volumes. Currently it is only available as an encoded binary on"comp.emulators.apple2". Alternatively, use ShrinkIt to make a compresseddisk-image which can be decompressed on either a PC or Unix system using"Nulib". For further information on Apple][compression, check the"FAQ_comp.sys.apple2".

PROGRAM LOCATIONS:
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/diskread.bsc
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/nulib.zip

ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/nulib-go32.zip

Disk-image formats & conversion

AMIGA

The Apple 2000 emulator on the Amiga uses three disk-image formats (thissection is excerpted from the "Apple 2000.doc" file included with theApple 2000 package).

1) DDD - Dalton Disk Disintegrator archives (DDD was a common diskcompression utility for the Apple). The emulator automatically decompressesthem. Archives of this format can be saved on the Amiga and transferredback to the Apple.

2) .DISK - Filenames with a .DISK suffix are raw disk-images with nocompression. They are capable of storing images of non-DOS and copy-protected disks, but are about 220K in size. According to thedocumentation, this format is to be phased out of the emulator.

3) .PROG - Filenames with a .PROG suffix are executable files; these aresingle files that were runnable from Apple DOS 3.3/ProDOS and did notrequire any disk access thereafter. These files now do not even requirebooting any Apple disk and are simply loaded into the appropriate Applememory areas and instantly started.

As to using the "standard" 143360 byte disk-image (see below) with Apple 2000,they MUST be fully translated and decompressed - especially with regard to the128 byte header added by Binhex 5 (.bin) on the Macintosh. Then rename the143360 byte disk-image with a .disk extension.

These Amiga formats are included for the sake of completeness - no disk-images are stored in these formats on FTP sites (except for Apple][archives,which contain Applesoft/Basic files and can be used with the .PROG extensionon the Amiga).

MACINTOSH/PC/UNIX

In the area of Macintosh/PC emulators, there are four main types of disk-images. The first three are also covered in Part 13 of the "FAQ_emulators.ibmpc.apple2" and the last in the "SIMIIE.DOC" file included as part of theSimIIe package. THE FOLLOWING INSTRUCTIONS APPLY TO UNCOMPRESSED DISK-IMAGEFILES ONLY! Disk-images CAN be copied between platforms and they WILL work witha variety of emulators. ALWAYS ensure BINARY translation between platforms!

1) DOS3.3 Order (DO) - The most widely-used disk-image format, used byApl2em, Applewin and Stop The Madness. It usually has a .DSK extension.This image is 143360 bytes in size.

On the MACINTOSH - To use this format with Stop The Madness, ensure that thefile is downloaded or transferred in binary - not Macbinary - mode (and is143360 bytes) and then set

the file type to DSK5 and the creator to A2EM usingResEdit, UUlite, etc. A small system extension called "Snitch" is of extremeusefulness here - it adds the ability to alter file type/creator informationfrom the (System 7) Finder's Get Info window. STM will NOT recognize the disk-image unless it has the correct file type/creator, even if it is of thecorrect size and is a binary file. Alternatively, try Easy Convert, which willautomatically set the file type and creator information.

PROGRAM LOCATIONS:
ftp://cassandra.ucr.edu/pub/apple2/converter.hqx
ftp://cassandra.ucr.edu/pub/apple2/easy.convert.1.0.sea.hqx
ftp://sumex-aim.stanford.edu/info-mac/cfg/snitch-101.hqx (INFO-MAC & MIRRORS)

On the PC - To use this format with Apl2Em or Applewin, ensure that the fileis downloaded or transferred in binary mode (and is 143360 bytes) and that ithas an eight character filename and .DSK extension. The current version ofApl2Em requires that the D1 and D2 disk-images have default names (usuallySYSTEM.DSK and BLANK.DSK respectively), so you will have to rename yourdisk-image files to use them. Applewin can actually read .dsk files of 143488 bytes (.bin files) but for compatibility with other emulators it is better to translate the files correctly prior to use. No other form of conversion is necessary.

2) ProDOS Order (PO) - Disk-image format which is the same size as the DOformat with its sectors in a different order. This format is used by someUnix emulators, but is rare on the Macintosh. This image is also 143360 bytesin size.

On the MACINTOSH - PO to DO: The application Easy Convert will re-order thesectors of a disk-image into DOS3.3 Order. The file must be downloaded inbinary mode, and then processed by Easy Convert. This will automatically setthe file type/creator attributes upon output to DSK5/A2EM.

PROGRAM LOCATIONS:
ftp://cassandra.ucr.edu/pub/apple2/easy.convert.1.0.sea.hqx

On the PC - PO to DO: ProDOS order disk-images are result of extracting Apple][ShrinkIt archives on a PC using Nulib (compiled from Unix source code). Tore-order the sectors into DOS3.3 order, user the Mapper program.

PROGRAM LOCATIONS:
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/mapper.zip

3) Nybblized DOS3.3 Order (NDO) - The use of the term "dos order" isincorrect in regard to this format, since the tracks are organised accordingto a nibble editor. This format is used by Applemu. This image is 232960bytes in size. Virtually no disk-images are stored in this format onFTP sites. Information for converting NDO back to DO is included solelyfor completeness.

On the MACINTOSH - There is currently no way to convert either from or to NDOon the Macintosh.

On the PC - DO to NDO: To convert PO disk-images to NDO (for use withApplemu), use the Em2Emu program.

NDO to DO: Use the program Emu2Em to re-order this format backinto the "standard" 143360 size.

PROGRAM LOCATIONS:
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/em2emu.zip.
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/incoming/emu2em.zip.

4) SimIIe (IIE) - This format is used solely by SimSystem IIe, and isrecognized by a .IIE extension. This image is 143390 bytes in size (SimIIeadds a header to the file amongst other things). Since no FTP sites storedisk-images in IIE format, there is currently no need for a program to convertIIE back to DO or NDO formats.

On the MACINTOSH - There is currently no way to convert either from or to IIEon the Macintosh.

On the PC - DO to IIE: Use Dsk2iie to convert disks from Apl2Em (DO) formatto SimIIe format. SimIIe utilities, including "dsk2iie", are available in the"SIM2DU10.ZIP" archive in the same directory.

PROGRAM LOCATIONS:
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/incoming/dsk2iie.zip
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/incoming/SIM2DU10.ZIP

Modifying Disk-Images

Applications exist which allow the incorporation of single DOS 3.3 files -such as those found on "pure" Apple][FTP sites or in the Apple binarynewsgroup - into disk-images. Programs such as "VIEWDISK" and "dsk_in"/"dsk_out" can write Apple files into disk-images, and extract same to MSDOSfiles. On the Macintosh, use "A2D33Uti098a1.sit.hqx" - this offers the abilityto transfer different types of Basic (binary) files as well as text files.These applications are of vital use to use who do not possess the ability tomake disk-images on an original Apple][; finding a single Applesoft file onan FTP site and using it with an emulator is no longer an insurmountableproblem (of course, the Amiga A2000 emulator can use Apple][files outsideof disk-images anyway!)

PROGRAM LOCATIONS:
ftp://cassandra.ucr.edu/pub/apple2/incoming/A2D33Uti098a1.sit.hqx
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/dsk_in.zip
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/dsk_out.zip
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/incoming/VIEWDISK.ZIP

To catalog a disk-image without running an emulator, try "catalogger" on the Macintosh or "dsk_cat.zip" on the PC.

PROGRAM LOCATIONS:
ftp://cassandra.ucr.edu/pub/apple2/incoming/catalogger.sit.hqx
ftp://{foghorn|wilbur}.stanford.edu/pub/apple2/dsk_cat.zip

**********---Alex Maddison
Address: PO Box 6399(amaddiso@extro.ucc.su.OZ.AU) North Sydney
Shoppingworld(maddison@freenet.fsu.edu) NSW. 2060
Australia--"Blackie, where'd you get your license from? A cereal box?""Two clues: my driving instructor had a pitchfork, and he wasn't a farmer." - Nightmare Cafe

Copyright issues

STM contains a couple of things that aren't copyright me. Notably, STM itself has ROM images and the Disk II module has an image of the DOS 3.3 system master disk. These items are the property of Apple Computer, and I'm including them without permission. If you personally own an Apple II, I think you're in the clear. If not, and this bothers you, then trash this whole mess. Unless you work for Apple's legal department, in which case label it "Exhibit A".