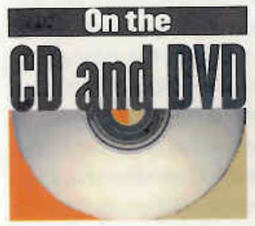
Once, cutting edge computer technology looked like this...

An Apple a day

Simon N. Goodwin continues his series on emulators by gathering a bushel of Apple devices for Linux.



Lots of emulator sources and some sample][and GS disk images are on our cover disc. comp.sys.apple2 is the main Apple][discussion group on Usenet, alongside sources, emulators and binary PD groups within comp...apple2.

Before the Mac, Apple made their fortune with 8-bit computers based on the 6502 processor. The Apple became **][**, gained a plus, then enhanced into the **][e** and became portable with the **][c** and **][c+** versions. All these are emulated by Linux, as well as later 16-bit **GS** 'multimedia' models.

Back when most computers gave mono character-mapped displays, the Apple had bit-mapped graphics and colour, albeit in low res or through an eccentric scheme where the colours available for each pixel depended on the display co-ordinate. When most home computers were cassette-driven the Apple had floppy disks, though in GCR format unlike FM and MFM standards. Apple emulation is worthwhile but not trivial. Apple's first machine, the Apple 1, was a board designed by former HP engineer and Atari *Breakout* game designer Steve Wozniak. Some 200 sold to enthusiasts at $666.66 each, in 1976. *Pom* is a portable Apple 1 emulator, written in Java, on the LXF cover disc.

Red Apple

A dozen emulators for UNIX run programs for the follow-up Apple][. This propelled Apple into the *Fortune 500* list of top US companies in just five years. The much-cloned Apple][boasted a built-in power supply and a proper keyboard and case. A pop-off lid revealed eight expansion slots, sockets for from 4K to 48K of on-board memory, and up to 16K of ROM.

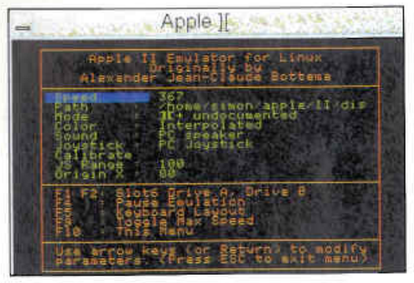
The initial Apple][had integer BASIC and a monitor, dissembler and mini-assembler in ROM, plus Sweet16, a 16-bit virtual machine packed into just 300 bytes of 6502 code. The firmware was exceptional, with syntax checking on entry, line and variable tracing and 100 character identifiers, but hampered by the lack of floating point arithmetic and multi-dimensional arrays. The end of the Seventies saw Apple mutate from a hobbyist operation into a real business. In the Apple **][+** Wozniak's Integer BASIC gave way to Applesoft, modelled on Microsoft's MBASIC for Intel and Zilog systems; the original red Apple][manual, with its matrix listings and hand-drawn diagrams, was replaced with smart ring-bound volumes. In business terms these were big steps, though hackers were less impressed. The hardware was well designed for such expansion and configurability, and emulators offer both options. Apple's killer application was the first spreadsheet, *VisiCalc*, which make micros viable as business systems by doing things no other computer of any size could manage, despite significant limitations. The Apple][offered no more than 48K RAM, a 40 column display all in CAPITALS, and only two arrow keys, so you had to press space to toggle between vertical and horizontal movements.

PICTURE: AL LUCKOW

Steve Wozniak — brilliantly designed the Apple 1 and][and was tempted back to the company by the GS development.

Such flaws were fixed by hardware extensions. The open architecture was exploited by Microsoft's Softcard, which included a Z80 processor and extra RAM so that Intel-architecture code could run on Apples. The Apple **//e** added RAM, keys and display hardware, which most Linux emulators also support. The Apple 3 flopped in 1980, leading Apple to focus on the radical architecture that spawned the Lisa and Macintosh range we described in issues 18 and 19. But 8-bit Apple **//e** production continued until 1994.

The portable Apple **//c** arrived in 1984, followed in 1988 by the **//c+** with a faster processor and 3.5-inch floppy drive, lacking slots, but with most of the common Apple **//e** expansion on board. Capabilities peaked with the Apple **IIGS**, launched in 1986 in response to 16-bit machines like the ST and Amiga. The GS is fully compatible with old Apple software, but has a 65816 processor, like the Super Nintendo, with 16-bit registers and 24-bit addressing. The hardware offers 12-bit colour up to 640 by 200 pixels, and 32 channel audio by synth and soundcard experts Ensoniq. A 1989 update fitted 1Mb RAM and doubled the ROM, but marked the end of the line for 8-bit Apples. But we can emulate them, from first to last, on Linux.



```
Apple II Emulator for Linux
Originally by
Alexander Jean-Claude Bottema

Speed     : 367
Path      : /home/simon/apple/II/dis
Mode      : ][+ undocumented
Color     : Interpolated
Sound     : PC speaker
Joystick  : PC Joystick
Calibrate
JS Range  : 100
Origin X  : 80

F1 F2: Slot6 Drive A, Drive B
[illegible] : Pause Emulation
[illegible] : Keyboard Layout
[illegible] : Toggle Max Speed
F10 : This Menu

Use arrow keys (or Return) to modify
parameters. (Press ESC to exit menu)
```

Menus make XApple2 much the friendliest Apple][emulator for Linux.

There are many issues common to all emulators. By default the Apple][boots from slot 6, drive 1, specified by appending ,S6, D1 to DOS commands. Slot and drive numbers become the default until new ones are stated.

PR #6 boots from the floppy controller card in slot 6, then CATALOG gives a directory listing. BASIC programs you can LOAD are marked A for Applesoft and I for the old ROM Integer BASIC. T means text and B marks binaries you can BLOAD to memory. Original systems used DOS 3.2, boosted to 140K capacity by DOS 3.3, later supplanted by the scalable ProDOS. Most emulators use '.dsk' files 143,360 bytes long. These are packed images of the data in the drive, which 'nibblizes' three data bytes from each four bytes on the disk. Copy-protected and DOS 3.2 images must use the raw nibble format.

The Linux program *AFID* can show you the 'catalog' and free space inside a '.dsk' and read and write individual files therein. It has a simple command-based interface, and can convert between Apple and UNIX ASCII variants and ProDOS and DOS 3.3 sector order. The reorder command may be vital to see all the files in a disk image which might otherwise seem mostly empty. The emulator code and archive names make them hard to tell apart unless you refer to Table 1 **(below)**. The second table notes speed and facilities in the best emulators; an authentic 280x192 pixel screen is easily lost in a big X display. Most of these require an 8-bit X display; if you get a blank window or segfault, adjust XF86Config and restart with Ctrl Alt Backspace.

These emulators require copyright system ROM files which Apple do not make freely available. Apple ROM images are included on the original DOS3.3 System Master Disk, in the "FPBASIC" and "INTBASIC" binary files. You may still need a real Apple to read these, as the disks are in GCR format rather than the PC standard FM. Amiga hardware can read them, though from AmigaOS rather than Linux 68K. CatWeasel ISA cards provide similar capabilities for PC hardware and can even read the 800K ProDOS speed-zone format.

The ProDOS emulator documentation lists five commands that copy the ROM from memory to disk on a real Apple **//e**. The A2 snarf document gives similar tips for older Apple ROMs and disks. Either way, you'll need to transfer the data over a serial link or use the Asimov FTP site. This table helps you identify Apple emulators from their initial authors, archive names and launch commands, and check your display depth is suitable."

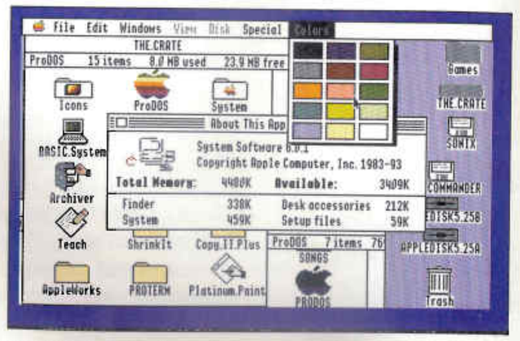
Apple][Emulators

Xapple2

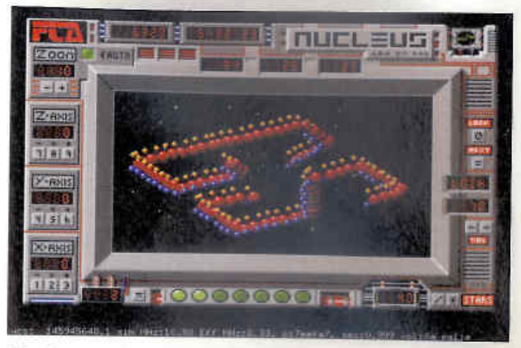
xapple2 has configurable speed, supports joysticks and undocumented opcodes used by some games, and a DOS 3.3 disk image database. The source compiles easily and uses **~/.apple** to locate ROM and disk images and set itself up. A Debian package is also available. Versions drive 40 and 80 column 8-bit X window displays and SVGAlib in 320x200 pixel mode. The 40 column view is tiny in X but scaled nicely in the version with 80 column capability. Three graphic settings fiddle hi-res colour artifacts for speed, clarity and authenticity.

Function keys control the emulator. F10 calls up menu overlays. Apple][mode is so authentic that keys appear in the Apple places, not where a PC keyboard would put them, but F5 shows the mapping and in Apple **//e** mode keys work as you'd expect from their legends. F1 and F2 insert and remove .dsk or .nib files, which may be gzipped, or renamed .do files. Press 'w' to write-enable drives. *Xapple2* can use .dsk or raw .nib disk images, gzipping if necessary, and supports renamed .do files. Simplistic drive emulation does not fool disk protection schemes and stops ProDOS formatting disks, but the package includes blank images and other examples.

Xapple2 works well on PC hardware, but is x86-specific and lacks support for serial ports or the obscure medium resolution Apple **//e** double GR mode. There are plans for an SDL version that will support more display depths. A patch lets you use '/dev/dsp' rather than the PC clicker for beeps, boosting audio quality and timing.


File Edit Windows View Disk Special Colors
THE.CRATE
ProDOS 15 items 8.0 MB used 23.9 MB free
Icons
ProDOS
System
BASIC.System
Archiver
Teach
ShrinkIt
Copy.II.Plus
AppleWorks
PROTERM
Platinum.Paint
About This App
System Software
Copyright Apple Computer, Inc. 1983-93
Total Memory: 4480K Available: 3409K
Finder 338K Desk accessories 212K
System 459K Setup files 59K
Games
THE.CRATE
SOMTX
COMMANDER
DISKS.258
APPLEDISKS.258
Trash

The GS desktop is reminiscent of a low-resolution colour Macintosh.



The impressive *Nucleus* demo, on our CD, is controlled from the numeric pad.

A2

Debian 2.2 compiled *A2* once I edited **curses.h** to remove the **'sys/'** before **'termio.h'**. *A2* expects DOS 3.3 disk images, so mapper.c, remaps ProDOS sectors to suit. It boots into the Apple ROM monitor, but you can access *A2's* friendlier one by pressing tilde (~) at any time. Then 'help' gives a list of commands; to boot from the sample disk image, type **'insert D1'**, **'reset'**, **`continue'**. This gets you to the **']'** Applesoft BASIC prompt. PR #6 boots from drive 1, slot 6. The command line option **'-i'** starts with the old ROM, **'-c'** starts the *A2* rather than the Apple monitor, You can append two disk image names, so after:

```
./a2 -ic D1-image D2-image
```

'continue' leaves you in integer BASIC with two disks inserted. *A2* works well, even in a console, but the lack of graphics, sound or paddles limits its appeal.

Portable technology was core to Apple thinking.

ProDOS

In 1984 Apple replaced their original flat-file DOS 3 with ProDOS, a disk operating system with a much cleaner machine-language interface and a hierarchical filing system with »

UNIX-friendly Apple emulators (table 1)

Archive	Model	Command	Author	Display
a2	2	a2	Rick Skrenta	Text
ap2e	2	apple2e	Randy Frank	Text
apple2	2	apple2	Peter Koch	8 bit
apple2-emul	2	xapple2	Alexander Bottema	8 bit
applelet	2	appletviewer	Steven Hugg	Any
emulator	2	apple2	Philip Stephens	8 bit
II+simulator	2	Apple.II	Ben Koning	8 bit
kegs.0.60	GS	kegs	Kent Dickey	Any
Pom1v0.61	1	java Exec	Arnauld Verhille	Any
prodosemu	2	prodos	Matthew Ghio	Text
xgs050	GS	xgs	Joshua Thompson	8 bit
yae	2	apple2	Doug Kwan	8/16/32
zaniWok	2	zaniWok	Matt Hostetter	2/16

« subdirectories. Eschewing disk image files, ProDOSemu remaps the UNIX file system so it appears as a disk named /UNIX on drive 1 in slot 7. So files are equally accessible in Linux and from within the emulator, but programs that run from protected disks cannot be used. ProDOS limits file names to a maximum of 15 upper-case ASCII characters. DOS 3 formats are not supported.

The ProDOS Apple //e emulator compiles with just one change on modern Linux systems. Line 21 of **'apple.h'** uses a **'bsd'** prefix for **'sgtty.h'** that is not valid for Linux. An adjacent line, commented out, suggests that this was a Berkeley UNIX compatibility tweak mistakenly left in the Linux release. You may remove the **'-DNOBEEP'** switch in the **Makefile OPT** line, because Linux now stomachs Randy Frank's BEEP code.

The Apple][e added keys and ROM but few other changes to the established Apple][formula.

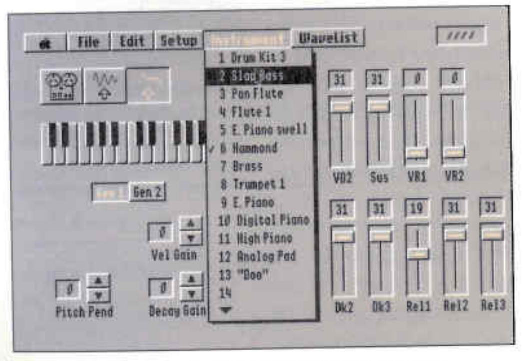
Emulator

The only documentation for Philip Stephens' *Emulator* is the source. It was written for Solaris 2 using OpenWindows, so the Makefile needs few tweaks. The path to the ROMs needs to be changed, and references to **'openwindows/'** removed and suffixed with **'/X11'** instead. **'make install'** uses **'mkfontdir'** to set up the font, and **'convertdisk'** is essential to make custom raw images from '.dsk' files. The little drive icons are a delight, changing when disks are 'inserted' by clicking and typing a name, and blinking with each access. *Emulator* reserves 12K for the 'main.rom' file, 48K for the main RAM and two areas of 4K and 12K, to mimic a 64K Apple][+ with a language card in slot 0. It needs 'diskIIcontroller.prom', the 256 byte bootstrap other emulators call 'slot6.rom'.

An Apple][+ with twin drives and monitor.

Java Applelet

Applelet is a neat but sluggish Java Apple][+ emulator. Its inbuilt speed reporter claimed almost half authentic speed under Sun's JDK 1.3 'appletloader', after 'policytool' let it access the sun.audio class. Sun's GUI is fussy about window managers; obscured gadgets make it unusable in *KWM* and *Blackbox*, but it works in *Enlightenment*. The emulator needs a font and 12K Apple][+ ROM image in the same directory, as 'charrom.bin' and 'apple2.bin'. 'fishies.bin' is the default 140K disk image on the website; you can change this in the DISK1 param tag in 'Applelet.html'. *Applelet* seems compatible but demands a very fast processor or Java runtimes. Our benchmarks rated raw 6502 emulation at about 1.2 MHz on a 500 MHz K6 host, but displays managed only 21 to 43 per cent of Apple //**e** performance, with spasmodic screen updates.



Synthlab **sounds out the power of the** ***Ensoniq*** **synthesiser in every GS, or emulator.**

Apple IIc: released back in April 1984, the model was terminated in November 1990.

I tried another five emulators with limited success. Koch's *Apple2* Simulator is portable but tardy. It has neat hardware virtualisation, but you must put **'slot3.c: noslot.c'** in the makefile unless you have the Videx extension ROM, and chop your ROM image into separate files with a utility I've written for the cover disk. It runs most programs, but segfaults unless run as root. The graphics cache makes updates lag, and when I tried the assembler HGR benchmark it gobbled all my memory, expiring after five minutes of increasingly desperate cache churning.

Ap2e emulates a 128K Apple with up to four ProDOS drives or two DOS 3.3 floppies and an 80 column card in slot 3. The text version compiled once I removed **'-Olimit 1500'** from the OPT line in the Silicon Graphics makefile. It needs a ProDOS image and an Apple2e ROM file in the same directory as the compiled code, named PRODOS and CDROM respectively.

//+ simulator dates back to 1989. It emulates the old model Apples but in text only, with a choice of integer or Applesoft BASIC. Version 1.0 compiles with just one warning — that the 'gets' function in debug.c 'is dangerous and should not be used', but gave a segfault when I tried to run it. *Yet Another Apple Emulator* is a dynamic translator for RISC systems. YAE can translate original 6502 code into SPARC or MIPS instructions on the fly, but Linux failed to compile **'iou.c'** in recent portable versions; older code compiled but rejected my system ROMs.

ZaniWok was an open-source project for the BSD-like system on Steve Jobs' NeXT workstations. It's a dynamic compiling emulator written entirely in C, but assumes a big-endian CPU and lacks support for low-res graphics. Programmers will find *ZaniWok* interesting but it could take a lot of work to get running on Linux.

Linux Apple emulator benchmarks (table 2)

System	A2	Java	xApple2		Emulator	Simulator		XGS	KEGS	A//
Pixels	N/A	x1	x1	x2	x2	x2	x1	x2	x2	x1
FOR-NEXT	3.5	52.6	3.5	4.4	6.9	10.4	7.5	8.0	5.1	60.0
TEXT SCROLL	6.5	95.9	3.5	7.3	20.8	16.1	6.0	3.7	5.0	41.5
HGR BASIC	N/A	178.9	3.8	5.0	12.3	31.8	14.3	13.5	6.5	56.0
COUNTING	4.7	109.0	5.3	6.4	11.5	14.2	12.5	6.8	8.2	134.0
HGR ASM	N/A	129.1	2.3	3.6	26.3	N/A	N/A	5.6	5.7	27.0

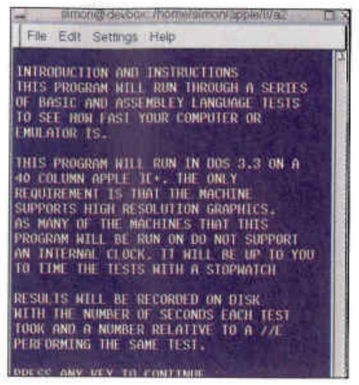
Notes: This table compares the speed of default configurations of nine emulations with a real Apple //e, in the last column. Results are seconds for each benchmark in spedtest.dsk, so smaller means faster. The test system was an AMD K6-2/500 with an 8-bit 1600x1280 pixel X display on a Matrox G400, running Debian Linux release 2.2. ***Emulator*** **is the one by Philip Stephens,** ***Simulator*** **is by Peter Koch.** ***Xapple2*** **and** ***Simulator*** **were tested with 1:1 and double pixel scaling in X and Y to show the overhead of a larger emulation screen.** ***A2*** **is text-only.** ***Simulator*** **crashes the HGR ASM test.**

Apple GS Emulators

At least two GS emulators work on Linux. The VirtualGS project aimed to produce X11 and PC Linux-specific versions of a portable Apple IIGS emulator, but I've only seen it mentioned in emulation and programmer FAQs. *KEGS* and *XGS* are more corporeal. I've tested both and put the sources on the cover disc. *XGS* stalled at version 0.50 five years ago, and needs tuning for Linux — you may find links to a beta version 0.53 but the files have gone. The host site says the project, and author, is not dead.

XGS Fixes

As the name suggests, *XGS* started out life on UNIX for X. It has since been ported to BeOS and MacOS. By default the release configures for an SVGA display and no sound, but if you try that you'll be disappointed because the SVGA 'vid-drv.c' file in the archive is half-baked, with missing identifiers, unterminated strings and it even has a function declared twice. Kludges made the source compile but not link, so I selected the X11 version, like this:



```
INTRODUCTION AND INSTRUCTIONS
THIS PROGRAM WILL RUN THROUGH A SERIES
OF BASIC AND ASSEMBLEY LANGUAGE TESTS
TO SEE HOW FAST YOUR COMPUTER OR
EMULATOR IS.

THIS PROGRAM WILL RUN IN DOS 3.3 ON A
40 COLUMN APPLE ][+. THE ONLY
REQUIREMENT IS THAT THE MACHINE
SUPPORTS HIGH RESOLUTION GRAPHICS.
AS MANY OF THE MACHINES THAT THIS
PROGRAM WILL BE RUN ON DO NOT SUPPORT
AN INTERNAL CLOCK. IT WILL BE UP TO YOU
TO TIME THE TESTS WITH A STOPWATCH

RESULTS WILL BE RECORDED ON DISK
WITH THE NUMBER OF SECONDS EACH TEST
TOOK AND A NUMBER RELATIVE TO A //E
PERFORMING THE SAME TEST.

PRESS ANY KEY TO CONTINUE
```
Spedtest was our benchmark for Apple emulators.

```
./configure —with-x —with-oss-sound —with-mit-shm
```

That gave similar problems, plus extra warnings after the usual system tests. The configuration makes links from generic sound and graphics files to platform-specific code in subdirectories of 'arch/', and once the first set of links have been made — for SVGAlib, in this case — fresh configurations are ignored.

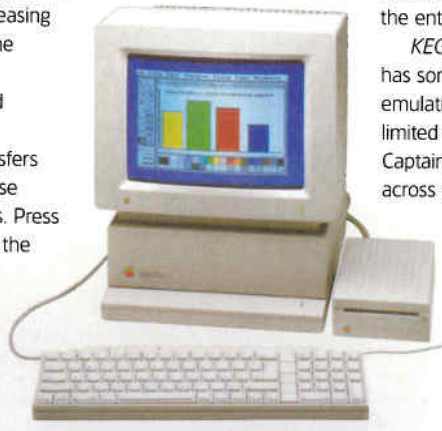
A buried line in the Makefile cures this, so type: **'make distclean'** before the **'./configure'** to get a working version for X11 and the standard Open Sound System audio drivers. The third switch requests MIT shared memory extensions for X. These are not essential but make updates much faster by stopping gratuitous graphics copying in Linux.

XGS needs an Apple GS ROM image called 'xgs.rom' in '/usr/local/share/xgs/' — either the original 128K version 01 or the 256K version 03 released in 1989. Enable writing to that directory, as *XGS* puts the battery-backed memory data there and can't boot from a disk image till it creates the corresponding xgs.ram file. Left and right ALT correspond to the GS command and option keys, and you can toggle mouse emulation of an Apple joystick or a single-button mouse — but not both — with F5 and F6. Control Home resets and Control End resets *XGS*.

KEGS

KEGS was born on a PA-RISC workstation but has been recoded in portable C and tested on Linux PPC as well as x86, so it should laugh in the face of endian problems. It works fastest on 8-bit X with MIT SHM, but does not insist on those. The **'-skip'** option boosts emulation speed on slow computers by reducing the redraw rate. *KEGS* can use '/dev/js0' and '/dev/dsp' for joystick input and stereo sound. Function keys toggle the sense and orientation of joystick signals, easing emulation of separate paddles. ALTs or the redundant Windows keys on recent PC keyboards can emulate Apple Option and Command keys.

F8 suppresses the X pointer and transfers control to the GS emulation one; otherwise you'll be confused by two mouse pointers. Press F8 again when you want to point outside the KEGS window. Your left mouse button emulates Apple's one. Your middle button switches speed between 1 MHz for 8-bit Apple][software, 2.8 MHz for 16 bit GS compatibility, and unlimited speed — up to 16 MHz, after sound and graphics overhead, on my machine. Tiny statistics about the CPU, video and drive emulation are updated every second beneath the main emulation screen.


Apple IIGS: there are at least two GS emulators which work on Linux.

The right mouse button interrupts the emulator, dropping you into a barely documented monitor in the launch shell. This can apparently trace, disassemble and set breakpoints, but you'll have to study the C source to learn how. The documentation is good but concedes a lack of depth on debugging and compatibility.

The key monitor commands are **'g'** to restart emulation after accidentally right mousing and **'q'** to quit after doing so deliberately. *KEGS* also drops to the monitor if it detects any access to undocumented ports or absent memory. This may be annoying as the GS had no memory management unit to trap sloppy code, so many programs make non-fatal wild accesses that invoke the monitor. The command-line option **'-ignbadac'** reduces but cannot eliminate such hiccups.

Disk emulation is excellent, with support for two 140K '.dsk' images in slot 6 or reading of nibblized protected disks, and 800K ProDOS images in s5d1 and s5d2. Slot 7 emulates the default boot device s7d1 and supports 32 logical drives, up to 4Gb in size, which can be ProDOS or HFS partitions shared with Linux, 2MG files as used by *XGS*, files in ProDOS block format, or original Apple CDs in Linux drives. A crude utility 'to_pro' formats ProDOS drive images of up to 32Mb and moves Linux files into them. The disk image swaps are handled by a 'kegs_conf' file, monitored as the emulation runs. Two columns list Apple and Linux device names, so:

```
s7d9 /dev/dsk/cd0:GO-Main
```

indicates the Golden Orchard compilation CD Main partition should appear in GS slot 7, drive 9. As you edit this file KEGS automatically removes and inserts disk images. When the GS reckons it has 'ejected' a disk, it comments out the entry by adding a hash at the start — neat!

KEGS 0.60 runs most programs well, but it still has some quirks. Serial modem and printer port emulation via UNIX sockets is said to be "very limited now, and only for adventurous souls." Captain Blood showed a line of graphic glitches across the screen. The 3200 colour mode that changes the palette every line is not implemented. *Ensoniq* sound emulation is buggy when running some applications on Linux, dropping into the monitor so much as to make the *Jam* music package almost unusable. Overall *KEGS* is fun, and the best GS emulation you're likely to find on Linux, though *XGS* would be impressive if *KEGS* did not exist. LXF

Links

A2 homepage:	http://www.skrenta.com/a2
Applelet source:	http://www.fasterlight.com/hugg/projects
Background files:	http://www.archaic-apples.com
Docs and utilities:	http://www.umich.edu/~archive/apple2
Emulator archives:	ftp://ftp.uni-kl.de/pub/apple2
Java Applelet:	http://www.geocities.com/ResearchTriangle/Lab/8089
KEGS:	http://www.crosswinds.net/~kentd
Many Apple files:	ftp://ftp.apple.asimov.net
POM Apple 1:	http://www.chez.com/apple1
XApple2:	ftp://ftp.ocis.net/pub/users/ldeutsch
XGS:	http://www.inwards.com/xgs

I'm ready for dessert. Have you got any pudding for me to sleep in?
whines as he wipes the bar. His teeth are a webwork of East European steel and brown decay.
XGS running the point-and-click adventure ***Neuromancer*****.**

Click the forward Arrow to go forward, click the Back Arrow to go back, or click MENU for the main menu screen.
MENU
The Apple GS master disks include an excellent tutorial.

The MicroProfessor 2, a rather neat Apple II clone with a detached keyboard.

Spot anything fishy about this Java Applelet?