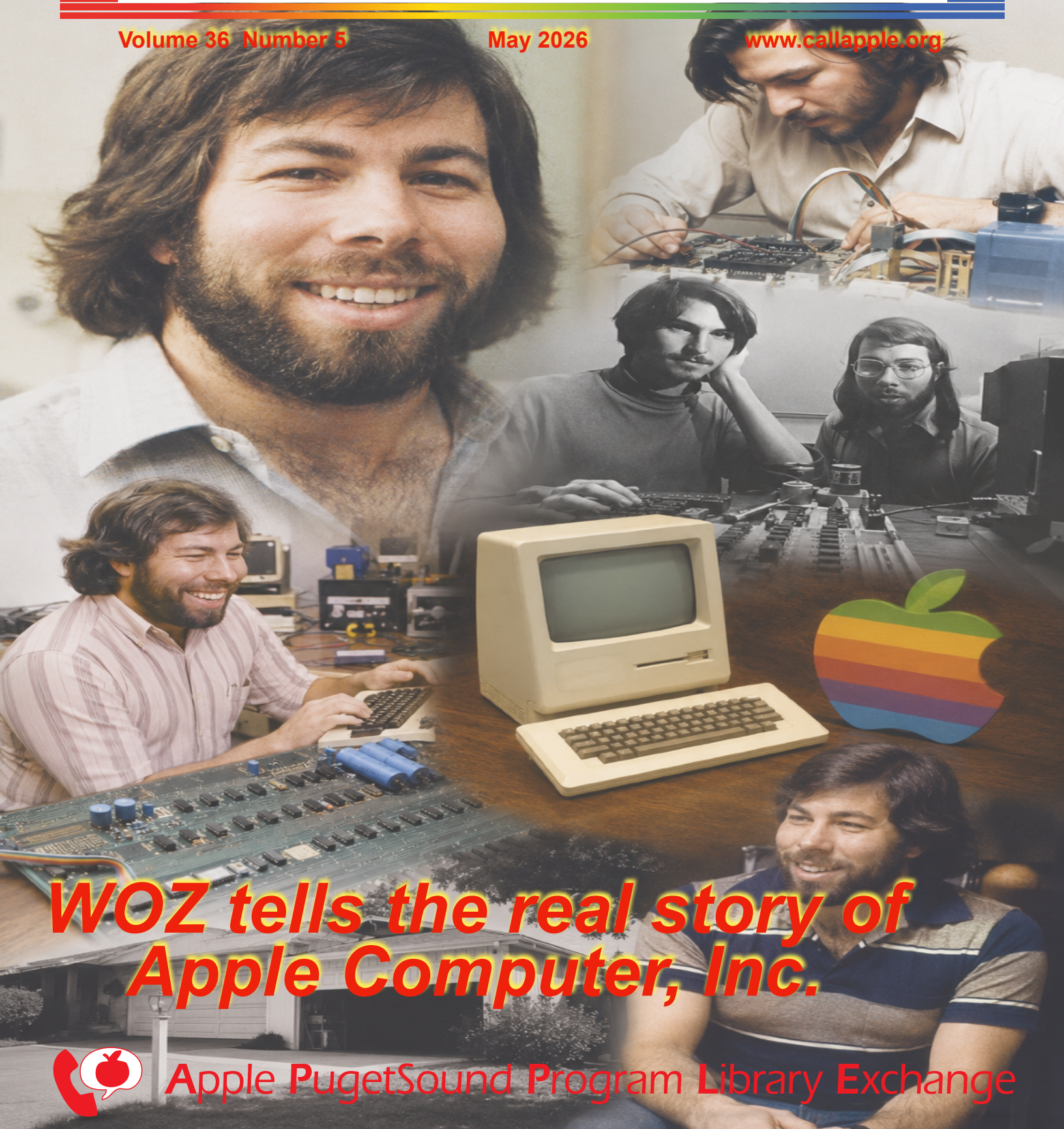
Call-A.P.P.L.E.™

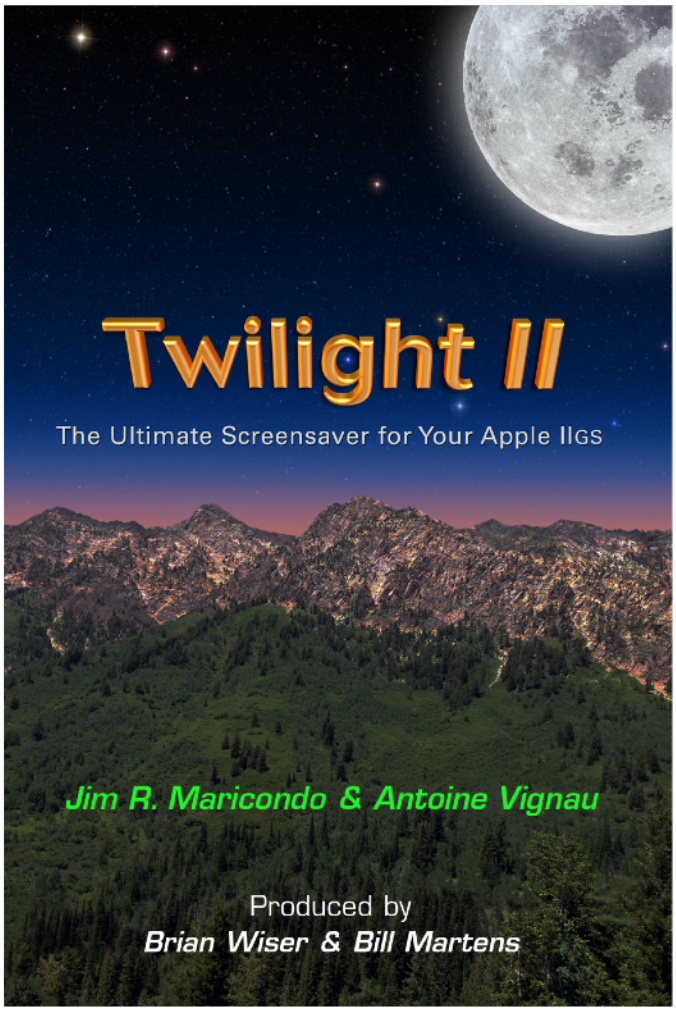
World's Largest Apple User Group Magazine — *Since 1978*

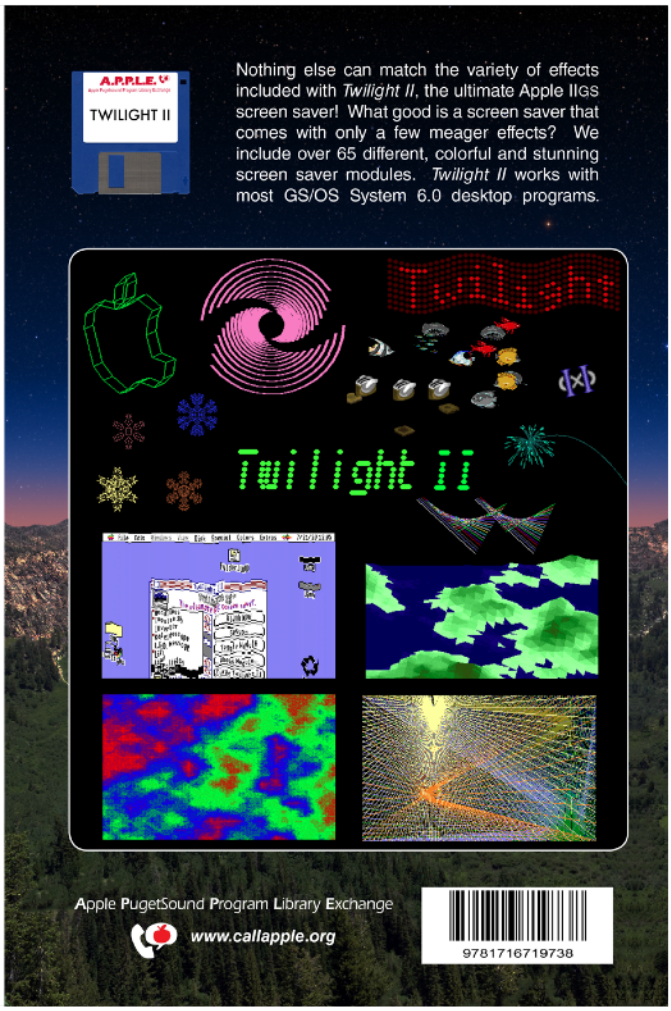
Volume 36 Number 5 May 2026 www.callapple.org



WOZ tells the real story of Apple Computer, Inc.

Apple PugetSound Program Library Exchange





Are You Upset?

Angered by the way the Apple IIGS has been treated lately? Sick of your friends bragging that they have the better computer? Does seeing *After Dark* on the IBM or Mac make you wonder, "Why can't my IIGS do that?" Do you like dazzling effects that will protect your valuable monitor from becoming useless when the same image is left onscreen for so long that it burned into the glass?

Are you tired of dull screen savers that slow you down and interfere with your work, from companies that don't want to upgrade their products? If so, then ***Twilight II version 2.0*** is the answer to your prayers!

Get the manual and app at:

www.callapple.org/books

Call-A.P.P.L.E.™

World's Largest Apple User Group Magazine — *Since 1978*

Volume 36 Number 5 **May 2026** **www.callapple.org**

Call-A.P.P.L.E. Magazine ISSN

8755-4909 1705-4109

Membership in A.P.P.L.E.

Membership fees are $27.95 per year, and include a one year subscription to A.P.P.L.E., user group discounts, and archival PDF materials including *Call-A.P.P.L.E.* magazine, *Mac-A.P.P.L.E. magazine*, legacy software, and other A.P.P.L.E.-produced websites. Annual membership dues are always due the month you first pay.

Subscriptions

Subscriptions to *Call-A.P.P.L.E.* magazine can be attained by joining the Apple Pugetsound Program Library Exchange (A.P.P.L.E.) user group, founded in 1978. The magazine is one of the premium benefits. For more information, please visit the membership page: www.callapple.org/members

Advertising

If you wish to advertise in *Call-A.P.P.L.E.* magazine, email sales@callapple.org for our very competitive rates. We offer in-magazine, on-site, and full sponsorship for prospective advertisers.

Submissions

Call-A.P.P.L.E. is always looking for new and interesting ideas and articles related to Apple, Linux/Unix, technology of general interest, technical or editorial, modern or retro. All submissions must be original works of the person submitting and free of distribution restrictions. Please email your submission proposal to: editor@callapple.org. By submitting materials to us, you agree to give A.P.P.L.E. the royalty-free right to publish and reuse your submission with your name in any form in any media. We reserve the right to edit, change, or not use anything submitted.

Copyright

Golden Grail Apple II CD-ROM

CD-ROM is definitely a hot part of today's home computer industry, and the Apple IIGS market is no exception! With the advent of Sequential Systems' DiscQuest, extensive libraries of informational CDs usable on an Apple IIGS are all the rage, including an encyclopedia. But there has been a shortage of compilation CDs (CDs that contain a vast assortment of freeware and shareware programs, information, and data), until now!

Jim Maricondo and Bill Martens have gone through and re-worked the original **Golden Orchard CD**, presenting you with a bevy of useful files you that you will find on only on **Golden Grail**. No longer wait for slow downloads, or mail order shareware! **Golden Grail** puts *over 600 megabytes* of Apple II files at your fingertips. It's Emulator-ready — every file is ready to mount and run on modern Apple II emulators like *Ample*, *GSSquared*, *GSplus*, and *KEGS*. If you don't have a CD-ROM drive, perhaps now is the time to consider purchasing one. The cost of drives has plummeted, and the library of Apple II compatible CDs only continues to grow! Now is the time to join the CD-ROM revolution!

Golden Grail Content Summary

Note: Shareware fees must still be paid to some of the programs' authors if you end up using the programs frequently.

- **Applications** 81mb
  - Demos 10mb
  - Games 27mb
  - Graphics & Sound Demos 7mb
  - Graphics Utilities 5mb
  - Sound & Music Programs 10mb
  - System Utilities 4mb
  - Telecommunications 5mb
- **Disk Images** 145mb
  - FTA Software 22mb
- **Graphics** 51mb
  - 3200 Color Pictures 8mb
  - Animations 10mb
  - APF Pictures 5mb
  - GIF Pictures 12mb
- **Music** 64mb
  - MIDI Songs 5mb
  - MODs 22mb
  - SoundSmith Songs 16mb
  - SynthLab Songs 12mb
- **Sounds** 13mb
- **Apple Software** 28mb
  - System Software
  - HyperCard IIGS
- **AppleWorks** 10mb
- **BASIC Programs** 1mb
- **Stacks** 28mb
  - HyperCard IIGS 6mb
  - HyperStudio 22mb
- **Bitmap Fonts** 2mb
- **TrueType Fonts** 28mb
- **Deprotects & Cheats** 6mb
- **Icons** 1mb
- **CDAs** 2mb
- **NDAs** 2mb
- **Inits** 1mb
- **Finder Extensions** 1mb
- **Patches & Updates** 1mb
- **Text Files** 25mb
- **Programming** 124mb
  - From Apple 31mb
  - Data Compression 2mb
  - Assembly Source 8mb
  - C Source 10mb
  - Utilities 18mb

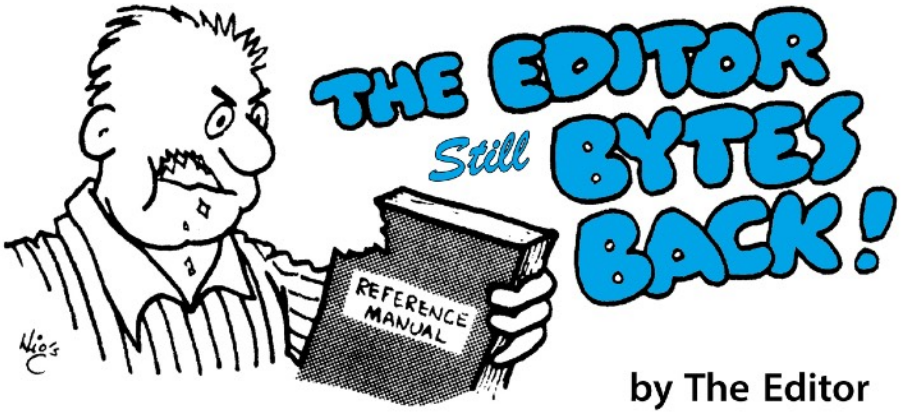
This is just a partial listing! ***Golden Grail*** *is the only CD available that contains over 600 megabytes of files useful to Apple II owners! This CD will pay for itself many times over! And now all files are 100% Apple II Emulator Ready!*

https://digisoft.callapple.org

II Infinitum

The Editor *Still* Bytes Back!

by The Editor



Over the years we have featured a number of writers from all over the world. One that we have featured is Randall Kindig from Floppy Days fame. He has been someone who has interviewed some of the biggest names in computing history and this month he scored a winner with the Apple fans out there.

With the Byte Shop celebrating 50 years, Randall sits down with the one and only, Steve "WOZ" Wozniak and tells us what the beginning of Apple Computer, Inc. was really like. Be sure and check out the interview.

We also bring a number of new tools and utilities to the purview of our members this month, including the new Global Computing Events calendar, which you will find featured prominently each month on our page of the month's computing events. This month's events stretch over two pages and covers the gamut of computing related events around the world.

While we have been busy this month with a number of tools and utilities, the games have not taken a back seat with ***Ultima III*** nearly at release quality along with two ***Ultimore*** Games and the utilities which allow users to cheat a bit in the games.

A late breaking story that you will only read about here is the return of ***Structris***. Originally written by Martin Haye in 2012 and released as an iOS app in 2015 by A.P.P.L.E., the game has been reworked and will be back in the Apple App Store later this month.

Also this month, the registration for the 2026 rendition of the annual Apple II fans pilgrimage to that holy land called Kansas City. But alas, this year's event will not be in KC, instead being held in Springfield, Indiana. If you plan on attending, check out their website.

Also, this month, you see the 8th part of the ***SAB Pilot*** series of program rewrites by Forrest Lowe. The series continues and will be part of our on going series for a few more issues until we issue the fully qualified re-written Apple II version of ***SAB Pilot*** for the Apple II complete with editor and interpreter.

Remember to check out our A.P.P.L.E. store for the latest goods and software. This end of month will finally bring the release of a number of new games, tools, and of course more general information related to the retro Apple computing hobby.

For those of you who are regular readers of our monthly magazine, we also invite you to write articles if you have a program or a project you would like to share.

For those who have renewed their memberships this month, we thank you for your continued support. To our new members, Welcome and we hope you enjoy your membership in the World's Largest User Group.

For those of you on Social media, you can follow us on any one of the following :

- Facebook: https://www.facebook.com/APPLEug
- Twitter: https://x.com/callapple
- Mastodon: https://mastodon.social/@callapple
- Bluesky: @call-apple.bsky.social

Editors note. We will be relocating our editorial office once again this next week. Next week we will be coming to you from the the heat and humidity of Tokorozawa Japan once again.

No services that are online are affected by this relocation but end user support may be down for a few hours thanks to this move.

Your Apple IIc still has more to do!

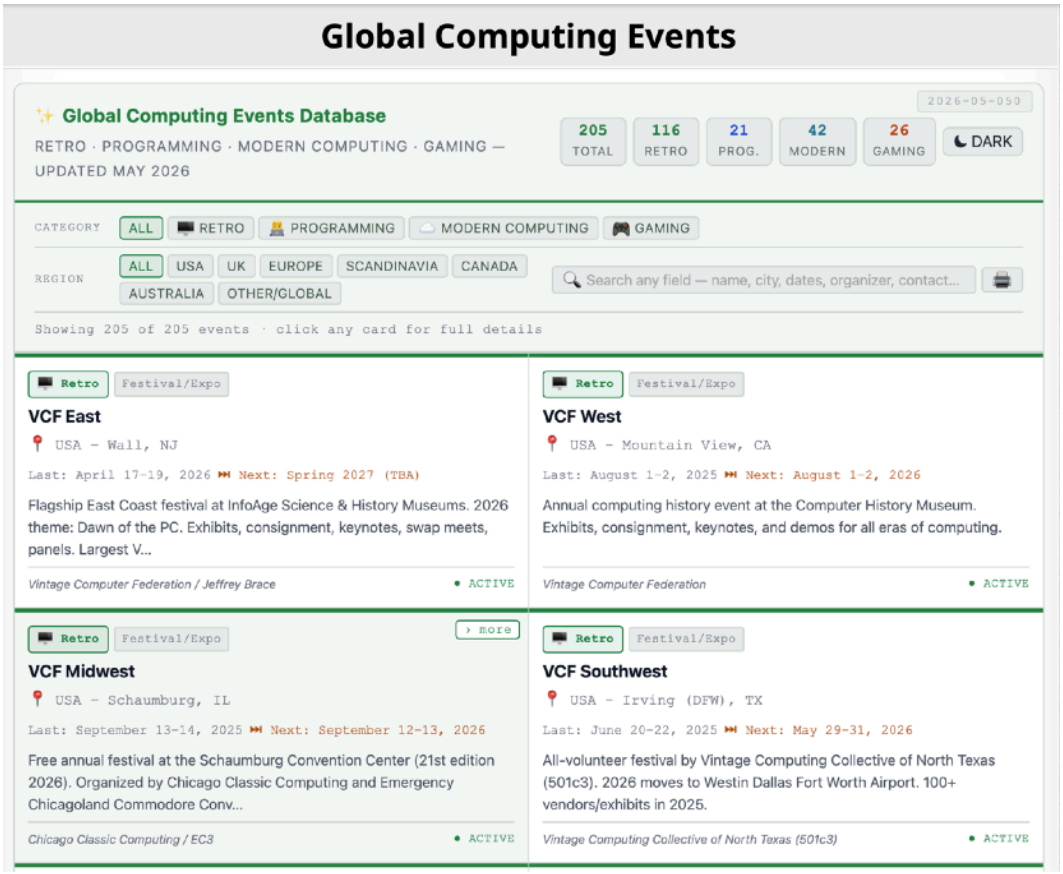
8-BIT SHACK

http://8bitshack.net

8-Bit Shack creates modern retro software experiences for the Apple][– including games, demos, enhancements, and utilities. It's everything you loved about 8-bit computing, plus brand-new reasons to keep coming back.

New Nifty Tools From the Minds of A.P.P.L.E.

by **A.P.P.L.E. Staff**



Over the past 6 months, the staff at A.P.P.L.E. have been hard at work, developing a major project. But as with any major programming project, side projects raise their head out of small bits and pieces of the larger project.

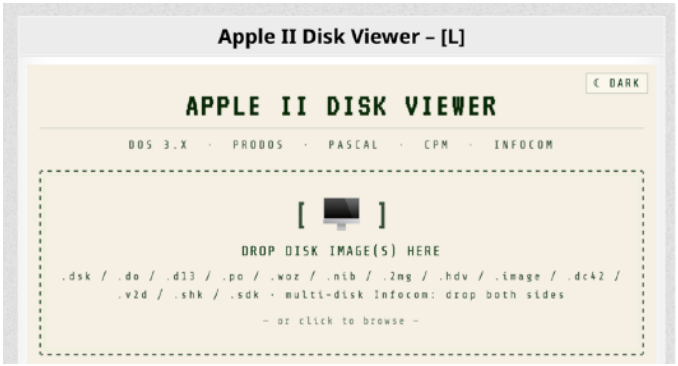
Out of our project which we have nicknamed Thunder-dome after the great Tina Turner led Mad Max 2 movie, have come a series of side projects which are just now making their way into the main website.

Global Events Calendar. This project came about because the events each month kept scrolling off the page and becoming more of a hodgepodge of scattered events across sever of our web postings and magazine pages.

That is no longer the case. now we have all of the retro and modern computing and gaming related events in one location. The page not only lists all of the events but also allows users to search by event type, location and even their own key words. What is really cool, is that is you look at the tabs that are presented, it shows the event information, when the event was last held and when the currently scheduled next dates for the event are. It also shows the status of the event for whether it is active, cancelled, in-active, or temporarily halted which happens occasionally.

The Global Computing Events Calendar is available from A.P.P.L.E. at https://www.callapple.org/events-2.

Apple II Disk Viewer



This wonderful little tool, allows users to drag any Apple II series disks to the page and the contents of the disk will be revealed by the tool. While there have been a number of tools before, this is

the first one that contains all of the capabilities of the previous viewer and gives the user a few extra goodies including the ability to finally view Binary source files such as those from Big Mac or Merlin as text.

We have also incorporated any other features that we could find in the Apple II retro computing realm. Obviously, these features now being all in one easy to use tool makes them more useful than ever, One such feature is our Disk Viewer's capability of retrieving a z5 file from a double disk upload. Yes, drag Side A and Side B of a game and you wind up with the complete story file which can then be downloaded and used in our web-based Krebf Z-Machine Player which we will talk about a bit later.

We did not go further with Block Viewers or that sort of item, instead focusing on the files on the disks.

This online tool is available to all of our members at https://www.callapple.org/apple-ii-disk-viewer/.

Z-Gen Apple II Z-Machine Disk Generator

This was one of our first projects to come about late last year as part of the Inform 6 Game library that we are building of in-house games. That library came about as a result of the need to build a demo game for another project we are working on. We needed a sample game which we talked about before, Castle Alethra.

Although this game gas not been released, while the other 5 games we have produced have been released, the main focus was to create a generator which would finally allow Apple II users and game producers to easily produce Apple II disks that actually work on the real machines as well as emulators. Z-Gen definitely fills this mission nicely.

This tool is avilable to our members at: https://www.callapple.org/zgen.

KREBF Z- Machine Player

This Z-File player was originally written as a platform specific program by Fredrik Ramsberg, the current author of the PunyInform Library, a library which allows users to create smaller z-machine files for use on a variety of 8-bit platforms. But since KREBF was not available online, we went into dev mode again with the Javascript, and the result is our version of the KREBF Z-Machine Player.

We have added all of our latest games to the player as well as Fredrik's Mars2024 game from one of the recent PunyJams that occur occasionally. The player even allows users to upload any other z-machine file including all versions z3 through z8 excluding z6.

The Player is available at https://www.callapple.org/player.

Ultima III for macOS 26

Tools are not the only thing we have been working on this past year or so. Two years ago, Leon McNeill released the source code for his version of the Macintosh version of Origin Software / Richard Garriott's original game Ultima III. While the game worked great on the early versions of Mac OS X as well as on Mac OS 9.2.2, the original was written in Carbon which is no longer used. Even Leon thought it was pointless in updating it and even said that it might be impossible.

After nearly 8 months of coding, the game is deep into very stable and playable beta's with the latest version of the game, V.4.0.0 Build 0183 being released as we go to press with this issue of the magazine.

We have had a few issues with it along the way, but thanks to a lot of effort and time spent debugging / playing the game, we are expecting a finished product just in time for KansasFest 2026. In the interim though, you can join A.P.P.L.E. and get the beta from our website at: https://www.callapple.org/u3

Ultima III Cheater

This nifty little tool goes hand in hand with the game Ultima III and allows users to actually do just what the title suggests, cheat at Ultima III.

Also, initially written in Carbon, this conversion took a lot less time and effort than the game did and we even managed to update the tool and add some bells and whistles to the original, improving greatly the usability of the tool.

Ultima III Cheater.app

Ultima III.app

The beta of the Ultima IIII Cheater is available from our website at: https://www.callapple.org/u3

The Future: We have a slew of other tools and games we will be adding to the pie here. Thanks to Brian Wiser, Forrest Lowe, Fredrik Ramsberg, Bill Tomlinson, and all of the other community members and A.P.P.L.E. User Group Members who have been helping us out with beta testing and additional support. Also, thanks to our A.P.P.L.E. members especially since, without you and your support, we could not bring these tools to life.

Apple II Series Emulators: The Complete Rundown Updated

by A.P.P.L.E. Staff

Over the past several years, we have provided this list to other parties as a comprehensive list of the emulators for the Apple II series computers without favor or exclusion. This year, we decided to make this available to everyone and let users make their own judgements about the quality of the products listed. After a glaring omission in our previous month's issue, we decided to reprint the article this month.

There are a few dead projects not listed since they are only available in the Internet Archive and not necessarily in fully qualified state of completion.

If you see an emulator listed here which has been updated, changed, or is no longer in development, please let us know. The list is also available to readers under the A.P.P.L.E. website's EXTRAS menu as the first item on the menu.

Emulator – URL – Author – Notes

- **8BitWorkshop** – http://8bitworkshop.com/ – Steven Hugg – A completely online development multi-platform environment written by the author of the original Java Applet II+ Emulator
- **Accurapple** - https://gitlab.com/wiz21/accurapple - Stéphane Champailler - A new Apple II emulator programmed in RUST (requires RUST 1.82 or newer) which is an attempt to replicate an Apple 2e Revision B wired to a PAL CRT monitor.
- **ActiveGS** – http://activegs.freetoolsassociation.com – Olivier Goguel – Used by VirtualApple.Org but only works on Waterfox Classic Browser now.
- **ActiveGS iOS** – https://github.com/ogoguel/activegs-ios – Olivier Goguel – An Apple IIgs Emulator for iOS based on KEGS, not in App Store. Uses the VirtualApple.Org Database
- **Agat** – http://agatemulator.sourceforge.net – Odintsov Oleg – Russian Emulator for the Russian Apple II Clone, the AGAT computer.
- **Apple2e** – https://github.com/inindev/apple2e – John Clark – In progress Apple IIe emulator with much work still needed.
- **Apple2go** – http://a2go.applearchives.com – Marc Ressl | Nick Westgate – Java based Apple II Emulator originally. Currently used on the VirtualApple.org website Apple II Section
- **Apple2portable** – http://psp-news.dcemu.co.uk/apple2portable.shtml – Team Xos – Apple II Emulator for the PSP
- **Apple2-Go** – https://github.com/freewilll/apple2-go – Klaus Dormann – Emulator written in GO based on the original Apple2go by Marc Ressl.
- **Apple2000** – https://github.com/kkralian/Apple2000 – Kevin Kralian – Apple II Emulator for the Amiga
- **Aiie** – https://github.com/JorjBauer/aiie – Jory Bauer – Emulator written for the Teensy
- **Updated Ample** – https://github.com/ksherlock/ample – Kelvin Sherlock – MAME Frontend for MAC OS featuring Apple and Apple clones
- **Apple In PC** – https://github.com/sosaria7/appleinpc – Keonwoo Kim – Korean emulator Apple In PC is at version 0.1.46.1
- **AppleIIemu** – https://github.com/allender/apple2emu – Mark Allender – Cross Platform Apple II Emulator. Includes full Integrated 6502 debugger
- **Apple IIix** – https://github.com/mauiaaron/apple2 – Aaron Culliney – An Apple IIe Emulator that runs on multiple android platforms written in C99
- **Apple IIjs** – https://github.com/gmegidish/apple2js – Gil Megidish – Original Binary emulator in Javascript.
- **AppleIIts** – https://github.com/chris-torrence/apple2ts – Chris Torrence – Emulator created from scratch in primarily TypeScript and highlighted in Chris's video cast on YouTube.
- **AppleIIjs / Apple IIjse** – https://www.scullinsteel.com – Will Scullin – Full featured Apple II Emulator based in Javascript. This is one of the best online emulators available today and is regularly updated.
- **ApplePi** – https://github.com/FZBunny/applepi – J.B. Ward – Apple II Emulatorfor the Raspberry Pi

- **ApplePy** – https://github.com/jtauber/applepy – James Tauber – An Apple II Emulator written in python. Last updated 8 years ago.
- **Appler** – https://github.com/zajo/appler – Alexander Patalenski | Emil Dotchevski – Apple II Emulator for the 8080. Runs in DOSbox
- **Applerm** – https://github.com/toyoshim/Applerm-II – Takashi Toyoshima – Appler for the ARM Cortex M0
- **AppleToo** – https://github.com/nicholasbs/appletoo – Nicholas Bergson-Shilcock **–** An Apple II emulator written in Javascript. Last update 5 years ago
- **Updated AppleWin** – https://github.com/AppleWin/AppleWin – Tom Charlesworth – Apple II Emulator for Windows. This is probably the most developed Apple II Emulator and was created originally in the 1990's. Updated in 2022
- **Arduino-Apple II** – https://github.com/dpeckett/arduino-appleii – Damian Beckett – Written for the Arduino – No Releases
- **Bobbin** – https://github.com/micahcowan/bobbin – Micah Cowan – A terminal-driven Apple II-series emulator aimed at efficient development for 8-bit Apple computers.
- **Calormen** – https://www.calormen.com/jsbasic/ – Joshua Bell – Applesoft interpreter in Javascript running in a browser
- **Candy Apple** – http://www.zerohoh.com/candyapple – Garnet Ulrich – The original Android based Apple II Emulator. No Longer in the Google Play store.
- **Catakig** – http://catakig.sourceforge.net – Colin Klipsch – One of the early Apple II Emulators
- **Clemens IIgs** – https://github.com/samkusin/clemens_iigs – Samir Since – A new IIgs Emulator with the backend written in C and the frontend written in C++
- **Updated ClockSignal** – https://github.com/TomHarte/CLK -Tom Harte – Multiple platforms and runs Apple II and Mac.
- **Crapple** - https://github.com/cbmeeks/crapple - C.B. Meeks - An Apple II emulator for Linux.
- **Cyaniide Applesoft** – https://paleotronic.com/applesoft/ – Melody Ayres-Griffiths | April Ayres-Griffiths – Apple II Emulator by Melody and April
- **Cyaniide Merlin** – https://paleotronic.com/merlin/ – Melody Ayres-Griffiths | April Ayres-Griffiths – Apple II Emulator featuring the Merlin Assembler by Melody and April
- **Epple II** – https://github.com/cmosher01/Epple-II – Christopher Mosher – Apple II Emulator for the ESP8266
- **Epple** – https://github.com/hrvach/espple – Hrvoje Čavrak – Apple I Emulator for Windows, Ubuntu Linux and macOS.
- **Florence** – http://www.kashum.com/florence.pl?cmd=help – Richard Bennett – Apple IIgs Emulator written in Javascript. Never completed
- **GoApple** – https://github.com/zellyn/goapple2 – Zellyn Hunter – Emulator written in Go
- **GSplus** – https://github.com/digarok/gsplus – Dagen Brock – Main page at https://apple2.gs/plus/ . Currently one of the best Apple IIgs Emulators next to KEGS.
- **GSport** – https://david-schmidt.github.io/gsport/ – David Schmidt – A very good Apple IIgs Emulator
- **Updated GSSquared -** https://github.com/jawaidbazyar2/gssquared **-** Jawaid Bazyar - GSSquared is a complete emulator for the Apple II series of computers. It is written in C++ and runs on Windows, Linux, and macOS.
- **izappleii** – https://github.com/ivanizag/izapple2 – Iván Izaguirre – Emulator written completely in Go
- **JACE** – https://github.com/badvision/jace – Michael Pohoreski | Brendon Robert – Emulator written entirely in Java. Last update was more than 2 years ago.
- **KEGS** – http://kegs.sourceforge.net – Kent Dickey – Kents Emulated GS – The grand daddy of the Apple IIgs Emulators and the one that most of the others are based on.
- **KEGS Android** – https://github.com/jamessanford/kegs – James Sanford – Android version of Kegs in the Google Play Store
- **LinApple** – https://github.com/linappleii/linapple – Various Authors – Apple II Emulator for Linux

- **Updated MAME** – https://www.mamedev.org – Various Authors – The MAME project which emulates many platforms including the Apple II Series
- **Updated Mariani** - https://github.com/sh95014/AppleWin - Mariani is an emulator of the Apple][and //e computers for macOS.
- **MicroM8** – https://paleotronic.com/software/microm8/ – Melody Ayres-Griffiths | April Ayres-Griffiths – The MicroM8 project is a three dimensional Apple II emulator written by Paleotronic.
- **MII Apple IIe Emulator for Linux** - https://github.com/buserror/mii_emu - Michel Pollet - A linux based clock accurate Apple IIe emulator for the Linux Operating System.
- **Oasis** – http://www.kd77.net/apple2/ – Unknown – Apple II Oasis has not been worked on in many years, but the website and resources are still online.
- **Octalyzer** – https://web.archive.org/web/20170530191159/http://www.octalyzer.com/ – Melody Ayres-Griffiths | April Ayres-Griffiths – Melody & April's original Apple II emulator. No longer developed, website offline.
- **OpenEmulator** – http://openemulator.github.io – Various Authors – An emulator which runs the entire Apple II series computers
- **PalmApple** – http://palmapple.sourceforge.net – Hilary Cheng – Apple II emulator for the Palm series of PDAs.
- **PomDS** – https://wowroms.com/P/emulators-software/33/PomDS.html – Alek Maul – The last know location to download the Apple II Emulator for the Nintendo DS.
- **Steve II** – https://github.com/trudnai/Steve2 – Tamas Rudnai – Introduced at KansasFest a few years ago
- **Sweet16** – http://www.sheppyware.net/software-mac/sweet16/index.html – Eric Shepard – An emulator developed originally on the BEOS platform. Runs on MAC OS
- **Virtu** – https://github.com/digital-jellyfish/Virtu – Nick Westgate | Sean Fausett – Emulator for Silverlight on Windows
- **Virtual II** – https://www.virtualii.com – Gerard Putter – A full featured Apple II Emulator and the only one known to have specific Apple IIc emulation other than MAME. The best MAC OS based emulator. Updated in November 2024
- **XGS** – https://github.com/jmthompson/xgs – Joshua M. Thompson – An Apple IIgs Emulator for Linux

RetroMacCast Episode #736: Scott's Frog Design MacBook Concept Creation

by A.P.P.L.E. Staff

Welcome to the 736th Episode of the RetroMacCast. In this episode of the,James and John discuss their latest eBay finds including the following items:

retro Apple mouse pad, unreleased Mac 128k developer manuals, and Macintosh II. Scott joins the podcast to show the Hartmut Esslinger Frog Design MacBook concept he brought to life. News includes an Apple Lisa created with an FPGA, turning a $20 AliExpress clock into a functioning Mac, and the LaCie FM Radio Tuner.

If you would like to subscribe to the RetroMacCast or to listen to their latest episodes, be sure the check out their Facebook page or follow them on Twitter. You can also watch the RetroMacCast full episodes on YouTube.

For more information about this episode or to catch up on past episodes, be future and check out their website at: RetroMacCast.

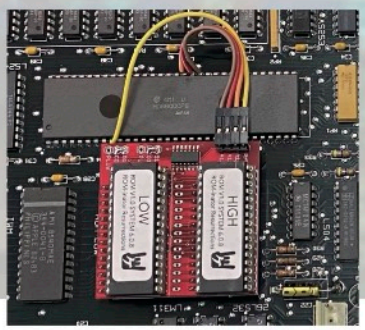
Kero's Mac Mods Online Store

To all Call-A.P.P.L.E. readers — Three essential tools to maximize the capabilities of custom ROMs and optimize your Macintosh!

ROM-INATOR RESURRECTIONS

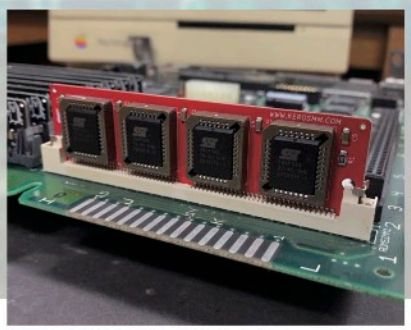
*Licensed by BMOW

ROM card ideal for Mac Plus, 512K, 128K. Kit product that can boot from a ROM disk.



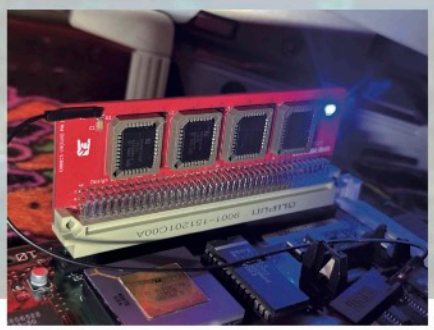
SMC ROM SIMM 2MB CARD

Removable ROM chips that can be programmed with a general-purpose programmer designed specifically for 68030 machines.



SMC PDS ROM CARD

A revolutionary ROM card that can be installed in PDS slot without using the SE/30's poorly contacting SIMM socket.



SPIISD V2

Limited–production Complete Set
with Branchus Creation's Shell Case

Housed in the custom-designed SPIISD V2 shell case created by Bruce Rayne, founder of Branchus Creations.
This set is delivered as a fully assembled, ready-to-run SmartPort solution and includes:

- SPIISD V2 SmartPort emulator board
- Custom-color 0.96" OLED display (yellow & blue)
- IDC20 to DB19 SmartPort adapter
- 20-pin ribbon cable (20 cm)
- Preloaded microSD card

All components are factory-installed inside the SPIISD V2 enclosure, allowing immediate operation straight out of the box with no additional assembly required.
The complete system is priced at **USD 64**, offering exceptional value for a fully integrated SmartPort storage solution!

Floppy Days #162: Interview with Steve Wozniak
Apple Computer, Inc. Co-Founder

by Randy Kindig / A.P.P.L.E. Staff

Patreon: https://www.patreon.com/FloppyDays

Sponsors:

- 8-Bit Classics
- Arcade Shopper
- FutureVision Research

Hello, and welcome to episode 162 of the Floppy Days Podcast, for December 2025. I am Randy Kindig, your host for this podcast.

Floppy Days: I am here with Steve Wozniak. While for most of us Retro Computing fans, he needs no introduction, this is more for the general public. Steve Wozniak, better known as Woz, is one of the co-founders of Apple Computer, Incorporated, known today simply as Apple, Inc.

WOZ : hi there

FLOPPY DAYS : Steve how are you sir?

WOZ : real good yeah yeah yeah

FLOPPY DAYS : I don't mean to be a fan boy or anything but talking to you was on my bucket list so you're making my day definitely

WOZ : oh good. yeah usually just takes asking

FLOPPY DAYS : oh well, it's hard i know you're a busy guy and and I appreciate that. But I just have a couple of questions for you. I think you've seen the questions already. Did you have any questions before we got started here?

WOZ : I didn't, I didn't really look deeply about them or think about them, but I'll just go with whatever gets asked.

FLOPPY DAYS : Okay. Sounds good. I appreciate that.

WOZ : We can go as long as you wish to.

FLOPPY DAYS : Oh, nice. Okay. the Byte Shop 50th anniversary event is coming up this weekend. Are you going to be able to make it?

WOZ : Yes, I am. I plan to make the dinner tomorrow and the event Sunday.

FLOPPY DAYS : Oh, excellent. I wish I could be there.

WOZ : It was a little uncertain. We were on a long road trip visiting relatives and all that. There was a reason we had to come back early, so we're here.

FLOPPY DAYS : Okay. The event, as you know, is a celebration of the personal computer and its impact on many different industries at the time. The Byte Shop stores were meant to be a place of introduction to the affordable PC for everyone. And I just had a couple of questions for you around that. Specifically, what memories or stories do you have about the Byte Shop?

WOZ : Ah, well, at the time, I was at the Homebrew Computer Club, and there were, I was reading some, you know, I don't know, electronic magazines and hobbyist magazines. And there was very little in the way of computer stores. There were a few around the country trying to start up. They were just nothings, mom and pop, you know, one person trying to figure out a way to do something. And the Byte Shop was here, was local here in Mountain View. and I didn't know Paul Terrell, but he was at the Homebrew Computer Club where he saw me, you know, demonstrating my computer. And this is before Jobs knew it existed or was even around, demonstrating it and showing it to people. I was too shy to ever raise my hand and talk. But the way I was, I was such a great designer, I would design clever little things, impress people, and then I could talk. That was my method of socialization. Always included a little bit of fun and pranks and things.

And I guess Paul saw me there, although I didn't know it and I didn't know him. But I gathered a group of friends, some in high school, that just believed in what I was doing. It was so different than the other computers. It was really the very first time that it said, a computer, when you buy it, has a keyboard and a video display.

And the word personal computer has been interpreted to mean a lot of different things. You know, I knew the, like a lot of other people knew that you make a processor and a bus and you connect some switches and lights and you plug in bits of data, ones and zeros in the memory a bit at a time, and then that processor can run them. That's sort of been the history of computers.

And of course, people can design, you know, in those days it was rather expensive, but you could design boards that send data out on big serial lines and talk to a teletype for input and output. But that was by far more expensive than any processor would be to even build on your own. And the product that people were trying to build had a pre-built processor called a microprocessor. And it ran out on a standard, what's called a bus, the address and data pins. And you can plug in boards and somehow adjust those boards to look to certain addresses and do things. And you could build up a computer.

But the computers that they were selling were really just low-cost affordable. So it was a microprocessor, some switches, some lights, some parts. Take dozens of hours of your own time, maybe, and hook it all together. And now toggle some switches and get some bits in the memory. You could work for, you know, half an hour and get a little program in that made tones.

But I had built that exact computer of my own design five years before. Built it when you had to build the processor out of TTL chips. And I designed a very simple processor. And I demonstrated it to a local newspaper and everything. And now I was a little past that in time because we built little, we built the first handheld scientific calculators at Hewlett Packard where I was working. I was designing them. They had little one-bit processors, one bit at a time of data flows through, and you could add and subtract and this.

And basically you started it up and it had some code. And the reason we used RPM, reverse Polish notation, we couldn't afford enough chips to have the code to parse an equation. We would say what the computer wants. Here's a five, store it. Here's a seven, store it. Here's plus, add them. That's what we would do. But it was kind of

human. You could see it in display. That wasn't advanced. It was only because we didn't have the memory to do what you're taught in school. Five plus seven equals.And you see the result.

That's, we just couldn't do it because memory was so expensive. You get so few transistors on a chip then, but it was a processor. As soon as you turned it on, it started saying, is the key being pressed? Is the key being pressed? Is the key being pressed? And if you pressed a five, it said, oh, five got pressed. I'll put a five in the display. It was that sort of thing.

So I thought, oh my gosh, you know, first of all, I had solved the output problem when Pong came out a couple of years before. Oh, a television can be a Pong game. I was a television engineer if I wanted to, so it was easy to build the digital circuits that scan the horizontal lines and the vertical frames of television data. And so that was easy for me to just build out of digital chips.

I used very few chips for everything I did. And that was the output. How do you do the input? The most expensive step on the way to the personal computer. It cost $60 then. It's probably $600 now for a keyboard that didn't even have lowercase. It was called the ASR33. That's a teletype model that has no lowercase. Oh, my gosh. That $60 was so hard to come up with. Of all the money I had to put out on the way to the personal computers, that was it.

But now I had built a device. The reason I got that keyboard was I saw a friend from the Homebrew Computer Club, John Draper. Steve Jobs and I went over to his house. He was in a basement of a Cupertino house designing a first...the engineer that lived in that house was designing a telephone you could carry around your home. kind of the first portable phone. you could carry around your phone. It would talk with the little base station. This is long before cellular or well before cellular.

And I went over and John Draper was typing on a teletype, playing a game of chess with a computer in Boston. What? And he didn't have, there's no computer account. He just phoned a number in Palo Alto that got on the start of the internet. That was the ARPANET, the first playing around with the U.S. government trying to switch packets between faraway computers. There were only six computers on the ARPANET. MIT, Stanford Research Institute, UCLA, UC Santa Barbara, University of Utah, and one other one.

And I wanted this. I want to be involved in everything. I'm excited by this stuff, and I can design anything. So I designed a little device that put letters of the alphabet on my TV, and now I could call a number that was actually... I had to pay money to call from Sunnyvale to Palo Alto. It was a pay call back then. Long distance calls cost money. And I would call and get on the ARPANET. And then I could type data to a faraway computer. I could choose which computer, log in as a guest, see some files and run some programs. And I could go to one computer and then to another and to another. And it was just so fantastic.

But I had this keyboard that worked with your TV. I did not know that a guy named Don Lancaster had had a popular electronics article about a TV typewriter. Later looked at his circuit. It was more analog, but he did some clever designing in his too. Mine was pure digital, so it can never operate differently.

And I took and I said, the first night of the Homebrew Computer Club, I thought it was, I was

told by a friend, you might want to go to this club. It's for people that have TV terminals and things or that invent them. Nobody had one, so I wanted to go. I'll be a hero. And I took my design to pass out. And I got there and everybody was talking about this Altair computer. Okay. And the Altair was sort of often gets credited to start a personal computers. But what do you mean by

personal? I'll tell you what I mean. I mean, a low cost, useful computer. Some of these computers

coming out were low cost and they were totally useless until you put a huge amount of money into

them, including all the money for a teletype that costs as much as a couple of cars. But I wanted

the whole solution. I already had a terminal talk to faraway computers. And I said, oh my gosh,

Why don't I put my computer right here?

The microprocessor is the brain. And I went for dynamic memory, which was one fourth the cost of static memory. Everybody else was coming out with these little kits. They only had static memory because they weren't real engineers. They were just copying Intel's data sheet. They knew how to make, how to copy the data sheet. But on a data sheet, there's no way you can show dynamic memory because it has complicated refreshing of every bit inside every two thousandth of a second.

So I went down. I just said, I'll put on my microprocessor dynamic memory and a little program like we had in our Hewlett Packard calculators. I called it a monitor program. And what you did was switches and lights. You toggle in a number and you push a button and it goes into an address register. Then you toggle another number and push a button and it goes into that place in memory. And you do that over and over and over and over. And I said, no, no, no.

Why don't I just write a short little program that I can type data in, in binary, the language of computers, hexadecimal actually, type it in and we'll tell it to store it at a certain location. I'd say at a certain location, or you could say certain locations, show me what's there. And it would show you all the bytes that started at that location. You could just scroll it on screen. Or the third thing you had to do with when you had switches and like main front plate on a computer, the way they were done back then. You could say, here's an address, ones and zeros on switches, push a button, it would run it, the run button. So I also said you could type in an address and say run. And oh my gosh, it was just the whole thing with absolutely no front panel. Every computer before it ever had a front panel.

And people started looking at it, the computer club, they would look over my shoulder. They'd see that I didn't have very many chips and they weren't that that expensive. And it did the whole job. I could type on a keyboard and have things show up on a display. And I started writing a basic program. Bill Gates had written a basic program for the Intel microprocessor. I couldn't afford an Intel microprocessor, $400 back then. And I could get a $40 processor, Motorola processor, because I was an HP employee. I designed it around that. But then another one came out, which was pin for pin compatible with the Motorola. So it matched all my designs that I did on my drafting table. And I bought it for $20, cash, cash, at a show in San Francisco, 6502 microprocessor.

And I sniffed the wind and I said, if people are ever going to have computers in their home, they're going to want to play games. And you need BASIC. I had never programmed in BASIC. I was a scientist. I programmed in the scientific languages, Fortran, Algol, PL1. But I said, you've got to have games. So I started writing my own basic. I went to Hewlett-Packard one night and opened up the basic manual, started making notes on the syntax and making all my diagrams. And I managed to actually, I'd never taken a class in it, but I could figure out how to do things. And it would actually, you could type in programs and store them and run them and all that. And it was based on Hewlett-Packard basic. I thought basic was basic. I found out that Hewlett Packard's basic and digital equipment's basic were totally different in how they handled characters. So it wouldn't really run the games right out of the book.

But I took it down and showed it off at the club and got interest. Every computer, since I was showing that one to the club, every computer that came out was a keyboard and a video display. That was the paradigm had changed the world. It was, I wanted all these people in Homebrew Computer Club, and this would include Paul Terrell, wanted people to someday, the day that when everyone would be able to afford computers, or many of us at home would afford computers, and they would help us do things we couldn't do without them, and they

would help education, and they would make the geek an important guy, you know.

And so I had a very useful format, but I wanted these other people in the club to help start a revolution. We sometimes spoke of a revolution. We had Stanford professors and people from Berkeley, you know, telling our club what, how society was going to change. And I wanted all these people, maybe there were 250 of us, whatever could fit in that Stanford Linear Accelerator Center Slack auditorium. And I just wanted to help them. So I gave away my designs, no copyright notices, public domain, open source, everything. Let's do it. You could get these parts and build a working, useful computer for $300 in parts. You just had to wire it yourself.

Most of the club people in the club were not like myself. They were not builders and wirers who would solder wires. So they just wanted to buy something kind of more built.

Well, then it was around this time, Steve Jobs came into town. And he'd never been to the club. And there's a movie that shows him hauling me down to the club. That's so phony. I'd been to every club meeting since the day it started. the most important day of my life, every other Wednesday. And I took him down with my TV set and my board and my keyboard and demonstrated it, and he saw the interest in it. And that's when he said we should start a company.

But he'd been coming in and selling my stuff for five years. You know, once a year he'd come into town, what is my latest invention, a pawn game, whatever, and do that. I even wound up designing Breakout with him for Atari. He didn't design it, but he helped be part of it.

Anyway, and I believed in games, and I had this computer going there. And so Steve had this idea. We would have a company, and the way we would make money is he knew how to make a PC board. He'd learned that during our blue box days. He made a PC board for our blue box, the illegal device that could make free calls anywhere in the world.

And so he knew PC boards. So his idea was we'll invest some money. I had to sell my most valuable possession, my HP 65 calculator for 500 bucks. And he sold a van. And we got some money to pay a guy to lay out a PC board. We'd build the PC boards at a place in Santa Clara for $20 each. And then we'd sell them for 40. Oh, my gosh. To get our money back, we might have to sell 50 of them. I didn't know 50 would buy this computer, this, you know, computer at our club. But we said, oh, so what? We'll have a company for once in our lives. You know, you don't have to make money necessarily.

now it wasn't too long into the when we decided to go ahead and start making this and we um and um you know we didn't have any money that's why we're gonna pay 20 bucks per board and sell them for 40. Paul Terrell at a Byte Shop he had the belief that this was not just a little one thing going on this was going to be a future a future for the world small low-cost computers and, and he said ours was the best.

And so he worked out a deal with Steve Jobs to we would supply 100 computers at $500 each. That's $50,000. May not sound like that much today, but that was twice my salary at Hewlett Packard. Scary when you don't have any money. I had no savings account. Steve had no savings account. Where are we going to get the money to build these? You know, $50,000 deal.

But Paul Terrell was the first one to take a step; that was actually taking a financial risk and that, and actually believing in this market, seeing the future was really positive.

FLOPPY DAYS : Were you actually with Steve when you were talking with Paul about selling the Apple I?

WOZ : No, I was a designer. I was a designer. No, I was a designer, almost always working on designs, proving that. And Steve was a businessman. And Paul Terrell, I think, contacted Steve.

FLOPPY DAYS : Did you go into the Byte Shop at all yourself? I mean, were you a regular in the store?

WOZ : Not before this deal. I don't think before this deal I did. I wasn't a "look for a computer to buy" person. They were all Terrell's all around.

FLOPPY DAYS : But you were after the deal?

WOZ : Yeah. Oh, yeah. I went to the Byte Shop. And not too many times.

But anyway, the deal got worked out. So where are we going to get the money? Allen Baum, who is the person who told me there's this club of people interested in technology starting up. You know, and my video terminal would be popular. They'd want those sort of people. He was the one, he and his father loaned us $5,000. Now, the rest, when the parts, we had parts to chips because Paul Terrell wanted things completely built. I think Steve fooled him a little. Steve worked a deal as marketing wise that we'd only give him a board with the chips on it. And he still had to like Ikea, plug a keyboard in, plug some transformers for power, plug in a video display. He had to, it was a build it yourself. It wasn't really that complete, but a store could make it kind of complete, wooden case, whatever.

And so the chips cost the large amount of the money. I mean, where are we going to ever get $50,000? The chips would come out of boxes just as the PC boards got made. And at the same company that was making our PC boards, which I forget the name, they had a lot of assemblers that would assemble all the chips in and then put them over a wave solder and solder them to the boards. And it only cost $13 to have each board completely made with chips soldered in. Why are they selling all these kits of computers? That's how the other people were trying to start it. Sell kits of parts with instructions how to build it if it only costs $13 to go from the design to the building of it.

So anyway, we did that. And then we would bring about 10 or 20 of them to the house. Eventually, we had to move out to the garage. And it was Steve's house. And we moved to the garage. And it was never a company. There were never like tables set up. That's a mistaken thing. You know, every desks that have phones and manuals. No, it was nothing like that. Was there ever a computer design from Apple ever discussed in this garage? No. Was there ever any engineering done in the garage? No, I did all the engineering. Was there any discussion of future products? It really wasn't a company.

But we'd have to take these things that were assembled for us at a company in Santa Clara, put them down and turn them on. They wouldn't work. And I'd sit there with an oscilloscope and I found out, just determined some methods to find the usual failures. Usually when Steve Jobs' sister, who was paid a dollar a board to plug the chips in, sometimes a pin, one pin of the chip would bend under and not go in the socket. And by problem with the oscilloscope, you kind of find one line blank and pull that chip out and fix it. Or you'd find a mixed signal. it wasn't zero and it wasn't one. It was kind of in between. That meant two pins were shorted on the board. And we'd find these traces on the board. The PC boards weren't all that good. And we cleared the traces. Those were the two types of errors.

And eventually we get through most boards. Didn't work, right? As soon as we connected them up to a keyboard and a power and a monitor. And we put the good ones in white boxes and Steve would drive them down to the Byte Shop and give them to Paul Terrell for cash. Remember, we had 30 days credit on the parts and we had a loan and got paid cash.

And we didn't have big complicated scheme going through company lawyers and approvals and sign-offs. So it was fast enough. We could kind of cycle them in 10 days. And Paul paid us the cash. And that was just a start because we knew the Apple II was coming. That was just an incredible breakthrough coming up and that would be worth real, much bigger money. And I don't know if Steve

Jobs ever told Paul Terrell that we had that even before we delivered an Apple 1 to Paul.

FLOPPY DAYS : So you already had an Apple II in mind before?

WOZ : No, I hadn't mind before, but I had it actually built before we ever delivered an Apple 1. And it was based largely on some schemes of violating the rules of color TV by doing it digitally, violating the mathematics in all the books on color TV. I did a lot of things so far out of the box in all my designs, but especially on the Apple II. And it worked. I mean, we had colors, but they weren't even legal necessarily, but it just worked on every TV.

So the Apple II was coming, and that would be a big one because, as I mentioned before, games were going to be the heart of the start of computers in the home. You didn't get a computer in the home to do inventory levels and sales figures. No, it better play games. I want to write programs to solve puzzles myself, to solve little puzzles or store data or whatever. I want to write my own programs, but mainly it was the games and we had the color machine.

And that machine was what got us an investment from an angel, Mike Markkula. But the Apple II was the only computer that made money, the only product made money for Apple for the first 10 years of the company. It got bigger and bigger and went public and we got bigger and bigger and the Fortune 100 level moved up and up. One product did it all. Really, that was the start of Apple.

On the business side of things: Steve Jobs tried to present himself as, oh, the big brain that thought up computers and all that. But look, it was Paul Terrell who took a risk, who believed in a market. You got to believe in the future and risk. And we didn't have money of our own. And Mike Markkula, who invested in us with the Apple II, had the same forward vision looking at it that way too.

But yeah, the Byte Shop, I look at the picture there and I don't remember that. I don't remember that picture for sure, but it was kind of a tiny little place to get to. But heck, this industry was zero when we started. Luckily, it was zero. You know, luckily it was zero for Paul as well as us. Because all the big existing computer companies and their distributors didn't see us as any threat and didn't go out and try to compete or put us down.

Publicity was astounding because it was all young people who never had success before doing this thing. So real good articles got written and got it going.

But the Byte Shop was definitely first in this area and first for us.

FLOPPY DAYS : So I want to go back to something you said earlier about people didn't get home computers to do home inventory or those sorts of things. It was to play games. I sold it to my wife as, you know, look, we can do our checking at home on a computer. You know, that's how I got my first home computer. But, you know, did we really use them for that? Not really.

WOZ : Sure. And, yeah, and that was true for the Apple 1 and the Apple II. But people started writing programs. And those programs just, it was inspiring. You'd go in and find a little program on cassette tape. for the Apple 1 and the early Apple II. And oh my gosh, it might be called checkbook. It might be color math and put up math solving problems for school. These things weren't necessarily productive. There was a lot of money for, you know, that kind of value. It wasn't like a real business yet, but boy, it was so interesting to see. You can do this and own it and your company doesn't own it. The big guy with millions of dollars and huge computers don't own it. It was that feeling. You've got it for yourself and you've got some people around that are like you, that'll help you. You know, Paul and other people, all the people in the computer club were that sort.

I like the fact that we were kind of, we were the under guys, the under guys. We weren't the big, huge, rich, wealthy companies and doing that, you know. And of course, a rich person could theoretically buy the old expensive, expensive computer for a million dollars. Or if they owned the

company, they could just have one and they could hook up the expensive teletype terminal and and actually type in and do the sort of things we did with personal computers, but it wasn't available to everyone.

Low cost, useful. And I say useful because low cost, you could buy these little kits with switches and lights that had 256 bytes of memory. The same thing I'd done, you know, five years before, you know, and that's, I'm sorry, that's not ever going to do anything useful.

FLOPPY DAYS : Right, right. So, Steve, what would you have to say to Paul Terrell about the importance of the Byte Shop in personal computing history?

WOZ : Well, the Byte Shop helped, you know, the people feel like there was a community. There were other people that believed in the same thing. And he certainly helped Apple get a start. Oh, my God, totally. And I also think the way that he did it, I admire his integrity, and I tell him that. And, you know, you've got to have some people that really are not just trying to steal from everybody and be shysters and all that.

And who would have ever put, what was the reason for putting a Byte Shop there? You had to be inspired. And that meant other people that wanted such a thing were inspired that there was a store representing us.

FLOPPY DAYS : Anything you want to say to the other attendees of the event, that might be watching this podcast?

WOZ : Oh, there might come a time where you can be helpful and you can afford to help other startups that nobody necessarily believes are going to go anywhere. And maybe they won't, but an awful lot of good come out of what to you isn't too much. If you can, you know, afford help people start companies.

I think that entrepreneurships are the most important thing in the world because entrepreneurships often come up with a new industry that doesn't just take over for an old industry. It's like new things that didn't exist. It's new generated wealth. I think it's really the most important part. We always want to get better in life, everything in life. And that's only going to come about from change. And entrepreneurship is change, innovation, creativity. And so definitely inspire it. And boy, putting yourself into it. See, I'm not a salesperson. I'm not a company runner. I'm not a company runner even.

So to start a Byte Shop would be almost too scary; out of my range. But some people who think, oh, my gosh, I can actually... I want to get in on this new thing that's starting up and I want to use some money and open a store. That is just so incredible to see a successful example, you know, that really inspired a lot like the Byte Shop.

FLOPPY DAYS : All right. Perfect. I really didn't have any other questions prepared, Steve. This came up so quickly.

WOZ : We'll have some at the event maybe.

FLOPPY DAYS : Yeah.

WOZ : People will ask the most interesting questions.

FLOPPY DAYS : Yeah, I'm sure they will.

WOZ : But people who actually attended the Byte Shop; bought their first computer there. That's probably a more important story than I have. Well, the starting of Apple is quite a bit.

FLOPPY DAYS : Yeah, that's a lot.

WOZ : And Byte did it. Byte really did it. I mean, Paul, the way Steve negotiates, Paul..., otherwise, if it weren't Steve Jobs but somebody else. He might have owned a part of Apple as we started. You know, think about that.

FLOPPY DAYS : Yeah.

WOZ : Certainly deserved it if he wanted it.

FLOPPY DAYS : That's a sobering thought.

WOZ : But it is great to memorialize the Byte Shop. And the Byte Shop to me is Paul, Paul Terrell.

FLOPPY DAYS : Yeah, yeah, he's definitely got an important part in personal computing history, as do you, of course.

That's really all I had for you, sir. I don't want to, you know, monopolize your time, but I am so excited to do it.

WOZ : Contact me if you need more.

A.P.P.L.E. In Depth:
An Apple II for Your Briefcase — The Book II

by A.P.P.L.E. Staff

The 8086YES! BOOK II distills forty-nine years of Apple II engineering into a clamshell laptop that fits under your arm — packing a genuine 6502, a Z80 co-processor, 80-column video, Disk II compatibility, and a four-cell lithium battery into roughly 120 hand-soldered DIP chips.

]BOOK II by 8086YES! — Portable Apple II Plus compatible, 2026. Applesoft BASIC prompt shown on integrated RGB LCD display.

There is something quietly audacious about the 8086YES! BOOK II. In an era when retro-computing nostalgia normally manifests as a Raspberry Pi tucked behind a vintage bezel, the BOOK II takes the harder road: it rebuilds the Apple II Plus from first principles, using approximately 120 discrete TTL chips hand-soldered onto a custom motherboard, and then folds the whole thing into a modern clamshell laptop chassis complete with an RGB LCD, a backlit mechanical keyboard, and a quad-cell 18650 lithium battery pack. The result is, to the best of this reviewer's knowledge, the world's first true portable Apple II — and it is very much a machine that earns the word "true."

The BOOK II is produced by 8086YES!, a Chinese maker collective based in Guangzhou that has previously shipped the Pocket8086 (an IBM PC-XT compatible pocket machine built around a genuine Intel 8086 processor) and the Pocket386 (a 386SX-40 MHz machine in a similar form factor). The BOOK II represents a significant technical departure from that x86 lineage: the 6502 ecosystem's bus timing, video generation, and I/O architecture demand a completely different approach to discrete-logic reimplementation, and the BOOK II handles all of it with commendable fidelity.

The BOOK II rebuilds the Apple II Plus from first principles, using approximately 120 discrete TTL chips hand-soldered onto a custom motherboard — and then folds the whole thing into a modern clamshell laptop chassis.

Availability is severely limited. Because each unit requires manual assembly of roughly 120 DIP packages — a process that 8086YES! themselves describe as "excessively time-consuming" — the BOOK II is produced in very small quantities. The Tindie listing, priced at $460 USD, sold out within weeks of its April 2026 debut. Buyers who manage to acquire one are getting something genuinely rare: a hand-built, architecturally faithful portable Apple II that would have seemed like science fiction in 1977 when Steve Wozniak first unveiled the machine that would define a generation of computing.

Historical Context: The Machine That Started It All

To appreciate the BOOK II, one must first appreciate what it is emulating — and why it has refused to emulate it in software. The Apple II, introduced on June 10, 1977, was Steve Wozniak's masterpiece of minimalist hardware engineering. Built around the MOS Technology 6502 processor running at 1.023 MHz (precisely two-sevenths of the NTSC color subcarrier frequency — a deliberate timing choice that enabled color video generation without additional hardware), the original Apple II came with as little as 4 KB of RAM and stored programs on audio cassette tape. It had

seven expansion slots, a built-in speaker, color graphics at 280×192 resolution in six colors, and a 52-key uppercase-only keyboard.

The machine was a commercial phenomenon. By 1980, Apple had completed a $100 million IPO that sold in minutes; by 1982, annual sales had reached $1 billion; by 1984, more than two million Apple IIs had been shipped. The platform spawned an entire ecosystem of expansion hardware, software titles, and — critically for the BOOK II's design — a set of established clone specifications that allow compatible machines to run original software without modification.

The Apple II Plus, which the BOOK II most closely targets, added Applesoft BASIC in ROM (replacing the integer-only BASIC of the original), improved the disk-booting ROM, and eventually standardized at 48 KB of RAM. It retained the eight-slot expansion bus architecture that made the Apple II so extensible — and it is that bus, faithfully reproduced in the BOOK II, that gives the portable machine its authentic character.

Architecture: TTL Logic, Not Emulation

The most important thing to understand about the BOOK II is what it is not: it is not a microcontroller running an Apple II emulator. It is not an FPGA implementation of the 6502 bus. It is a ground-up reconstruction of Apple II Plus compatible hardware using standard TTL (Transistor-Transistor Logic) chips of the same family used in the original 1977 design.

The motherboard implements the full Apple II memory map: 48 KB of static RAM in the base configuration, the soft-switch addresses that control graphics modes and page-flipping, the floating-point bus behavior, and the timing relationships between the 6502's two-phase clock and the video generation circuitry. The ROMs are sourced from actual Apple II compatible machines — 8086YES! uses image dumps from early Apple II clones that fall outside the specific copyright coverage of Apple's own Applesoft and Monitor ROMs. This means that when the BOOK II boots, it is running the same actual binary code that ran in those original machines, on hardware that responds to memory accesses in the same cycle-accurate way.

⚠ A Note on DIP Count

The BOOK II motherboard contains approximately 120 DIP-package integrated circuits. At the time the original Apple II was manufactured, the main logic board contained roughly 60–70 chips. The BOOK II's higher count reflects the addition of five integrated expansion cards — 80-column video, Language Card, Z80 Softcard, printer controller, and Disk II controller — that would each have occupied a separate card in a standard Apple II chassis.

The block diagram below illustrates the major subsystems and their interconnections. Note that the internal slot assignments mirror the real Apple II slot numbering convention — a design choice that ensures software using slot-specific ROM calls will find the expected hardware at the expected address.

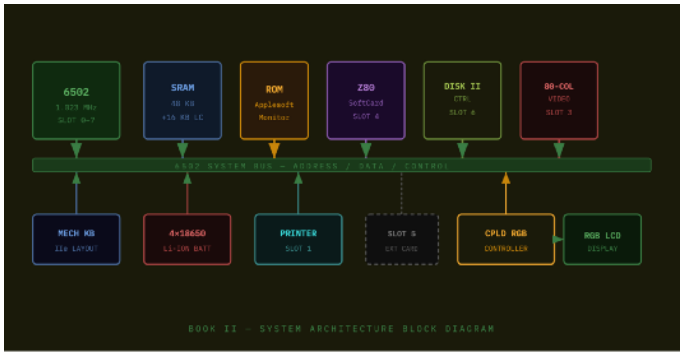


Figure 1. *BOOK II system architecture. The 6502 bus backbone connects discrete RAM, ROM, and all five integrated expansion subsystems. The CPLD-based RGB controller converts video signals from both the main board and the 80-column card for the internal LCD. Slot 5 is the only expansion position exposed externally via a 50-pin connector.*

The CPU: Staying True to the 6502

The heart of the BOOK II is, naturally, a MOS Technology 6502 or compatible variant. The 6502 is one of the most consequential microprocessors ever designed: conceived by Chuck Peddle and Wil Mathis after leaving Motorola, it was introduced at the 1975 Wescon trade show at a price of $25 — approximately one-fifth the cost of competing processors at the time. Its elegant internal architecture, minimal chip count for support logic, and clean addressing modes made it ideal for the cost-constrained personal computer market.

In the BOOK II, the 6502 runs at its canonical 1.023 MHz — the exact frequency of the original Apple II, tied to the NTSC color subcarrier through a 14.318 MHz crystal divided by 14. This is not an arbitrary choice: Apple II software frequently depends on cycle-counting for timing-sensitive operations, particularly in game code that polls the vertical blank signal or uses speaker-toggling routines to generate audio. Running at any other frequency would silently break a significant fraction of the software library.

The 6502's address and data buses connect to the same discrete TTL glue logic that the original Apple II used to implement bank selection, soft-switch

decode, and I/O register access. Apple II memory-mapped I/O operates through a tightly specified set of addresses in the $C000–$CFFF page, and the BOOK II implements all of them in hardware, meaning that original machine language programs that poke and peek those addresses will find exactly the hardware they expect.

The Z80 Softcard: Dual-Processor Computing, 1980-Style

One of the BOOK II's most intellectually interesting integrated features is its Z80 Softcard implementation on Slot 4. The Microsoft Z-80 SoftCard was Microsoft's very first hardware product, introduced in 1980, and it solved a problem that was central to early microcomputer business computing: how to run CP/M on an Apple II.

CP/M — Digital Research's Control Program/Monitor — was the dominant business operating system of the late 1970s and early 1980s. It ran on the Intel 8080 and its derivatives, including the Zilog Z80. Apple II owners who wanted access to the CP/M software library — which included early versions of WordStar, dBASE II, and the original Turbo Pascal — needed a way to run Z80 code on their 6502 machines.

The solution was architecturally remarkable: since the 6502 cannot be simply halted while another processor uses the bus, Microsoft's SoftCard employed a simulated DMA operation to pause the 6502 and monitored the Z80 refresh line to coordinate memory access. The 6502 was still required to handle I/O, timing, and system services even while the Z80 executed CP/M code. Dedicated address-translation circuitry prevented stack and buffer conflicts between the two architectures' very different memory models.

The BOOK II integrates a compatible implementation of this coprocessor arrangement, meaning that CP/M software can run natively on the BOOK II just as it did on SoftCard-equipped Apple IIs in 1980. From a software-compatibility standpoint, the BOOK II is simultaneously an Apple II Plus and a CP/M machine — a dual-personality that was considered a premium feature forty-six years ago.

The Display System: From Composite to RGB

The BOOK II's display system is where the most significant modern engineering appears, and it is a genuinely elegant solution to a hard problem.

The original Apple II generated composite video — a single analog signal that mixed luminance and chrominance information according to the NTSC standard. Color was encoded using the phase relationship between the color burst reference and the pixel-rate signal, which meant that color and horizontal pixel position were inseparably linked. This is why Apple II Hires graphics have only six colors in color mode: adjacent pixels of opposite polarity generate complementary colors through the NTSC chroma decode circuitry of a color television.

On a modern RGB LCD, composite video tricks don't apply. The BOOK II's solution is a CPLD (Complex Programmable Logic Device) that implements the entire video conversion in pure digital logic — no frame buffer, no microcontroller, no resampling. The CPLD monitors the raw video data as it comes off the mainboard, reconstructs the intended pixel pattern from knowledge of the Apple II's video generation algorithm, and converts it to direct RGB values for the LCD controller. Because the image is not resampled, each pixel on the LCD corresponds exactly to a pixel in the original television signal — a property that preserves the authentic sharpness of text mode while giving the user control over Hires rendering behavior.

The LCD controller switch on the rear panel selects among five display modes, which can be thought of as a virtualization of the monitor type that would have been connected to an original Apple II:

SWITCH	MODE NAME	BEHAVIOR IN HIRES	RECOMMENDED USE
1	80-Column	Video from 80-col card	Word processors, spreadsheets
2	Half-Pixel	Excess pixels masked; color preserved	Mixed color/text software
3	Full-Pixel	Identical to color composite monitor	Color-specific graphic demos
4	Monochrome	Color stripped; monochrome only	Productivity software, crisp text
5	Reserved	–	Future use

A separate text color selector switch (positions A through E) controls the color applied to character-mode text throughout the display — white, grey, green, orange, or yellow — using the CPLD's logic without any modification to the video signal seen by an external monitor connected to the secondary video output.

Expansion Slot Architecture: Eight Slots, Five Integrated

The Apple II's eight expansion slots were the architectural feature that most distinguished it from contemporary personal computers. Where the Commodore PET and TRS-80 offered limited or no expandability, the Apple II's bus exposed nearly every significant signal on the 6502 — address lines, data lines, I/O select strobes, phase clocks, the SYNC signal, and the RESET and IRQ lines. Any card in any slot could address the full 64 KB

memory space, generate video signals, respond to I/O selects, and trigger interrupts.

The BOOK II integrates five of the most common expansion cards directly onto the motherboard, assigning them the canonical slot numbers that software expects:

SLOT	FUNCTION	SOFTWARE ADDRESS	STATUS
SLOT 0	16 KB Language Card RAM	$C080–$C08F	Integrated
SLOT 1	Printer Interface (CN36)	$C090–$C09F / $C100	Integrated
SLOT 2	(Not populated)	$C0A0–$C0AF	–
SLOT 3	80-Column Video Card	$C0C0–$C0CF / $C300	Integrated
SLOT 4	Z80 Softcard Co-Processor	$C0D0–$C0DF / $C400	Integrated
SLOT 5	External Expansion (50-pin)	$C0E0–$C0EF / $C500	**External connector**
SLOT 6	Disk II Controller	$C0E0–$C0EF region / $C600	Integrated
SLOT 7	(Not populated)	$C0F0–$C0FF	–

The only user-accessible expansion bus is on Slot 5, exposed as a 50-pin connector on the rear panel. This connector provides the full Apple II bus, allowing a user to plug in any original Apple II peripheral card — a modem card, an accelerator, a SCSI controller, or even a second disk interface — using hardware designed and sold decades ago. Given the thriving market for Apple II peripheral reproductions (the Uthernet II ethernet card, various RAM expansion cards, and SD card disk emulators are all actively manufactured), the inclusion of a real bus slot means the BOOK II is genuinely expandable rather than merely nostalgic.

Storage: Disk II and the Floppy Emulator Provision

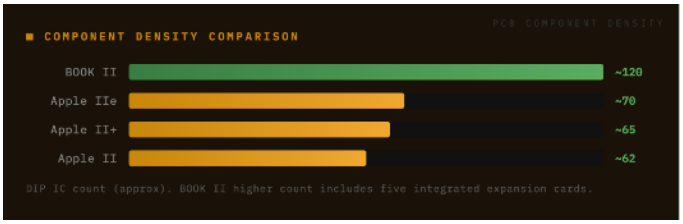
The BOOK II's Disk II controller implementation supports both the original 5.25-inch Apple Disk II floppy drive — introduced in 1978 at $495 and one of the most elegant floppy interfaces ever designed, using a cleverly demodulated GCR encoding scheme that Wozniak implemented almost entirely in software — and modern floppy drive emulator boards that accept SD cards or USB storage.

The external 20-pin Disk II interface on the rear panel allows connection of the original drive hardware for users who want the full period-authentic experience, including the distinctive mechanical chatter of the Disk II stepper motor seeking tracks. More practically, a reserved internal mounting position with pre-installed screw bosses allows a compact floppy emulator board to be installed inside the BOOK II chassis, providing modern solid-state storage in the original form factor. Users with the technical capability to design a custom emulator board are provided with reference dimensions; commercially available emulators designed to the Apple Disk II pinout should be mechanically compatible with minor adaptation.

Power System: Four 18650 Cells and a 12V Adapter

The BOOK II's portability is made possible by a four-cell 18650 lithium-ion battery pack. The 18650 form factor — named for its 18mm diameter and 65mm length — is the dominant cell format for high-capacity portable electronics, used in everything from laptop computers to electric vehicles. A typical high-quality 18650 cell delivers 3,000–3,500 mAh at 3.7V nominal; four cells wired in series provide approximately 14.8V with a total capacity of 11–14 Wh.

The BOOK II also accepts a 12V, 3A external power adapter (36W), which both operates the machine and charges the internal cells. The cooling fan visible in the rear panel IO list suggests that the TTL logic's collective heat dissipation is sufficient to require active thermal management — a reasonable expectation given that 120 DIP chips, each dissipating tens of milliwatts, can collectively produce several watts of heat in a compact enclosure.



Mechanical Feel

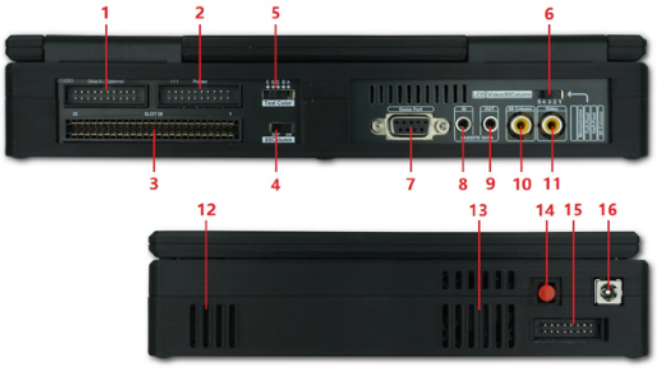
The BOOK II's keyboard is one of its most user-visible departures from strict period authenticity — and it is a welcome one. The original Apple II had a 52-key uppercase-only keyboard with no directional cursor keys, no numeric keypad, and reset and control keys that could cause serious frustration for users accidentally brushing them during fast typing. The Apple IIe, released in 1983, improved matters substantially: it added lowercase

support, dedicated cursor keys, and a more complete punctuation set.

The BOOK II's keyboard features a layout more closely resembling the IIe, with more keys than the original Apple II — a pragmatic choice for a machine intended for actual use rather than museum display. The low-profile mechanical switches with blue backlighting provide tactile feedback absent from membrane keyboards while keeping the unit's overall height suitable for a laptop-style chassis. The blue illumination is a distinctly modern touch that would have seemed otherworldly to a 1977 Apple II user but is entirely welcome for modern retro-computing enthusiasts.

I/O Reference: All Sixteen Rear-Panel Connectors

The BOOK II rear panel is generously populated with sixteen connectors, switches, and ports. The following reference map documents each interface:



[1] 20-pin External Disk II drive connector	[2] Printer interface (20-pin ribbon → CN36)
[3] 50-pin system bus slot (Slot 5 external)	[4] 80-column card enable/disable switch
[5] Text color selection switch (A–E)	[6] RGB LCD control switch (positions 1–5)
[7] DB9 game controller (Apple IIe compatible)	[8] Cassette data input
[9] Cassette data output	[10] 80-column card video output (external monitor)
[11] Motherboard video output (external monitor)	[12] Speaker
[13] Cooling fan	[14] Power switch
[15] Game joystick interface (alternate)	[16] Power adapter input (12V, 3A)

The cassette interfaces — connectors 8 and 9 — are worth noting. The Apple II's original storage medium was audio cassette tape, using a simple FSK encoding scheme that wrote and read data at approximately 1,500 bits per second. While no serious Apple II user is loading programs from cassette in 2026, the cassette port has long served as a convenient general-purpose audio I/O channel, and some software demonstrations use it for sound input. Its inclusion on the BOOK II is a completeness gesture that hardware purists will appreciate.

The presence of two separate video output connectors — one for the main motherboard video, one for the 80-column card — reflects the Apple II's historical architecture, where the 80-column card generated its own separate video signal rather than multiplexing with the main board's output. Both connectors can drive external monitors simultaneously with the internal LCD, which is controlled by the Slot 6 switch.

Comparison: BOOK II vs. Original Hardware vs. Software Emulation

Where does the BOOK II sit in the spectrum between running original hardware and running software emulation? The following comparison table maps key characteristics across the three approaches:

ATTRIBUTE	BOOK II	ORIGINAL APPLE II+	MAME / EMULATOR
Real 6502 Silicon	✓ Yes	✓ Yes	✗ No
Original Bus Timing	✓ Accurate	✓ Authentic	~ Approximate
Z80 / CP/M Support	✓ Built-in	~ With SoftCard card	~ Varies by emulator
80-Column Video	✓ Built-in	~ Requires card	✓ Software-simulated
Battery Portable	✓ Yes	✗ No	✓ (host device)
Disk II Compatible	✓ Original drives	✓ Original drives	~ Disk images only
Expansion Bus	✓ Slot 5 (50-pin)	✓ All 8 slots	✗ No
ROM Source	~ Compatible clone ROMs	✓ Original Apple ROMs	~ Dumps (varies)
Modern Display	✓ RGB LCD	✗ Composite/ NTSC only	✓ Host display
Approx. Retail Cost	$460 USD	$150–$400 (used)	Free / $0

The BOOK II occupies a distinctive niche: it is more portable and more capable (in terms of integrated features) than an original Apple II Plus, while providing a degree of hardware authenticity that no software emulator can match. The cycle-accurate 6502, real bus timing, and genuine DIP-chip implementation mean that edge-case software behavior — copy-protection schemes that rely on non-standard track access timing, programs that use undocumented 6502 opcodes, demos that exploit video timing glitches — will work correctly on the BOOK II where it might fail on even the best emulators.

Software Compatibility and Practical Usage

The BOOK II's target compatibility is Apple II Plus, which means approximately the first four years of the Apple II software library — from 1977 through

roughly 1981 — should run without modification. The integrated 80-column card extends this to include productivity software written for the Apple IIe's expanded display, though software that requires the enhanced IIe's 65C02 processor (rather than the original NMOS 6502) or the IIe's double-resolution graphics mode will not run correctly.

Activating the 80-column card requires typing `PR#3` from the Applesoft BASIC prompt — the same command used on a real Apple IIe with a 80-column card in Slot 3. Simultaneously, the LCD controller switch must be moved to position 1. Some productivity software — particularly CP/M applications accessed through the Z80 Softcard — will auto-detect the 80-column card on startup and switch modes automatically. Users who want to prevent this auto-detection can disable the 80-column card via the rear-panel switch before powering on, causing the system to present as a 40-column Apple II Plus.

The printer interface has been tested with the OKI 5200F impact printer; other Centronics-compatible printers using the standard Apple II printer card protocol should also function, though compatibility has not been documented beyond the OKI test case. The DB9 game controller port is identical to the Apple IIe's implementation and accepts the A2-JOY and any IIe-compatible joystick or paddle controller.

Availability and Pricing

The BOOK II sold initially at $460 USD on Tindie, including shipping to the United States. At time of writing, the listing shows as sold out since May 17, 2026. The extreme labor intensity of assembly — 8086YES! cites the 120 DIP chip count as making the process "excessively time-consuming" — means that production runs will remain limited, and prospective buyers are advised to monitor the Tindie listing and the 8086YES! website at *8086cpu.com* for restock notifications.

For context, $460 for a portable Apple II Plus compatible machine with five integrated expansion cards, RGB display, and mechanical keyboard represents reasonable value for what the hardware provides. Original Apple II Plus machines in good working condition typically sell for $150–$400 on vintage computing marketplaces, without the 80-column card, Z80 Softcard, Language Card, modern display, or portability. The BOOK II's all-in-one nature substantially simplifies setup and eliminates the hunt for period-correct expansion cards.

Assessment

The 8086YES! BOOK II is a remarkable piece of engineering: an architecturally faithful, hardware-accurate Apple II Plus compatible computer built as a portable laptop in 2026. Its use of discrete TTL logic rather than FPGA or microcontroller emulation is a principled choice that preserves the authentic behavior of original Apple II software at the hardware level, including all the timing-dependent and bus-behavior-dependent quirks that define the platform's character.

The integration of five common expansion cards — the Language Card, 80-column video, Z80 Softcard, printer interface, and Disk II controller — into the motherboard transforms what would have been a bare-bones Apple II Plus into a fully equipped productivity and development machine without the need for a separate card cage or power supply. The 50-pin external bus connector on Slot 5 preserves real expandability for users who need capabilities beyond the integrated feature set.

The CPLD-based RGB display system is the BOOK II's most original engineering contribution: a pure-logic video converter that faithfully translates the Apple II's NTSC-timed video signals to a modern LCD without resampling or frame-buffering, while providing useful display mode options that the original hardware never offered. Half-Pixel mode in particular — masking excess pixels to deliver monochrome sharpness while preserving color information — solves a real usability problem for Apple II text software on color displays.

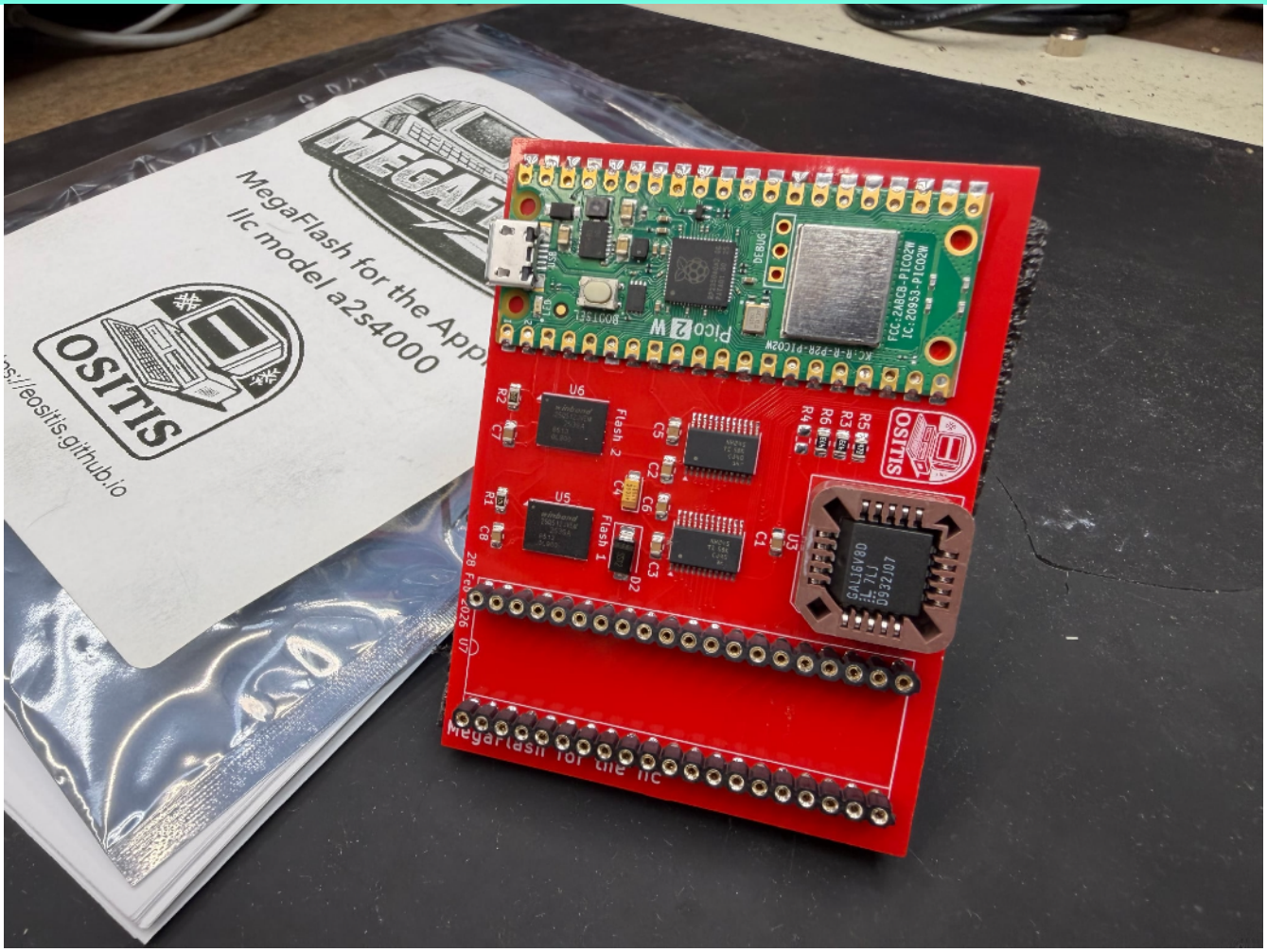
For the serious Apple II enthusiast, the retrocomputing collector, or the embedded systems engineer who wants to understand how a complete early microcomputer system was actually built in TTL logic, the BOOK II is an exceptional machine: portable enough to carry in a briefcase, faithful enough to run the full Apple II software canon, and ambitious enough to represent a genuine achievement in contemporary retrocomputing hardware design.

▪ Where to Learn More

Full documentation, motherboard PDFs, and ordering information are available at the 8086YES! website: **https://8086cpu.com/Z80/6502/110.html**. The Tindie listing (when in stock) is at: **https://www.tindie.com/products/cycle/book-ii-apple2-plus-compatible-laptop-computer/**. Technical documentation including the BOOK II User Manual PDF is linked from both pages.

Getting to Know MegaFlash

by Elmars Ositis / A.P.P.L.E. Staff



MegaFlash is probably one of the most significant hardware upgrades for the Apple IIc line of computers in many years. MegaFlash was designed by Thomas Fok, and I first learned about it over a year ago in a cryptic post in the Facebook Apple Enthusiasts group. The project outline very closely matched what I envision as a path to bringing modern features to the Apple IIc platform.

A Little Backstory

Thomas started this project in the summer of 2023. First working on the SmartPort firmware. Initially the development work was done in the AppleWin emulator, and development done using the cc65 toolchain. The first development board was built to test the hardware interface between the Pico and the Apple II bust. Thomas found the hardware side to be relatively simple, with most of the work being in on the software side. Version 1.0 was completed in April of 2025.

Thomas developed MegaFlash to work on the RAM expansion port available in the Apple IIc model A2S4100 and Apple IIc Plus. This is not actually a RAM expansion port, but rather a RAMdisk port for the famous Apple ‘Slinky’ RAM disk product. There are a number of limitations to this port, most importantly, address lines A2 and A3 are missing.

When Thomas published the project on GitHub (https://github.com/thomasfok/megaflash), I had just completed designing a 6502 processor adapter for Dr Guzman’s LogicAnalyzer project. As I dug through the design docs, I realized MegaFlash could easily be adapted to work with the original Apple IIc via the CPU socket - the same way RamWorks, Mockingboard and a few others used to connect to the Apple IIc motherboard.

So… What is MegaFlash?

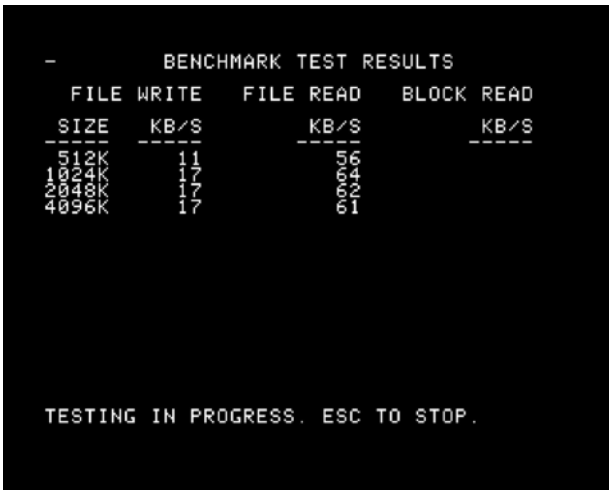
At its core, MegaFlash is permanent solid-state disk storage for the IIc platform - with performance matching the ‘Slinky' RAM expansion card from Apple. However, where standard Slinky cards are limited to a 1 MB of storage, MegaFlash offers 4x

32MB (or more) volumes of permanent internal storage. In addition, MegaFlash adds a 400KB RAM disk, and a ROM disk with all the critical utilities needed on an Apple II.

MegaFlash resides on logical slot 4. On the IIc platform this is a very special slot, as Apple designed the IIc to boot from the Slinky card, when installed. With MegaFlash installed, the IIc automatically boots from the first assigned SSD Slot. If you break out to the system, PR#4 will boot vol1, IN#4 will boot from vol2. If you have a FujiNet, FloppyEMU or SP][SD installed, PR#5 boots the system from those. PR#6 works as always, booting from the floppy disk.

It is important to note that MegaFlash is not designed as a replacement for the SmartPort devices we know. MegaFlash is designed for maximum storage performance on the IIc platform. To contrast: data transfers from SD-card based systems are relatively slow. The data connection to the MegaFlash onboard flash memory is 3x faster than can be achieved with an SD-Card. This translates to superb performance on an unaccelerated IIc, and if you have a ZipChip or IIc+, then storage performance is even more extreme.

Most projects stop there, but Thomas took MegaFlash to the next level. Thomas went to work on the system ROM. In addition to the already familiar ROM4x/5x bug fixes, Closed-Apple-CTRL-Reset gets you to the MegaFlash Control Panel. Thomas then focused on AppleSoft, fixing bugs and adding an optional redirect to the Pico on the MegaFlash for match calculations. With AppleSoft FPU acceleration enabled in the MegaFlash control panel, SIN/COS functions will calculate 5x faster, square root - 7x faster, as examples. As a practical example, the Mandelbrot program we ran as kids runs about 20% faster with the AppleSoft math acceleration enabled.



MegaFlash also includes the beginnings of a network stack for the Apple IIc. The MegaFlash real time clock (RTC) is synchronized to the correct time on the internet using the NTP protocol over WiFi. From the Control Panel, you can also import and export disk images using the TFTP protocol.

The Development is in the Details

What you have seen is just the beginning. During quiet periods, while waiting for parts to arrive from suppliers, I have been working on porting Uthernet II emulation to the platform. Maybe, by the time this article goes to press, the initial test packets will be flowing.

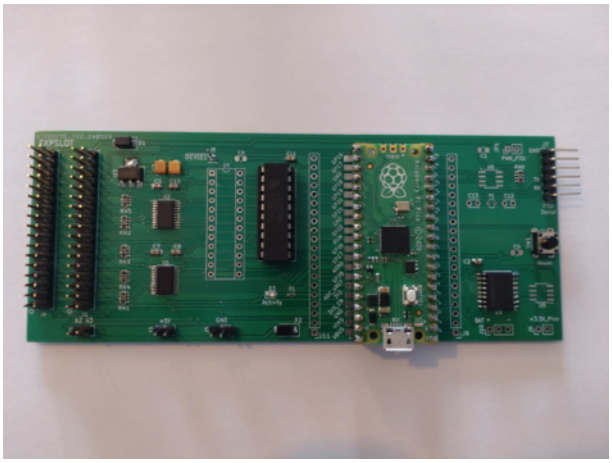
To be clear: performance claims should always be taken with a grain of salt; it all depends on how the machine is used. To test firmware updates and validate changes, I have had to work on performance testing tools for regression testing. For storage performance, I updated *Stress ProDOS Filesystem* (S.P.F.) to use the MegaFlash clock. To test AppleSoft performance improvements, I have built a new CPU performance testing tool called a2speed. Both are available in my Github repository (link at the end), and I will be sharing the MegaFlash clock patch with the S.P.F. developer, so his version is also up to date.

The road has not been without its challenges. For Thomas, the first issue was the communication and synchronization between processors. The Pico has

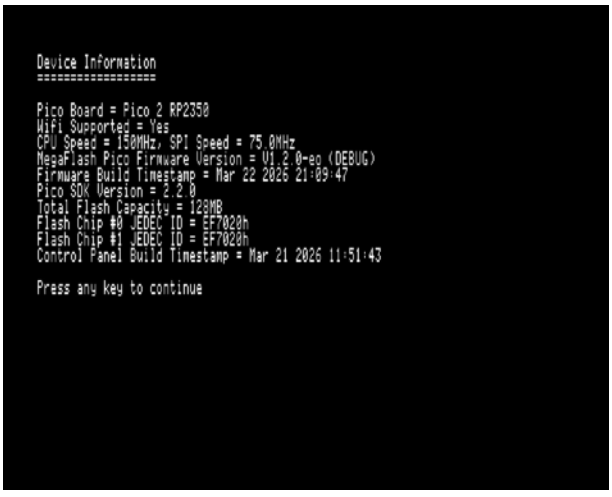
a dual-core CPU. It also has an I/O processor called PIO. Together with the 65C02, 4 processors are running at the same time. When something did not work as expected, it took a long time to figure out which processor was causing the problem.

The second issue for Thomas was the lack of documentation for the Apple IIc Plus. MegaFlash was built on this platform, specifically for this platform. There is no firmware listing of IIc Plus ROM. There are no detailed descriptions of how the accelerator and MIG chip work.

Making it Real

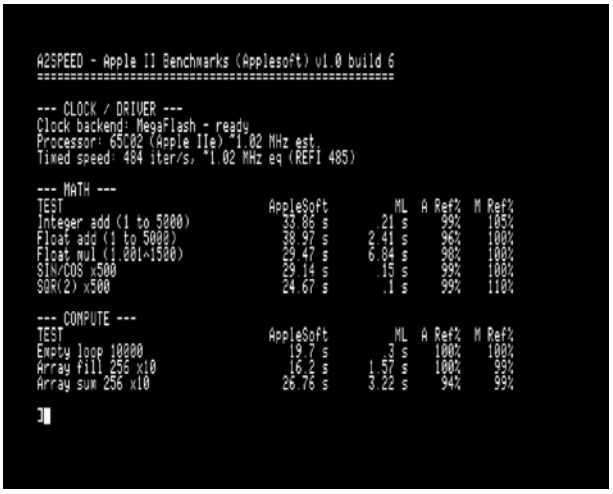


For me, I have made plenty of projects for myself. However, once I had built and tested a MegaFlash for myself, I understood that MegaFlash is something that has to get to market: it is revolutionary. Getting there was the challenge, and is still a challenge. With one-off projects, one can grab some parts, throw them together and be amazed that it works. For a product, one must consider how to manufacture it consistently. The question must be asked, “how easily can a semi-technical or non-technical user install the product?”



Parts cost and availability have been a huge challenge. The tariff fluctuations this past year, am I right?! PCB prototyping and production costs have skyrocketed. Production costs have stayed the same, but customs and customs handling fees have shot through the roof. If I am not careful on how to place and ship the order, the cost of the PCB can triple by the time it lands on my workbench. PCBway has been a big help here. Once they learned about the project, they offered to help with the PCB development costs. This has been a huge help getting the board to market.

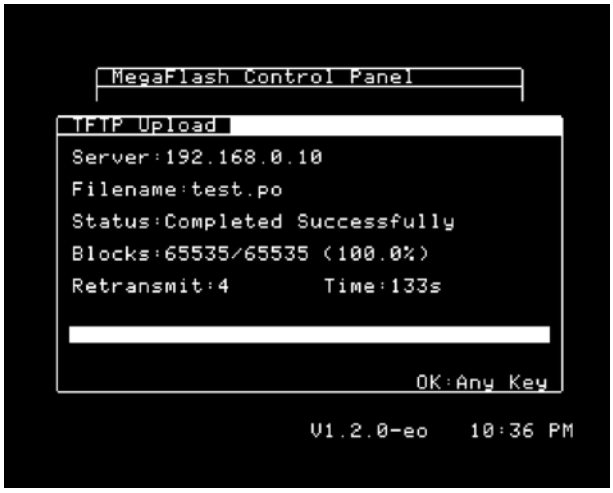
What’s Next?



Thomas and I are not done developing the software. The initial product on the market is for the original IIc and RAM expansion IIc (models a2s4000 and a2s4100). For these machines, MegaFlash will plug into the CPU socket, where it has access to all of the address lines.

For the IIc Plus, I hope to have the market version ready by the time this article goes to press. This version will use the RAM expansion port and connector J14, which exposes the missing A2 and A3 pins. With MegaFlash connected to these interfaces, we avoid several timing and cache issues that would be an issue if connected to the CPU socket.

Address lines A2 and A3 are important, as with those, MegaFlash has full access to all 16 IO addresses for slot 4. This is what will allow more functionality to be added, such as the Uthernet II emulation mentioned earlier.



Thomas and I both want to get MegaFlash ported over to the rest of the Apple II platform. Will it be exactly like on the IIc? Hard to say at the moment.

Where can you get MegaFlash?

At time of publication, MegaFlash for Apple IIc is available on my Tindie store (link here) and at Joe's Computer Museum (https://jcm-1.com)

Who are we?

Thomas Fok is originally from Hong Kong, now lives in the UK. He started his career as an electronics engineer, but has since moved on to greener pastures, although electronics is still his hobby. His first computer was an Asian Apple IIe clone, and the inspiration for the electronics and computing industry.

Elmars Ositis grew up on the outskirts of Silicon Valley. His first computer was one of the very first 16k Apple Ii computers every made, and has been a lifelong Apple user. Grew up in a household with his parents' cottage electronics manufacturing business in the basement, and daily exposure to the latest changes the computing industry.

Public reference data for additional color:

https://eositis.wordpress.com/projects/apple-IIc/

https://eositis.github.io/megaflash

Https://github.com/thomasfok/megaflash

Email for clarifications, etc - not to be published

Elmars Ositis - eositis@ois.lv

Thomas Fok - thomas@melody-soft.com

DOCUPAK VOLUME 1

APPLE PUGETSOUND PROGRAM LIBRARY EXCHANGE
A Washington State Non-Profit Corporation

THE WOZPAK][

and Other Assorted Goodies

A Collection of APPLE II Documentation
Furnished Through Courtesy of
APPLE COMPUTER, INC.
Assembled and Distributed to Members of
APPLE PUGETSOUND PROGRAM LIBRARY EXCHANGE

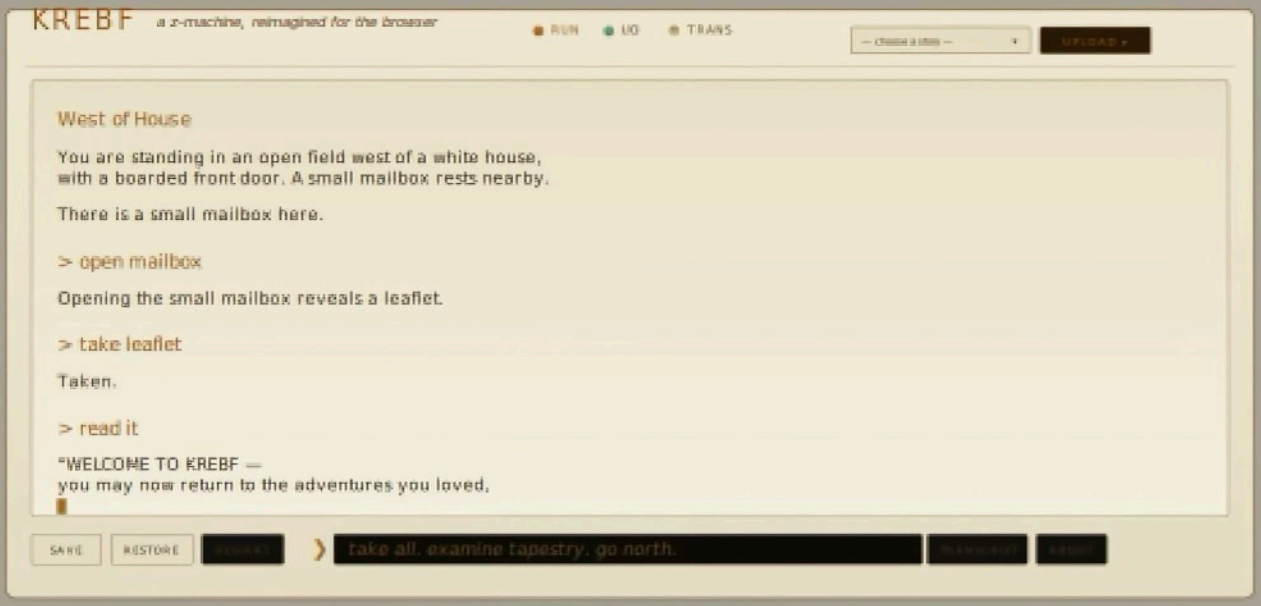
Revised Edition

6708 39th Avenue S.W. Seattle, Washington 98136 (206) 932-6588

APPLE PUGETSOUND PROGRAM LIBRARY EXCHANGE · 2026

KREBF

a z-machine,
reimagined for the browser



Play every Infocom title, on any WordPress page.

A complete Z-Machine interpreter in a single shortcode.
Drop [krebf] into a post — your visitors play Zork, Trinity,
Hitchhiker's Guide, A Mind Forever Voyaging, directly in the browser.

[z]

Versions 1-8
Every classic Infocom title, every modern Inform release. Only v6 graphics is absent.

>_

One shortcode
Pull-down dropdown lists your uploaded stories. ?game= URLs deep-link to any.

Secure by default
Path-traversal proof, SSRF-defended, nonce-gated, capability-checked.

AVAILABLE NOW · FREE SOFTWARE · MIT LICENSE

callapple.org/krebf-support

Published by APPLE PUGETSOUND PROGRAM LIBRARY EXCHANGE · 2026

ATOM 20

WASTELAND SURVIVORS

The bombs have fallen. Sixty-one days in darkness.
The food is gone. The fate of mankind is yours.

A POST-APOCALYPTIC TEXT ADVENTURE

EXPLORE

81 rooms across 12 regions
A ravaged American landscape
from shelter to sanctuary

SURVIVE

Manage food, water, and radiation
10 major puzzles to solve
Every choice matters

LEAD

Recruit 8 unique survivors
Each with vital skills
Build a community from ashes

"The republic is not a flag or a document.
It is a decision, made again each morning, to organize the common life
around the dignity of every person in it."

Z-MACHINE VERSION 5 • INFORM 6 • REQUIRES Z-CODE INTERPRETER

PUBLISHED BY

A.P.P.L.E.

Apple Pugetsound Program Library Exchange

www.callapple.org

Retro 365 – Budokan: The Martial Spirit, a title that says it all

by Ernst Krogtoft / A.P.P.L.E. Staff

It is the late 1980s, and *Michael Kosaka*, a black belt and instructor of *Aikido*, has been tinkering with an idea for some time. As both a practitioner of Japanese martial arts and an experienced game designer, now with *Electronic Arts*, he envisions a comprehensive martial arts game that not only reflects the real-world techniques he knows so well, but just as importantly, manages to convey the underlying philosophy, the very spirit of the art itself.

It is an ambitious idea, one that seems, at least on the surface, at odds with the limitations of contemporary home computers. Real-world martial arts encompass a wide range of disciplines and techniques, far more than can easily be mapped to a joystick or keyboard without becoming unwieldy. At the same time, character sprites must be large enough to convey believable movement, each requiring its own set of animations, and all of it needing to translate seamlessly from input to action. It is a tall order for the time, one that has, until now, kept his ambitions at bay.

In the early years of the IBM PC, action games had often been defined by limitations. The platform, powerful in some respects, was anything but uniform. Systems varied in graphics and performance, and input relied heavily on the keyboard rather than specialized controllers. For developers aiming to create fast, responsive gameplay, it presented a constant challenge, and for

players, the results often felt inconsistent and frustrating. Fighting games proved especially difficult to realize as they demanded precision and responsiveness, with little room for delay, as timing was crucial.

Bushido: The Way of the Warrior by *Ebenel Enterprises*, released in 1983 by *Advanced Computer Products*, introduced the idea of martial arts combat to the PC. While ambitious and impressive, movement was rigid, feedback slow and uneven, and the connection between player input and on-screen action felt disconnected. Even so, it demonstrated that fighting games were not confined to arcades or to the era's dedicated game-oriented systems like the Atari 8-bit or the newly introduced Commodore 64.

Created by John and Robert Lee as the first martial arts fighting game for the business-oriented IBM PC. Bushido was impressive for its time, but its lack of responsiveness and awkward control scheme made it difficult to master. It was released by Advanced Computer Products, a computer business founded by Dave Freeman in the summer of 1976. Freeman had a background from Fairchild and National Semiconductor and had experienced the microprocessor revolution firsthand. In November of 1976, he opened up one of the earliest computer retail stores in the US to join the personal computer revolution.

As the 1980s progressed, advances in hardware, combined with a growing understanding of the IBM PC's strengths and limitations, began to shape a new generation of action titles. Many leaned toward faster, more arcade-like experiences, while a few explored a slower, more deliberate approach. Within the latter, Kosaka's idea found its place, and by 1988, his prospect of a *true* martial arts simulation seemed within reach.

With more than 20 titles to his name as a designer, artist, and animator, having worked with companies such as *Atari* and *Epyx*, Kosaka possessed a skill set that extended well beyond gameplay and mechanics into visual style and presentation. In 1987, when he joined Electronic Arts, he became the company's first in-house designer and helped contribute to its first wave of internally developed titles.

Founded by *Trip Hawkins* in 1982 with a philosophy of treating developers as "software artists," Electronic Arts had initially operated more like a label than a traditional studio. The company acted as a publisher, curating and distributing the work of independent creators. But by the middle of the decade, as games grew more complex and new, more capable platforms emerged, the company had begun building internal development teams. The lone programmer working from a bedroom was gradually giving way to structured development groups, where designers, artists, and programmers collaborated under the same roof.

Electronic Arts continued to value individual vision and experimentation, and was increasingly willing to back projects that diverged from established formulas. Kosaka's martial arts project, to become *Budokan: The Martial Spirit*, quickly found support within the growing company, and a team was assembled.

Designer and programmer *Ray Tobey* joined the project, bringing the technical expertise necessary to realize Kosaka's vision. Tobey's career traced the arc of the early game industry itself. He had entered the scene as a teenager, part of a generation for whom computers were not tools of business, but instruments of creation. His breakout success came with *Skyfox* in 1984, a fast-paced flight game that demonstrated both technical skill and an intuitive sense of gameplay. The title, picked up and published by Electronic Arts, became one of the company's early hits, selling hundreds of thousands of copies and establishing Tobey as a rising talent.

By the time work began on Budokan in the summer of 1988, Tobey was no longer an external developer submitting projects from afar. He was part of Electronic Arts' internal development core. Alongside Tobey and Kosaka was Rick Tiberi, contributing additional programming.

Michael Kosaka, Rick Tiberi, and Ray Tobey.
Image from the 1990 Sega Mega Drive manual (the same image, in color, was used for the packaging of the IBM PC version).

Drawing from his own years of martial arts practice, Kosaka wanted to create a game defined by authenticity. He didn't want a simple sequence of disconnected fights, as many arcade-style titles offered. Instead, he envisioned a structured journey, a progression that mirrored the discipline and dedication of real-life training. Designed in a way so mastery could only be achieved through practice and persistence. Techniques had to be learned and perfected, timing and positioning understood, and stamina and energy carefully managed. Repetition and gradual improvement were central to the experience, a philosophy grounded in the very principles of the art.

Kosaka and a small team of artists created a calm, orderly visual style, rendering the Japanese dojo in harmonious colors and balanced composition. The environment felt serene and purposeful, and reinforced the sense of discipline and respect. Character sprites for each of the four martial arts disciplines included around 30 moves and between 60 and 70 animation frames.

From a technical standpoint, the development of Budokan reflected Electronic Arts' commitment to staying on the cutting edge of technology. The IBM PC platform was notoriously fragmented, with systems ranging from monochrome and CGA to EGA, and now, the emerging VGA standard, introduced by IBM in 1987 with its PS/2 line of computers. The company wanted its titles to leverage new technology without leaving existing users behind. Budokan was developed to scale across standards, ensuring accessibility while also taking full advantage of the newest technology on the most capable systems. The result was one of the earliest PC games to offer 256-color VGA graphics.

The game's fitting score was composed by *Rob Hubbard*, who had crossed the Atlantic to join Electronic Arts in 1988 as the company's first employee devoted primarily to music. Already renowned for his work in the British Commodore 64 scene, Hubbard created a soundtrack that drew from traditional Japanese music, complementing the game's atmosphere and setting. He also produced the game's digitized sound effects, which underpinned movements and actions.

When Electronic Arts published Budokan: The Martial Spirit in November 1989, it was both a technical achievement and a thoughtful *simulation* with great attention to detail. The game managed to bridge the gap between authentic martial arts (and fighting games in general) and the evolving capabilities of home computers, offering players not just combat, but a journey of learning, discipline, and mastery, all in glorious 256-color, if you had the hardware to support it.

Electronic Arts published Budokan: The Martial Spirit for the IBM PC in November 1989.
It was positioned as a distinctive offering, something different from the arcade-inspired titles that dominated the market.

Budokan features four distinct disciplines: Karate, Kendo, Bo, and Nunchaku. Each requires a different approach, not only in terms of available moves, but in pacing and strategy. Weapons add reach and power, but also demand precision. Stamina affects speed and responsiveness, while ki governs the energy behind actions.

The structure provides a sense of progression, leading the player from training and sparring toward the Budokan arena itself.

Versions for other platforms followed in the years to come. In 1990, alongside a Commodore Amiga version, Budokan appeared on the Sega Genesis, marking one of Electronic Arts' early ventures into the console market. It was a period defined by negotiation and experimentation, as the company sought to establish itself beyond home computers.

Budokan was released in Europe, both the original IBM PC version (seen here) and ports for other, less capable systems.

Budokan was met with a mixture of praise and hesitation. Critics recognized its ambition, hailed its art and animation, and appreciated its attempt to move beyond genre conventions, exploring a more reflective, measured approach to fighting games. Many reviews highlighted its atmosphere and depth, describing it as a rare intellectual take on a field often defined by fast-paced action. At the same time, the game's deliberate pacing proved divisive.

Players accustomed to fast, immediate action often struggled with its controls and rhythm. It could feel slow or even frustrating, but for those willing to adapt, Budokan offered a level of mastery and nuance that was quite exceptional for the era (and still is). In this way, the very qualities that set it apart also limited its mass appeal. As the 1990s took off, the fighting genre would increasingly favor titles like Street Fighter and Mortal Kombat, leaving Budokan as a distinctive, thoughtful outlier.

Kosaka remained part of Electronic Arts' evolving creative environment, contributing to the company's ongoing transformation. He stayed with the company until 2001, holding various positions as designer, art director, and producer. That same year, he launched *Kosaka Interactive, Inc*., which operated briefly. In the years that followed, he continued his career across a range of companies, including *LucasArts*.

Sources: *Wikipedia*, LinkedIn, BionicLoad, DosDays, PocketMags, Tales From The Collection, Internet Archive…

The June / July Global Event Calendar

by A.P.P.L.E. Staff

WWDC (Apple Worldwide Developers Conference)

Next Dates: June 8–12, 2026

Description: Apple's annual developer event. June 8–12, free and online globally via Apple Developer app, website, and YouTube. Special in-person event at Apple Park on June 8 (limited, random selection from Apple Developer members). Unveils the latest iOS, macOS, and Apple platform frameworks, APIs, Swift features, and tools. Labs with Apple engineers.

Event URL: https://developer.apple.com/wwdc26/

JSNation 2026

Next Dates: June 11 & 15, 2026

Description: Major web development conference focusing on full-stack engineering. In-person Amsterdam on June 11, remote-only day June 15.

Event URL: https://jsnation.com/

Nova Demoparty

Next Dates: June 12–14, 2026

Description: Well-established UK demoparty since 2005 in the seaside town of Budleigh Salterton, Devon. Up to 100 sceners. Old and new school competitions, photography, floppy discus, live DJing. 5000-lumen HD projector.

Event URL: https://novaparty.org/

KubeCon + CloudNativeCon India 2026

Next Dates: June 18–19, 2026

Description: India edition of KubeCon. Kubernetes, containers, serverless, observability, security tracks. Growing community in South Asia.

Event URL: https://events.linuxfoundation.org/kubecon-cloudnativecon-india/

@Party (Boston Demoparty)

Next Dates: June 19–21, 2026

Description: Annual demoparty in Boston (Allston/Brighton). Multi-platform competitions (PC, Atari, Amiga, C64, Apple, ZX Spectrum). Seminars, music, and retro gaming.

Event URL: http://atparty-demoscene.net/

INIT HELLO (Apple II Conference)

Next Dates: June 19–21, 2026

Description: Apple II-focused conference at System Source Computer Museum north of Baltimore. Sessions, demos, and community gathering.

Event URL: https://init-hello.org/

VCF GB (Vintage Computer Festival Great Britain)

Next Dates: June 19–21, 2026

Description: Britain's original vintage computing festival, revived in 2026 at Leigh Spinners Mill & the Northwest Computer Museum, Leigh, Greater Manchester. Exhibitions, gaming, workshops, guest presentations, films, and live retro music. Endorsed by VCF.

Event URL: https://www.chimeraevents.co.uk/vcfgb

Pacific Commodore Expo NW (PaCommEx)

Next Dates: June 20–21, 2026

Description: Pacific Northwest Commodore expo at Old Rainier Brewery Intraspace, dedicated to C64, Amiga, and VIC-20.

Event URL: http://www.portcommodore.com/pacommex

ROMA.EXE

Next Dates: June 20–21, 2026

Description: Annual demoparty in Rome. Multiplatform demoscene competitions. Italy's leading demoparty.

Event URL: https://www.demoparty.net/unique-event/roma-exe-2026

Vancouver Retro Gaming Expo (VRGE)

Next Dates: June 20, 2026

Description: All-ages annual expo at the Anvil Centre. Vendor hall, artist alley, free-play stations, retro computer showcases by Chilliwack Retro-Computing Club, live music, tournaments, and panels.

Event URL: https://www.vancouvergamingexpo.com/

IEEE WCCI 2026

Next Dates: June 21–26, 2026

Description: IEEE World Congress on Computational Intelligence — combines IJCNN (neural networks), FUZZ-IEEE (fuzzy systems), CEC (evolutionary computation).

Event URL: https://wcci2026.org/

Open Source Summit North America

Next Dates: June 22–25, 2026

Description: The Linux Foundation's premier open source conference. Combines events covering Linux, cloud native, AI, embedded, networking, supply chain security, and IT systems. Workshops, hackathons, and tracks for developers, sysadmins, and DevOps engineers.

Event URL: https://events.linuxfoundation.org/open-source-summit-north-america/

Posadas

Next Dates: June 26–28, 2026

Description: Annual Amiga-focused demoparty in Posadas, Andalusia, Spain. Gathering of Spanish Amiga demoscene enthusiasts for competitions and community.

Event URL: https://www.demoparty.net/posadas/posadas-2026

PT Weekender (Protracker Weekender)

Next Dates: June 26–28, 2026 (Xth edition)

Description: Online demoparty focused on tracked music (Protracker format). Participants create module music over the weekend. Virtual event open worldwide.

Event URL: https://www.demoparty.net/ptweekender/pt-weekender-x

Remedy

Next Dates: June 2026 (TBA)

Description: Annual Swedish demoparty. Multiplatform demoscene competitions in a community-friendly setting.

Event URL: https://www.demoparty.net/remedy/remedy-2026

RetCon – The Retro Gaming Festival

Next Dates: June 2026 (TBA)

Description: Annual festival by Greenford Computer Club at Barnhill Community Centre. Playable vintage computers and consoles, industry legend talks. Running since 2018.

Event URL: https://retconfestival.co.uk/

Apple Folklore – Pineapple Pizza

by Andy Hertzfeld / A.P.P.L.E. Staff

Our reward

When I began working on the Mac project in February 1981, there was still only one single 68000-based Macintosh prototype in existence, the initial digital board that was wire-wrapped by Burrell himself. It was now sitting in the corner of Bud Tribble's office, on one of the empty desks, attached to a small, seven-inch monitor. When powered up, the code in the boot ROM filled the screen with the word 'hello', in lower case letters and a tiny font, rendered crisply on the distinctive black-on-white display.

Dan Kottke and Brian Howard were already busy wire-wrapping more prototype boards, carefully following Burrell's drawings. In a week or so, I received the second prototype for my office, so I could work on the low level I/O routines, interfacing the disk and keyboard, while Bud worked on the mouse driver and porting Bill's graphics routines.

The next big step for the hardware was to lay out a printed circuit board. We recruited Collette Askeland, the best PC board layout technician in the company, from the Apple II group. Burrell spent a week or two working intensely with Collette, who used a specialized CAD machine located in Bandley 3 to input the topology and route the signals, eventually outputting a tape containing all the information needed to fabricate the boards.

Burrell and Brian Howard checked and rechecked the layout, which was tediously expressed as thousands of node connections, and after a day or two they decided they were ready to send it out for fabrication. We were hoping to get the first sample boards back before the weekend, but it looked like they weren't going to make it. Finally, around 4:30pm on a Friday afternoon, they arrived.

Burrell figured that it would take at least two or three hours to assemble a board, and then even longer to troubleshoot the inevitable mistakes, so it was too late to try to get one working that evening. Maybe they would come in on Saturday to get started, or maybe they'd wait until Monday morning. While they were discussing it, Steve Jobs strolled into the hardware lab, excited as usual.

"Hey, I heard that the PC boards finally arrived. Are they going to work? When will you have one working?"

Burrell explained that the boards had just arrived, and that it would take at least a couple of hours to assemble one, so they were thinking about whether to start tomorrow morning or wait until Monday.

"Monday? Are you kidding?", replied Steve, "It's your PC board, Burrell, don't you want to see if it works tonight? I'll tell you what, if you can get it to work this evening, I'll take you and anyone else who sticks around out for Pineapple Pizza."

Steve knew that Pineapple Pizzas had recently replaced Bulgarian Beef as Burrell's current food obsession (which, as a staunch vegetarian, he thought was a positive development) and that Burrell wanted a Pineapple Pizza pretty much every chance he could get. Burrell looked at Brian Howard and shrugged. "OK, we may as well give it

a shot now. But I don't think we'll be able to get it working before the restaurants close."

So Burrell and Brian got busy, selecting a board and stuffing it with sockets, carefully soldering them in place, while five or six of the rest of us, including Steve, sat around and kibitzed. Burrell seemed a little tense and impatient, since he didn't like the pressure of bringing up a board in front of so many spectators. Every five minutes or so, he referred to the awaiting Pineapple Pizza, speculating about how good it was going to taste.

Finally, around 8pm or so, the board was assembled enough to try to power it on for the very first time. The prototype was hooked up to an Apple II power supply and a small monitor, and fired up as we held our breath. The screen should have been filled with 'hellos', but instead all that was there was a checkerboard pattern.

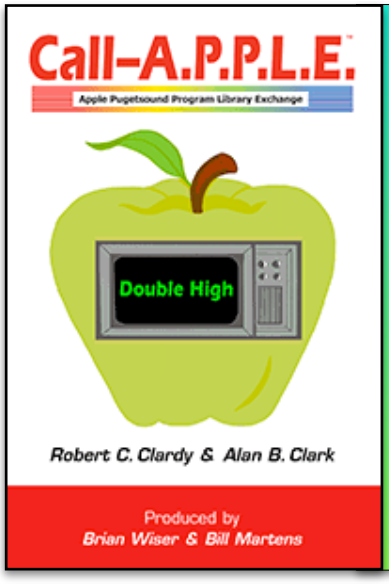
We were all disappointed, except for Burrell. "That's not too bad", he commented, "It means the RAM and the video generation are more or less working. The processor isn't resetting, but it looks like we're pretty close." He turned to look directly at Steve. "But I'm too hungry to keep working - I think it's time for some Pineapple Pizza."

Steve smiled and agreed that it was good enough for the first night, and it was time to celebrate. The seven or eight of us who stayed late drove in three cars to Burrell's favorite Italian restaurant, Frankie, Johnny and Luigi's in Mountain View, ordering three large Pineapple Pizzas, which tasted great.

"8bit-Slicks" is a remake of the legendary 90s game for the Apple 2, Commodore 64, Atari 8bit, Atari Lynx, NES, and Oric 1/Atmos. It is a game featuring:

- Up-to 4 players (local or online).
- 22 varied maps (jumps, obsctales, cross-points).
- Online competitive ranking.
- A friendly community.

Please visit the credit page for more on the history and development of this exciting game!



Double-High

Synergistic Software's
Double-High Resolution Text Editor and Tools

Features an easy-to-use editor for creation of your own unique character sets, with 46 fonts included

Available to A.P.P.L.E. Members Today!

https://www.callapple.org/apps

Orders can take up to a week for delivery, as we 3D print them and test each item individually.

Ultra High Capacity arrives.

8BitStuff's new 3.5 Floppy Emu Drive, sets a new standard for Apple II & Macintosh classic storage. HD, 800k and floppy capacity means you can stop playing diskette roulette. Simply store your large 2-disk application programs on one Micro SD card and forget about fumbling for clumsy "sets" of disks. And storing files just became twice as easy too, because you'll only need one SD card.

Compatible with all versions of ProDos and GS/OS, our design incorporates the same, top-quality BMOW technology that others use. Aesthetically, the 3.5 Emu Drive is every bit as handsome as the Apple Drive. We designed a nice, high-quality PLA case to stack with and exactly match the footprint of Apple's drive. But this drive gives you twice the capacity at an amazing price.

Our exclusive LED screen displays a menu UI for choosing from the disk images stored on the removable SD memory card, reads and writes emulated 140K, 400K, 800K, or 1.4MB floppy disk images, or hard disk images up to 2GB .

The 3.5 emu version of the 8 bit stuff drive works with the IIe, IIc, IIc+, IIgs, and classic Macs. For IIe owners, there are several controller cards, like the Yellowstone, that allows the IIe to accept data from the 3.5 drive. The 3.5 Emu Drive is also Macintosh compatible as an external floppy drive.

High capacity micro SD card is quickly becoming the standard for other personal computer platforms. Now you can bring the power and convenience of high density to your Apple computer.

Features

- Micro SD storage, will read most disk image formats from floppy to HD
- 2 LED indicators, red for on, green for activity
- Large 2.4 inch green screen, for easy reading and cool retro look
- External drive connector, looks like the original floppy drive
- Apple compatible, works as external drive with IIe, IIc, IIc plus, IIgs, Classic Mac
- Stack and footprint compatible with Apple Drive
- Conserver compatible.

Comes in four configurations:

- Basic case
- Emulator Kit
- Complete emulator (flat cable)
- Complete emulator (original cable)

Order today!

Find us at 8bitstuff.com

8-bit Stuff, where '80s style meets modern power.

CIRCLE 173 ON READER SERVICE CARD

HOW TO SHUT DOWN A DIMENSIONAL RIFT

First, you need a dimensional rift to shut down. (That's easy if you've got MOON BASE ALPHA,™ the sci-fi mystery from A.P.P.L.E.'s interactive fiction line. Because you'll be exploring an abandoned lunar base while uncovering secrets that were meant to stay buried.)

Next, type in your command: ALIGN THE RESONATORS ON THE PEDESTALS... But be careful—one wrong choice at the Nexus controls and you might find yourself scattered across infinite dimensions!

"The truth lies buried beneath the lunar

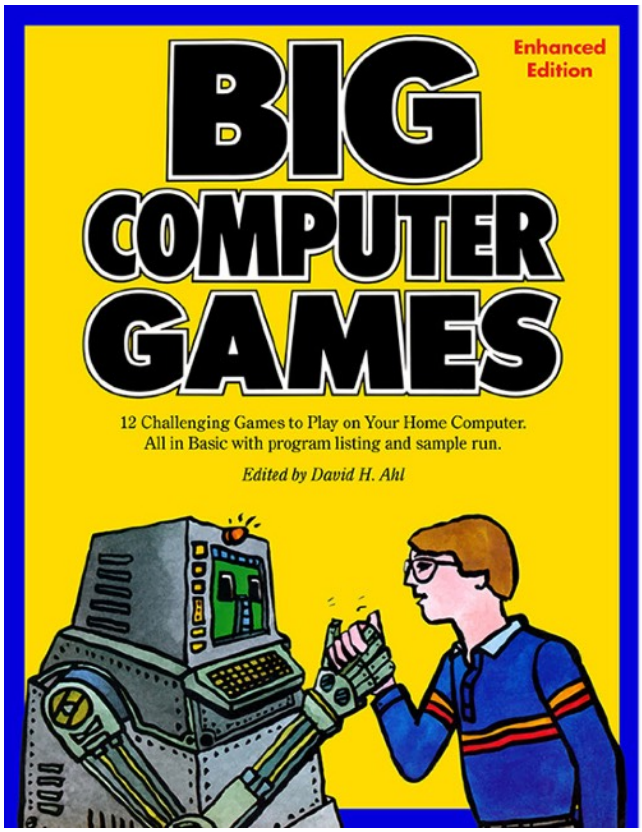
There's no telling what you'll discover in MOON BASE ALPHA because, like all of A.P.P.L.E.'s interactive fiction, MOON BASE ALPHA is designed so that whatever *you* choose to do makes the next thing happen. And you won't run out of mysteries to solve, either. The abandoned base is so vast, your investigation can last for weeks or even months.

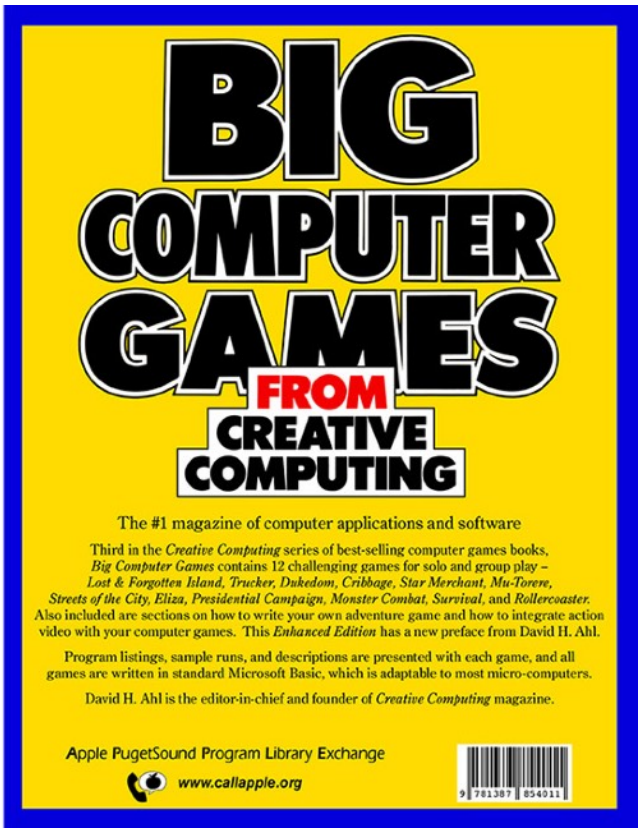
So if you want the closest thing on a disk to exploring the dark side of the moon, get MOON BASE ALPHA.* But brace yourself...what happened to those 87 crew members will haunt you!

A.P.P.L.E.

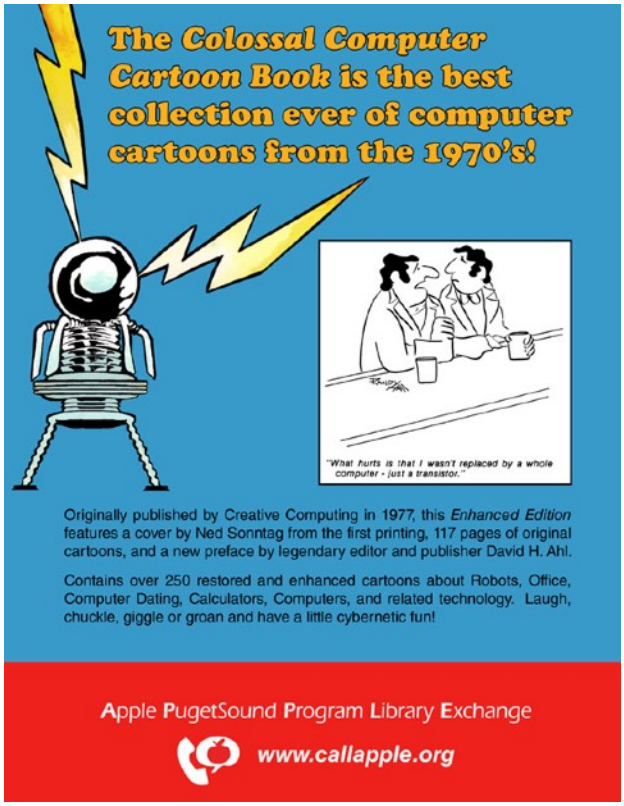
www.callapple.org

*It's compatible with most Z-machine interpreters.





Do you have all your Ahl ?



SAB 4K Pilot — Part 8

by Forrest Lowe / A.P.P.L.E. Staff

PART 8:
The USE (U:), END (E:) & REM (R:) command processors.

We now get to some small, powerful commands. U: is the subroutine call of PILOT, and E: is the subroutine return. R: is the remark feature, not exactly a command, but implemented as one.

The USE command's function is to change the program flow in a way you can return from, like a function call or a subroutine call. The address of the PILOT line following this statement is saved in the return table, and used as an address. The return index pointer from $E2, a value from $0 to $E, is retrieved and the next PILOT line pointer is stored in the return table at that offset. The index is incremented twice and tested to see if it is at $10. If not, it is saved in $E2 and the interpreter jumps to the alt-entry of the jump routine to actually implement the jump. If the next index value is $10, the routine sends a "NESTING TOO DEEP" message and exits the interpreter.

This routine occupies $1482 through $14AA. There are no nested subroutines.

As written, the last entry in the routine address table cannot be used. Since the Return Index points at the next available slot, it starts at zero, pointing at table entry zero, the first slot. When the routine is entered with the index value at $E, at the point that it prepares to go to the Jump routine to find its target, it has saved the return value in slot $E, and then updates the Return Index, and tests to see if it is $10. When it is, the code branches to the "NESTING TOO DEEP" message and exits the interpreter. This means the target routine is never run for index $E. By moving the test to the start of the routine, and testing for the return table being full before using it save the return address.

Only the three bytes following the YN call can be saved easily.

The E: command's function is to return the program flow to the line after the USE statement that got us here.

The END command is much like the return command in other languages, with the exception that it ends the program if no subroutine is active, stack underflow. It loads the next line's address and returns to the interpreter loop or exits the interpreter as appropriate. There is a problem in that the active line number is not saved, and the line number of the subroutine is not updated to the return address' line number, so after a call and return, certain error messages will report the wrong line number as in error.

This routine occupies $14AA through $14C5.

Here too, only the three bytes following the YN call can be saved easily, but by reusing the exit call in the U: code we can save 3 more bytes.

The R: processor does nothing. It holds a comment in the PILOT source. The code here consists of a single return statement.

This routine occupies $14C6 through $14C6. It makes no calls or anything. By changing the address called to any other return in the interpreter, we can achieve 100% code reduction.

The mentioned saves and reusing a jump result in saving 10 of the original 84 bytes

Part 8 Original Listing

```
: A S M
                1    MSGOUT   MAC              ; SEND MESSAGE TO SCREEN
                2             LDA   #>]1
                3             LDX   #<]1
                4             JSR   MSGWRITE
                5             <<<
                6    ************************************
                7    * PILOT EMULATOR
                8    * AUTHOR JAMES A. BAKER
                9    * SOURCE RECOVERED BY FORREST LOWE
                10   * JULY 2025 - JANUARY 2026
                11   ************************************
                12   *
                13   * U: USE PROCESSOR
                14   *
                15   ****************************************
                16   *
                17   *  SYSTEM VARIABLES
                18   *
                19   ****************************************
                20   *
                21   *  PROGRAM VARIABLES
                22   *
                23   LINECOPY EQU   $D0        ; COPY LINE FROM ADDR
                24   IRETURN  EQU   $E2        ; INDEX TO CUR RETURN ELEMENT
                25   *
                26   TRETURNA EQU   $18D6      ; RETURN TABLE
                27   ****************************************
                28   *
                29   *        IN PROGRAM LOCATIONS
                30   *
                31   MOD_TEST EQU   $11F8      ; GET NXT CHAR PROC YN, RETURN :
                32   ALT_J    EQU   $1262      ; JUMP SECND ENTRY
                33   MSGWRITE EQU   $179D      ; PRINT MSG ENDING $00
                34   MTOODEEP EQU   $17DC      ;NESTING TOO DEEP
                35   JUMPSHIP EQU   $18CF      ; EXIT JOURNEY
                36
                37   ****************************************
                38
                39            ORG   $1482
                40
                41   ****************************************
1482: 20 F8 11  42   CMD_U    JSR   MOD_TEST   ; GET NEXT CHAR AND DO YN
1485: B0 01     43            BCS   H1488      ; RETURN POINTING TO :
1487: 60        44            RTS
                45   *            USE ALT ENTRY
1488: A4 E2     46   H1488    LDY   IRETURN    ; GET RETURN INDEX
148A: A5 D0     47            LDA   LINECOPY   ; GET START PGM LINE NUM.
148C: 99 D6 18  48            STA   TRETURNA,Y ; SAVE LINE IN RETURN TABLE
148F: C8        49            INY
1490: A5 D1     50            LDA   LINECOPY+1 ; REST OF LINE NUM.
1492: 99 D6 18  51            STA   TRETURNA,Y ; SAVE LINE IN RETURN TABLE
1495: C8        52            INY
1496: 98        53            TYA
1497: 29 10     54            AND   #$10       ; LIMIT DEPTH TO 16
1499: D0 05     55            BNE   TOO_DEEP
149B: 84 E2     56            STY   IRETURN    ; SAVE RETURN INDEX
149D: 4C 62 12  57            JMP   ALT_J      ; NOW GO TO JUMP ROUTINE
                58   ************************
                59   TOO_DEEP MSGOUT MTOODEEP  ; NESTING TOO DEEP
14A0: A9 17     59            LDA   #>MTOODEEP
```

```
14A2: A2 DC     59            LDX   #<MTOODEEP
14A4: 20 9D 17  59            JSR   MSGWRITE
                59            <<<
                60   *TOO_DEEP LDA #$17
                61   * LDX #$DC
                62   * JSR H179D
14A7: 4C CF 18  63            JMP   JUMPSHIP
                64   ************************************
                65   *
                66   * E: END PROCESSOR
                67   *
                68   ************************************
                69
                70            ORG   $14AA
                71
14AA: 20 F8 11  72   CMD_E    JSR   MOD_TEST
14AD: B0 01     73            BCS   H14B0
14AF: 60        74            RTS
14B0: A4 E2     75   H14B0    LDY   IRETURN    ; GET NEXT RETURN INDEX
14B2: 88        76            DEY             ; CHANGE TO CURRENT RI
14B3: 30 0E     77            BMI   FAIL_E     ; IF NEGATIVE INDEX, END PGM
14B5: B9 D6 18  78            LDA   TRETURNA,Y ; NOW COPY SELECTED SLOT
14B8: 85 D1     79            STA   LINECOPY+1 ;  TO $D0/D1
14BA: 88        80            DEY
14BB: B9 D6 18  81            LDA   TRETURNA,Y
14BE: 85 D0     82            STA   LINECOPY
14C0: 84 E2     83            STY   IRETURN    ; SAVE THE NEW NEXT
14C2: 60        84            RTS             ; AND RETURN TO NEXT AFTER CALL
14C3: 4C CF 18  85   FAIL_E   JMP   JUMPSHIP
                86   ************************************
                87   *
                88   * R: REMARK PROCESSOR
                89   *
                90   ************************************
                91
                92            ORG   $14C6
                93
14C6: 60        94   CMD_R    RTS
                95   *            FOLLOWED BY L:


--End assembly, 69 bytes, Errors: 0


Symbol table - alphabetical order:

   ALT_J   =$1262   ?  CMD_E   =$14AA   ?  CMD_R   =$14C6   ?  CMD_U   =$1482
   FAIL_E  =$14C3      H1488   =$1488      H14B0   =$14B0      IRETURN =$E2
   JUMPSHIP=$18CF      LINECOPY=$D0        MOD_TEST=$11F8   MD MSGOUT  =$8000
   MSGWRITE=$179D      MTOODEEP=$17DC      TOO_DEEP=$14A0      TRETURNA=$18D6



Symbol table - numerical order:

   LINECOPY=$D0        IRETURN =$E2     MD MSGOUT  =$8000      MOD_TEST=$11F8
   ALT_J   =$1262   ?  CMD_U   =$1482      H1488   =$1488      TOO_DEEP=$14A0
?  CMD_E   =$14AA      H14B0   =$14B0      FAIL_E  =$14C3   ?  CMD_R   =$14C6
   MSGWRITE=$179D      MTOODEEP=$17DC      JUMPSHIP=$18CF      TRETURNA=$18D6
```

Part 8 New Listing

```
: A S M
                1    MSGOUT   MAC              ; SEND MESSAGE TO SCREEN
                2             LDA   #>]1
                3             LDX   #<]1
                4             JSR   MSGWRITE
                5             <<<
                6    ************************************
                7    * PILOT EMULATOR
                8    * AUTHOR JAMES A. BAKER
                9    * SOURCE RECOVERED BY FORREST LOWE
                10   * JULY 2025 - JANUARY 2026
                11   ************************************
                12   *
                13   * U: USE PROCESSOR
                14   *
                15   ****************************************
                16   *
                17   *  SYSTEM VARIABLES
                18   *
                19   ****************************************
                20   *
                21   *  PROGRAM VARIABLES
                22   *
                23   LINECOPY EQU   $D0        ; COPY LINE FROM ADDR
                24   IRETURN  EQU   $E2        ; INDEX TO CUR RETURN ELEMENT
                25   *
                26   TRETURNA EQU   $18D6      ; RETURN TABLE
                27   ****************************************
                28   *
                29   *        IN PROGRAM LOCATIONS
                30   *
                31   MOD_TEST EQU   $11F8      ; GET NXT CHAR PROC YN, RETURN :
                32   ALT_J    EQU   $1262      ; JUMP SECND ENTRY
                33   MSGWRITE EQU   $179D      ; PRINT MSG ENDING $00
                34   MTOODEEP EQU   $17DC      ;NESTING TOO DEEP
                35   JUMPSHIP EQU   $18CF      ; EXIT JOURNEY
                36
                37   ****************************************
                38
                39            ORG   $1482
                40
                41   ****************************************
1482: 20 F8 11  42   CMD_U    JSR   MOD_TEST   ; GET NEXT CHAR AND DO YN
1485: EA        43            NOP
1486: EA        44            NOP
1487: EA        45            NOP
1488: A4 E2     46            LDY   IRETURN    ; GET RETURN INDEX
148A: A5 D0     47            LDA   LINECOPY   ; GET START PGM LINE NUM.
148C: 99 D6 18  48            STA   TRETURNA,Y ; SAVE LINE IN RETURN TABLE
148F: C8        49            INY
1490: A5 D1     50            LDA   LINECOPY+1 ; REST OF LINE NUM.
1492: 99 D6 18  51            STA   TRETURNA,Y ; SAVE LINE IN RETURN TABLE
1495: C8        52            INY
1496: 98        53            TYA
1497: 29 10     54            AND   #$10       ; LIMIT DEPTH TO 16
1499: D0 05     55            BNE   TOO_DEEP
149B: 84 E2     56            STY   IRETURN    ; SAVE RETURN INDEX
149D: 4C 62 12  57            JMP   ALT_J      ; NOW GO TO JUMP ROUTINE
                58   ************************
                59   TOO_DEEP MSGOUT MTOODEEP  ; NESTING TOO DEEP
14A0: A9 17     59            LDA   #>MTOODEEP
```

```
14A2: A2 DC     59            LDX   #<MTOODEEP
14A4: 20 9D 17  59            JSR   MSGWRITE
                59            <<<
14A7: 4C CF 18  60   FAIL_E   JMP   JUMPSHIP
                61   ************************************
                62   *
                63   * E: END PROCESSOR
                64   *
                65   ************************************
14AA: 20 F8 11  66   CMD_E    JSR   MOD_TEST
14AD: EA        67            NOP
14AE: EA        68            NOP
14AF: EA        69            NOP
14B0: A4 E2     70            LDY   IRETURN    ; GET NEXT RETURN INDEX
14B2: 88        71            DEY             ; CHANGE TO CURRENT RI
14B3: 30 F2     72            BMI   FAIL_E     ; IF NEGATIVE INDEX, END PGM
14B5: B9 D6 18  73            LDA   TRETURNA,Y ; NOW COPY SELECTED SLOT
14B8: 85 D1     74            STA   LINECOPY+1 ;  TO $D0/D1
14BA: 88        75            DEY
14BB: B9 D6 18  76            LDA   TRETURNA,Y
14BE: 85 D0     77            STA   LINECOPY
14C0: 84 E2     78            STY   IRETURN    ; SAVE THE NEW NEXT
14C2: 60        79   C_E_RTS  RTS             ; AND RETURN TO NEXT AFTER CALL
14C3: EA        80            NOP
14C4: EA        81            NOP
14C5: EA        82            NOP
                83   ************************************
                84   *
                85   * R: REMARK PROCESSOR
                86   *
                87   ************************************
                88   CMD_R    EQU   C_E_RTS
14C6: EA        89            NOP
                90   *            FOLLOWED BY L:


--End assembly, 69 bytes, Errors: 0


Symbol table - alphabetical order:

   ALT_J   =$1262   ?  CMD_E   =$14AA   ?  CMD_R   =$14C2   ?  CMD_U   =$1482
   C_E_RTS =$14C2      FAIL_E  =$14A7      IRETURN =$E2        JUMPSHIP=$18CF
   LINECOPY=$D0        MOD_TEST=$11F8   MD MSGOUT  =$8000      MSGWRITE=$179D
   MTOODEEP=$17DC      TOO_DEEP=$14A0      TRETURNA=$18D6


Symbol table - numerical order:

   LINECOPY=$D0        IRETURN =$E2     MD MSGOUT  =$8000      MOD_TEST=$11F8
   ALT_J   =$1262   ?  CMD_U   =$1482      TOO_DEEP=$14A0      FAIL_E  =$14A7
?  CMD_E   =$14AA      C_E_RTS =$14C2   ?  CMD_R   =$14C2      MSGWRITE=$179D
   MTOODEEP=$17DC      JUMPSHIP=$18CF      TRETURNA=$18D6
```

THE TOWER

AN INTERACTIVE FICTION ADVENTURE

"Thirty years ago, something landed on the roof.
It was never seen again."

EXPLORE over 95 locations across 4 mysterious floors
DISCOVER hidden passages and ancient secrets
ESCAPE to Space Station Alpha... if you survive

Original Design: Bill Martens (1985) • Publisher: A.P.P.L.E.
INTERACTIVE FICTION • Requires Z-Machine Interpreter

GSSquared: Apple Series Emulator 0.8.0 Released

by Jawaid Bazyar / A.P.P.L.E. Staff

I am pleased to announce the release of **GSSquared v0.8.0**.

GSSquared is an Apple II series computer emulator.

Supported platforms

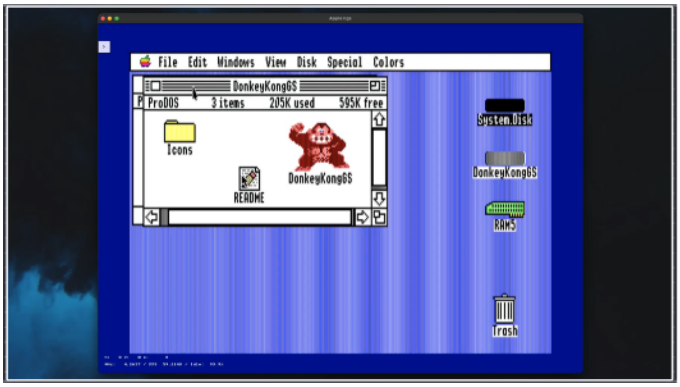
- Apple][
- Apple][Plus
- Apple //e
- Apple //e Enhanced
- Apple //e Enhanced with 65816 CPU and Super Hires Video
- Apple IIGS ROM 01

Pre-built binaries

- **macOS** (Intel and Apple Silicon)
- **Windows 10+**
- **Linux AppImage** (should run on many Linux distros; tested on Ubuntu 22)

This release brings full **3.5″ floppy support** on the Apple IIgs, a much-improved **Control Panel** for managing disks, and better handling when you quit with unsaved disk changes.

Features

- **Apple IIgs 3.5″ floppy drives** now support **WOZ disk images** for reading and writing, including many copy-protected titles. Copy-protected titles include Alien Mind, and Tomahawk.
- **5.25″ and 3.5″ floppy emulation** has been significantly improved. Copy-protected WOZ images that previously failed should now work, and changes to mounted disks can be saved back to the original image file when you unmount.
- The **Control Panel** has new, clearer drive icons for 3.5″ AppleDisk drives and hard drives. Icons match the machine you are emulating — Disk II on early Apple II models, AppleDisk on the //e and IIgs.
- **BazFast**, the built-in SmartPort hard drive, can now mount **multiple disk images at once**. Drag a `.pmap` file onto a BazFast icon to load a whole set of volumes in one step — handy for multi-disk setups like a hard-drive collection with separate ProDOS volumes.
- When you **quit or power off**, GSSquared now properly asks whether to save any disks that have unsaved changes.
- You can launch GSSquared directly into a specific machine from the command line: `-p PLATFORM` (for example, `-p iigs`). Closing the window will then exit the emulator automatically.
- 

- **Acc Floppy disk timing and head movement are more accurate, which helps games and copy-protection schemes that depend on precise disk behavior.**
- Mounting a disk image on the wrong kind of drive (for example, a 140K floppy image on a hard drive) now shows a clear error instead of failing silently. uracy

Bug Fixes

- The debugger open/close key is now **F10**, so it no longer conflicts with keys used inside Apple II software.
- Fixed a Windows build issue.

Internals

- The floppy and IWM disk code has been reorganized and rewritten as a foundation for future storage improvements.
- Documentation for disks, WOZ images, and BazFast has been updated.

https://github.com/jawaidbazyar2/gssquared/releases/tag/v0.8.0

Genius 1-3 for the Apple II Release 20 Downloads

by A.P.P.L.E. Staff

Apple II Version

GENIUS1 - INTO THE TOY WAREHOUSES - https://www.callapple.org/product/genius1-into-the-toy-warehouses/

GENIUS2 - INTO THE TOY CAVES - https://www.callapple.org/product/genius2-into-the-toy-caves/

GENIUS3 - INTO THE TOY PLANETS - https://www.callapple.org/product/genius3-into-the-toy-planets/

Atari Version

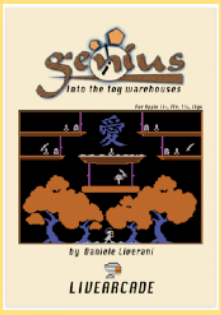
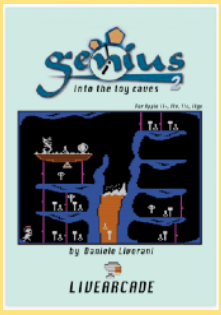
(GENIUS 1-3 RELEASE 1) : https://www.dropbox.com/scl/fi/8712ggneyjf8wxjk0nznz/GeniusEp1-2-3-Atari-8Bit-Version01.zip?rlkey=55uj2czpv1fnx9bgcs1nimkl5&e=1

Commodore 64 Version

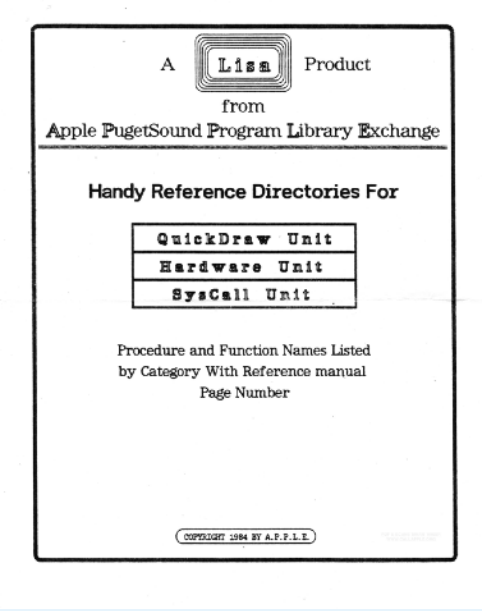
GENIUS1 – INTO THE TOY WAREHOUSES – https://csdb.dk/release/?id=198917
GENIUS2 – INTO THE TOY CAVES – https://csdb.dk/release/?id=152393
GENIUS3 – INTO THE TOY PLANETS – https://csdb.dk/release/?id=160776

Commodore Plus 4 Version

GENIUS1 – INTO THE TOY WAREHOUSES – https://plus4world.powweb.com/software/Genius
GENIUS2 – INTO THE TOY CAVES – https://plus4world.powweb.com/software/Genius_2
GENIUS3 – INTO THE TOY PLANETS – https://plus4world.powweb.com/software/Genius_3

GENIUS1	GENIUS2	GENIUS3
INTO THE TOY WAREHOUSES	INTO THE TOY CAVES	INTO THE TOY PLANETS
		

Handy Reference Directory-Quickdraw Demo for LISA


A Lisa Product
from
Apple PugetSound Program Library Exchange

Handy Reference Directories For

QuickDraw Unit
Hardware Unit
SysCall Unit

Procedure and Function Names Listed
by Category With Reference manual
Page Number

COPYRIGHT 1984 BY A.P.P.L.E.

The LISA Quickdraw Demo Workshop, produced by A.P.P.L.E. courtesy of David Redhed, was a reference to the LISA Quickdraw, Hardware and SysCall PASCAL related routines.

The original five page document along with the handy accompanying 3.5 inch floppy disk gave users an easy way to manage the user accessible routines of the LISA.

After receiving this from one of our many archives, we have decided to once again make this product available in the A.P.P.L.E. Store.

It includes a newly made PDF manual which includes some updates and the original 3.5 inch floppy disk image for the LISA and MAC XL computers in digital disk image format.

Introducing an insanely true replica of the Apple-1 manuals. Re-typesetted from scratch with unrivaled accuracy.

This is a nerd story, level 1,000: In April of 2021, German retro computer fan Armin Hierstetter was researching the exact measurements of the Apple-1 printed circuit board to build a case for it. During the research, Armin stumbled upon pictures of the original Apple-1 manual and soon found out that cheap laser printed copies of low-quality scans sold for 100 USD and more on eBay.

Because he did not want to pay that much for such a lousy job, he tracked down the scans used, enhanced them, and had the result printed by a professional printing company.

But although this enhanced version was saddle-stitched, printed on great paper, and had all the "dirt" removed from the scans, the results were rather poor.

A true typeset job!

Then Armin had an idea: What if he redesigned the manuals from scratch, using modern typesetting methods? Great idea, no? "If I had known before what kind of crazy mission laid in front of me", Armin said in an interview, "I wouldn't have touched this totally insane project with a ten-foot pole."

But Armin did what a man had to do and embarked his journey into typesetting madness. Several steps were necessary to implement the project, researching all typefaces being the first of them.

It was not particularly helpful that the main copy of the "Operation Manual" had been written on a typewriter and later copied into the design. In particular, the proportional font of the (presumably) IBM Electric typewriter used was not easy to find. But thanks to the great users of typography.guru, all character sets were gradually identified.

Now: Simply knowing which typefaces to use was only a fraction of the job that lay ahead. For the replica to be really as true to the original as humanly possible, the position of each individual letter had to be accurate to the fraction of a millimeter. In other words: Each and every character had to be positioned manually. Word by word, paragraph by paragraph and page by page. This… took… ages!

If you want to get a feeling about just how painstaking this job was, there is a video on youtube: tinyurl.com/apple-1

The next challenge: Recreating the schematics.

If you thought that pushing around characters for tens of hours was a crazy idea, you ain't seen nothing yet, because the worst was yet to come: re-drawing the schematics.

You need to understand that most of the lettering was made with a template specifically for technical drawings (or maybe a leroy set). This means: Ronald G. Wayne, the third founder of Apple (who gave back his 10 percent stake in one of today's most valuable companies in the world 12 days after Apple incorporated in 1976) and responsible for the schematics, drew every character by hand, which made finding a typeface particularly difficult. Again. Fortunately, the magic folk at typography.guru came to a rescue one more time…

Professional printing and binding. Including the fold-out schematics.

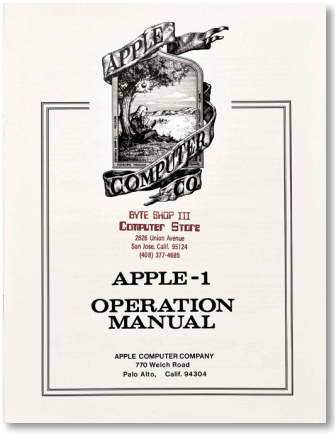
Now it was time to tackle the last hurdle: So far, the manuals only existed as some bits and bytes within an InDesign file. A printing company had to be found that could do a tiny print run at a reasonable price and be able to produce the 6-page gatefold section featuring the schematics. What's more, the job demanded some high-quality paper, specifically 150gsm "Munken Pure", a very decent paper with a slight yellowish tint to resemble the aged look of the original item. Not many companies will take such a tiny project on board, especially since the original US letter format is not standard in DIN-dominated countries like German-Land.

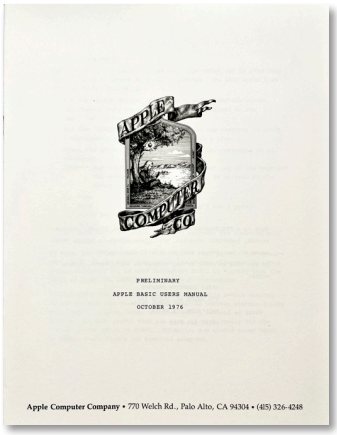
But, after some research, a fairly huge printing company embraced the challenge and printed an eye-watering piece of beauty. Actually, four pieces of beauty, because: Armin not only recreated the Operation Manual, but also the "Preliminary Basic Users Manual", the "Cassette Interface Manual" and a replica of the very first Apple-1 ad (which was the design inspiration of this very ad in case the thought: "This looks familiar!").

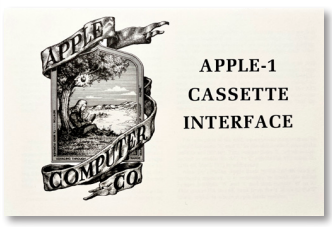
There is "one more thing"…

As you may know: The very first 50 Apple-1 computers have been ordered by Paul Terrell, founder of the Byte Shop. The shop stamped the Operation Manual of every Apple-1 sold with their stamp. Armin concluded that the recreations would not be complete without the option to include the stamp. So Armin tracked down Paul Terrell, who personally gave permission to recreate the stamp – what a cool guy.

Which concludes this nerd story, level 1,000. If you like this crazy little project and want to get your very own set of the manual replicas, you can do so at apple-1-manuals.com. Armin dashed out a special offer for all CALL-A.P.P.L.E. readers. **The promo code CALLAPPLE will give you a 15 percent discount during checkout.**







YES,

I want a set of the Apple-1 manual replicas, including the three manuals and the first full-page ad.

Fill out the form. Or, more seriously, head over to apple-1-manuals.com, to get your set!

Name

Street

City

Country

apple-1-manuals.com • Use code CALLAPPLE at checkout for a 15 % discount!

BlueSCSI

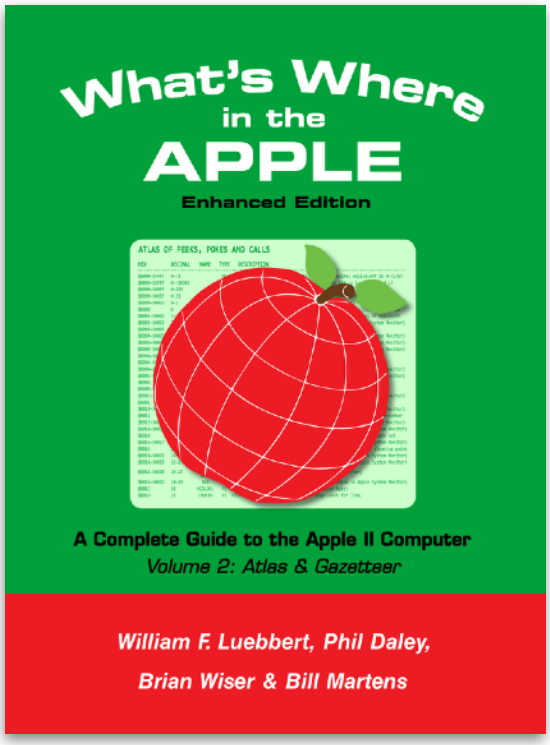
Works on Macintosh Plus, Macintosh SE/30, Classic, Classic II, Mac IIci, IIcx, IIx, Quadra 650/700, Color Classic, LC I/II/III, LC 475, LC 575, 6100, NeXT Cube 040, and 7500/100 PPC. Others machines in testing include Amiga, Apple IIe, Apple IIGS and others.

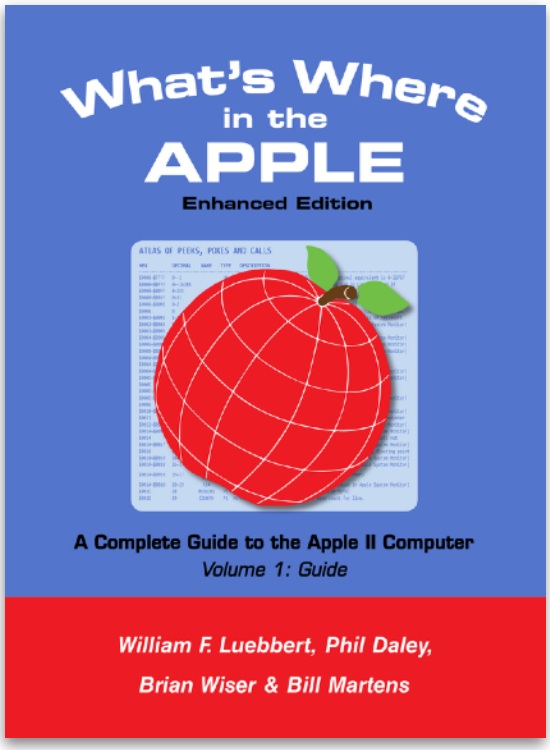
Kits and Assembled units available from:

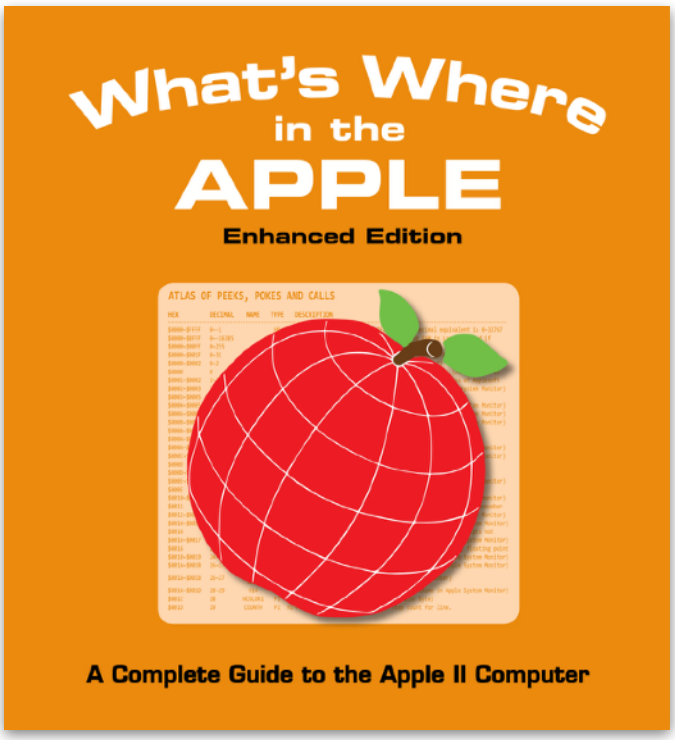
bluescsi.com

That was **What's Where in the Apple** in 1985. We could have stopped there, but we didn't! First updated in 2016 as an "Enhanced Edition" with major additions and corrections, the 2nd Enhanced Revisions are here! They are updated with more information than you can shake a stick at.

This famous book now contains the most comprehensive description of firmware and hardware ever published for the Apple II family. A new section with Guide, Atlas and Gazetteer now provides Apple IIe and IIc specific information. Thoroughly proofread over two years by the best experts in the hobby.





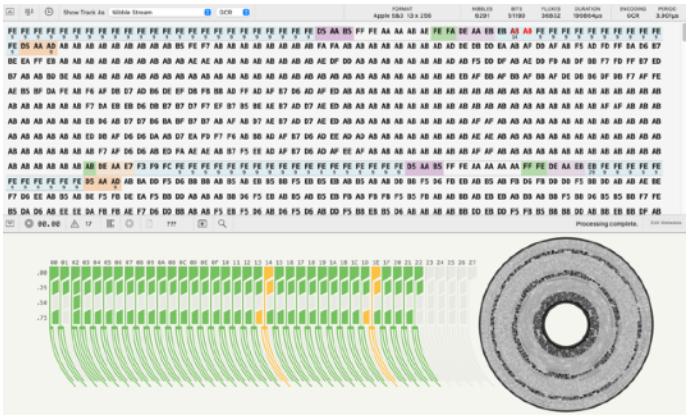


- Names and locations of Monitor, DOS, Integer BASIC and Applesoft routines and what they're used for.
- The numerical Atlas and alphabetical Gazetteer guide you to over 2,700 memory locations of PEEKs, POKEs, and CALLs in DOS and ProDOS.
- Moving easily between BASIC and Machine Language
- Easier, better, and faster software writing.
- Concepts and programming techniques for mastering Apple II inner workings and hidden mechanisms.

https://www.callapple.org/books

Applesauce FDC: 2.06.2 Software Update

by A.P.P.L.E. Staff / John Morris



Enhancements and fixes for 2.06.2

- Some improvements to data recovery for IBM-sectored disks.
- Added support for loading IMA and VFD image files.
- Added file system support for 40-track Tandy Coco RS-DOS disks. Only 35-track disks were supported previously.
- Allow loading of D88 images that have extra long comments that violate the format specification.
- Prevent a crash that could occur when loading DC42 images that have an invalid format specified.
- Added the ability to double-click a drive when picking which drive you want to use. (David Schmidt)
- Finalized support for the upcoming xpand-o-rama hardware release. There is updated firmware for both the Applesauce and Applesauce+.

Enhancements and fixes for 2.06.1

- Fixed a potential data loss when editing the metadata of an A2R that contains multiple RWCP chunks.
- Allow loading of SCP images that only contain a single track.
- MFI image export will now save all written tracks, not just the set that the file system defines. This change will primarily allow manufacturing marks to be included in exported images.

Enhancements and fixes for 2.06

- Added support for Hewlett-Packard 9885/9895 disk format, HP file system, as well as the HPI image format.
- More support for Amstrad CPC disks.
- Additional CP/M definitions for NEC PC-8001, DEC Rainbow, and Amstrad.
- Improvements to detecting clocked data in unusual formats.
- More tweaks to the NorthStar DOS support.
- Fix the loading of IMD files that were imaged at 300kbps.
- Wider support for various M2FM-based encoding.
- Eased up validity checks for exporting disks to Atari ATX image format.

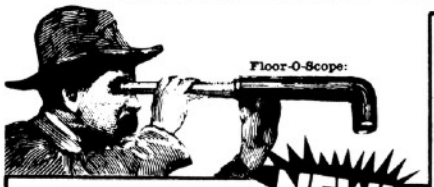
You can download the latest version of the software by doing the automatic update through the client software or from the Applesauce FDC Software page at:
https://applesaucefdc.com/software/

Coming in the Next Issue

Next month we have the latest breakdown of the game Journey from 1979 and the efforts to decompile and reproduce the SAB 4k Pilot Interpreter, Latest games and hardware and other items for the Mac and Apple II Series

DON'T BLOW YOUR BUCKS ON Locked-Up Software!

Beagle Bros Apple Utilities are Listable, Backup-able Customizable and Compatible with Normal Apple DOS.



NEW!

Apple Mechanic

Shape Writer/Byte-Zap Utility

by Bert Kersey

Another hot multiple-utility disk—Nine useful, listable, copyable & customizable programs—

SHAPE EDITOR: Put professional hi-res animation in your programs. Keyboard-draw any shape & let your Apple write a shape table & store it on disk. Design large/small custom typefaces too, with special characters. Many fonts on disk. LIST-able demos show how to use shapes to animate games, displays, and CHARTS & GRAPHS. A valuable time-saving utility/learning tool.

BYTE ZAP: A MUST utility. Rewrite any byte on a disk by loading a sector onto the screen for inspection. HEX/DECIMAL/ASCII display optional. Examine bytes via cursor control; enter hex, dec or ascii to change. Create illegal filenames, restore deleted files, change greeting program name, repair/protect disks, change DOS, examine program files. Clear illustrated instructions show how disk data is stored and how to access it.

MORE: A disk PACKED with useful music, text & hi-res tricks FOR USE IN YOUR PROGRAMS. Demo-writer, hi-res utilities and excellent, educational, entertaining documentation.

ONLY $0.00

- ☐ Apple Mechanic disk (48K min.)
- ☐ Beagle Bros Tip Book #5 (60 pgs.)
- ☐ Peeks, Pokes & Pointers Chart

Tip Disk #1

by Bert Kersey

100 programs from Beagle Bros' Tip Books 1, 2, 3 & 4— Dozens of tricks to make your Apple do things it's never done! All 100 programs are listable, copyable and changeable; each teaches another fascinating Apple programming technique.

ONLY $0.00

- ☐ Tip Disk #1 on disk (32K or 48K)
- ☐ Peeks, Pokes & Pointers Chart

(Note: No tip book with this disk)

Alpha Plot

Hi-Res Graphics/Text Utility

by Bert Kersey & Jack Cassidy

Here are a few of Alpha Plot's useful features. Compare with others on the market—

HI-RES DRAWING: Create hi-res pictures & charts with text, on both pages; all APPENDABLE TO YOUR PROGRAMS. Optional Xdraw cursor (see lines before drawing). Mix colors & Reverse (background opposite). Circles, Boxes, Ellipses; filled or outlined. COMPRESS HI-RES TO 1/3 DISK SPACE. Relocate any portion of an image anywhere on either page. Superimpose too & convert hi-res to lo-res for colorful abstracts!

HI-RES TEXT: Beautiful upper/lower case with descenders (no hardware required). Color & reverse characters positionable anywhere (no tab limits). Professional-looking PROPORTIONAL SPACING; adjustable character height & letter spacing. Multi-directional typing for graphs!

ONLY $0.00

- ☐ Alpha Plot on Disk (48K min.)
- ☐ Beagle Bros Apple Tip Book #4
- ☐ Peeks, Pokes & Pointers Chart

DOS BOSS

DISK COMMAND EDITOR

by Bert Kersey & Jack Cassidy

A classic Apple utility you will ENJOY! Rename DOS commands (CATALOG can be "Cat", etc.). PROTECT PROGRAMS; any unauthorized save-attempt produces a "Not Copyable" message. Also LIST-PREVENTION & 1-key program-run from catalog. Custom catalogs: Change Disk Volume message to your title; Omit/alter file codes. Re-write error messages: Syntax Error can be "Oops!!" or anything! Fascinating documentation included; Hours of good Apple reading!

Dos Boss's change features may be appended to your programs. Anyone using your disks (booted or not) formats their DOS as YOU designed it.

ONLY $0.00

- ☐ Dos Boss on Disk (32K/48K min.)
- ☐ Beagle Bros Apple Tip Book #2
- ☐ Peeks, Pokes & Pointers Chart

Utility City

21 Useful Utilities on One Disk

by Bert Kersey

LIST FORMATTER makes properly-spaced and indented listings with page breaks; each statement on new line, if-thens & loops called out; a great de-bugger! MULTI-COLUMN CATALOG in any page-width to printer or screen. Auto-post Run-Number & last-used Date in programs. Put INVISIBLE working commands in listings. Access program lines in memory for repair & illegal alteration. Alphabetize & store info on disk. Run any program while another stays intact. Renumber to 65535. Save inverse, trick and INVISIBLE FILE NAMES. Convert dec to hex & binary, or INT to FP. Append programs. Dump text screen to printer... 21 LISTABLE PROGRAMS TOTAL!

21 PROGRAMS FREE

- ☐ Utility City on disk (48K min.)
- ☐ Beagle Bros Apple Tip Book #3
- ☐ Peeks, Pokes & Pointers Chart

GOTO your Apple User Group

www.callapple.org

Apple Pugetsound Program Library Exchange (A.P.P.L.E.) founded in 1978, is a resource for retro and modern Apple news, software, and new magazines and books. Check out our Member Benefits!

Bonuses With Every Disk!

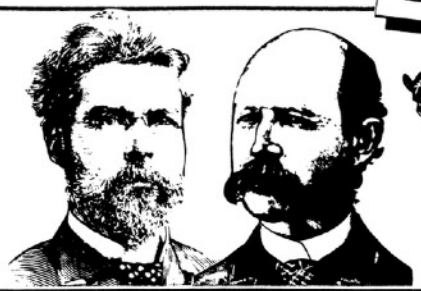
Poke your Apple all night long with this free reference poster! The most useable **PEEKS, POKES, POINTERS** and **CALLS,** scrounged up from every source imaginable!

Apple Tip Books too—

Each disk comes with a Gold Mine of valuable Apple information and hours of entertaining reading matter, including dozens of tips and keyboard experiments on all subjects— DOS, Copy Protection, Graphics, Shape Tables, Hardware and More. Sample programs too, such as "Programming the Reset Key" and "Copy Stoppers". Each disk comes with its own unique book.

24 Hour TOLL FREE Web Site

beagle.applearchives.com



Beagle Bros™ MICRO SOFTWARE

http://beagle.applearchives.com

"APPLE" is a registered trademark of Apple Computer, Inc.
Our "BEAGLE BROS" site is the only Officially Approved one.

OR ORDER BY CARRIER PIGEON

RUSH! The disk packages checked below Plus the Tip Book & PEEKS/POKES CHART.

☐ **Alpha Plot** ☐ **Dos Boss** ☐ **Tip Disk**
☐ **Apple Mechanic** ☐ **Utility City**

(Use this coupon or separate sheet.)

Apple II File and Utility Programs

by A.P.P.L.E. Staff

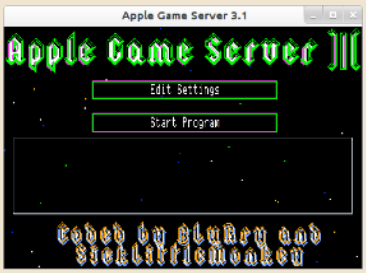
ADTPro 2.1.0

https://www.adtpro.com

ADTPro, which allows users to image Apple II and Apple III disks, using either a serial connection or an ethernet connection. The program also allows boot strapping of a machine via those same methodologies:

Apple Game Server 3.1

https://sourceforge.net/projects/a2gameserver



Apple Game Server allows users to boot their Apple II computers without floppy disks. The only requirement for this is a functional serial connection to a computer running the Apple Game Server software. Check out the Apple Game Server file repository:

HCS Software

https://www.callapple.org/HCS

Highlands Computer Services was founded by Butch Greathouse and Garry Rheinhardt in Renton, Wa. Two doors down from the founder of Synergistic Software, Robert C. Clardy, Butch and A.P.P.L.E.'s Joff Morgan spent a lot of time with Bob in the early days figuring out what the Apple II computer could do and creating software

Virtual II 13.0.1

https://www.virtualii.com

The Virtual II emulator requires users to have MacOS 10.13 ("High Sierra") as a minimum. Older versions are also available for download from the Virtual II Website. For users who have previously purchased Virtual II, the update is available from the website

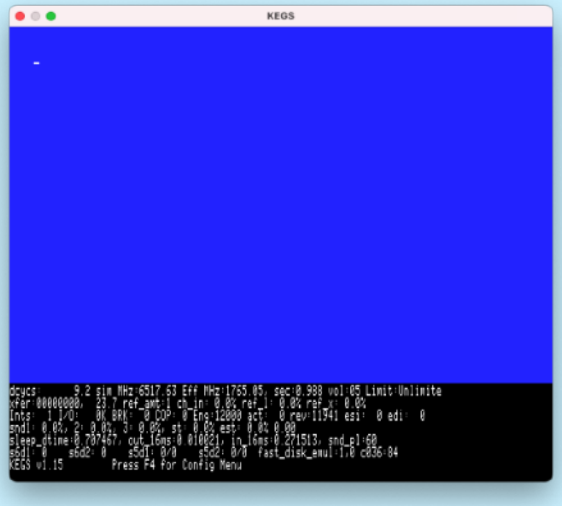
KEGS Apple II GS Emulator

https://kegs.sourceforge.net/

KEGS emulates an Apple IIgs accurately at between 500MHz and 950MHz on pretty much any Unix/Linux system, macOS X, or Windows 10+.

The current version of the emulator is 1.38.

itty-bitty-tty 1.06



itty-bitty-vtty is a vt100 emulator for the Apple IIgs developed by Kelvin Sherlock which provides an operating system free, accurate vt100 emulation. itty-bitty-vtty uses the modem port at 9600 baud and the standard 8-N-1 transfer settings, which is also the MAME Emulator defaults for those who wish to use it in coordination with the MAME Emulator.

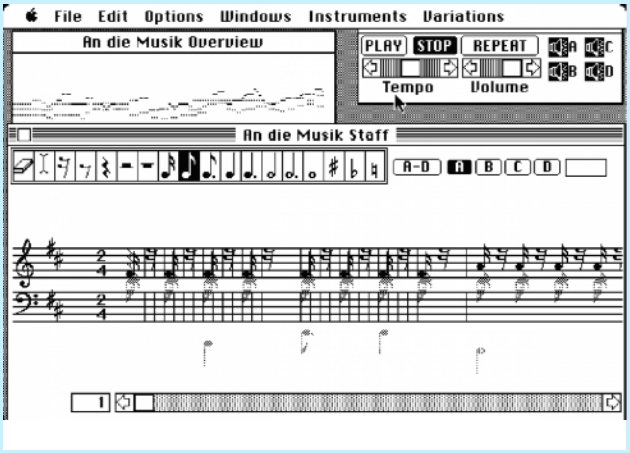
itty-bitty-vtty is based on the vt100 standards set forth in the vt100 online User Guide. According to the Github page for the itty-bitty-vtty vt 100 emulator, underspecified behaviors have been tested using the MAME emulator's vt100 emulation.

There are some limitations in hardware which have caused the following items to go unimplemented in the itty-bitty-vtty vt100 emulator:

- Alternate character sets
- Graphic rendition (except plain/reverse)
- 132-column mode

All other items in the User Guide are implemented and should provide the user with a realistic feel when using the itty-bitty-vtty vt100 emulator. You can download the latest release of on a ProDOS Order disk as well as the source code from the Github page: https://github.com/ksherlock/itty-bitty-vtty

Clock Signal Emulator



Tom Harte has updated the Clock Signal multi platform emulator. This release updates to the Apple II and Macintosh emulation sections of the program. This emulator allows users to not only emulate the Apple II and Macintosh, but also a host of other computing platforms including the following:

Acorn Electron

Amstrad CPC

Apple II/II Plus and IIe

Atari 2600

Atari ST

ColecoVision

Commodore Vic-20 & Commodore 1540/1

Macintosh 512ke and Plus

MSX 1

Oric 1/Atmos

Sega Master System

Sinclair ZX80/81

Sinclair ZX Spectrum

Tom has been consistently updating and adding new features to each of the platform he supports. In addition to this, he created a CP/M executor for macOS. For information about this product, check out Tom's Github page.

For more information or to download the latest release of the Clock Signal Emulator, go to the Github page for the emulator: https://github.com/TomHarte/CLK

TimeOut Edit BASIC

While we highlighted this at the release of it, we decided we should answer the question once again, "What is *AppleWorks* Good For?" which is a question we hear often in this day and age.

Well, our *AppleWorks* aficionado, Hugh Hood has the answer to that with his *TimeOut Edit BASIC* version 5.4.

According to the product blurb written by Hugh on his website, "**TimeOut Edit BASIC** is a 21st century addition to the family of *TimeOut* Applications for use with *AppleWorks 5.1* on the Apple II series of computers.

Considering that former Beagle Bros Alan Bird invented the TimeOut engine for *AppleWorks*, and also wrote arguably the best stand-alone BASIC program editor, *Program Writer*, it is surprising that a *TimeOut* application of this nature wasn't released years ago. But, as far as I can determine, it wasn't.

Quite simply put, the purpose of *TimeOut Edit BASIC* is to allow users to load, view and edit Applesoft BASIC programs directly from within *AppleWorks 5.1*. Configuration options include displaying control characters in inverse and indenting long lines."

You can download *TimeOut EDIT BASIC* and the complete source code for the program from:

http://www.apple2works.com/timeouteditbasic/

ROM-inator for Mac

You can still get the ROM-inator for the Mac Plus, Mac 512K and Mac 128 from Kay Koba:

Mac ROM-inator I kit for Mac Plus, Mac 512K / 128K

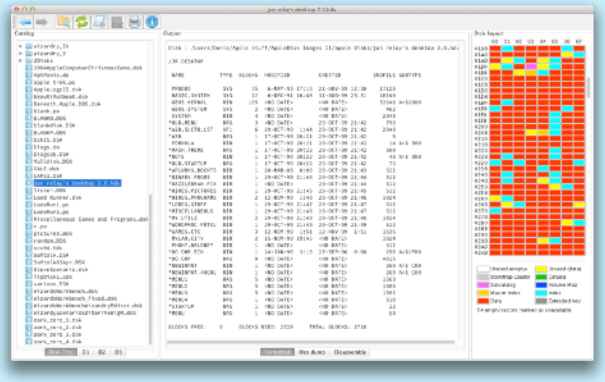
Economy Kit: 32 USD (Pin headers, IC sockets and wiring material are not included)

Standard Kit: 36 USD(All necessary parts are included, only tools are required)

Luxury Kit: 54 USD (Standard kit and 2 sets of ROM)

To Order: https://en.infinityproducts.co.jp/product-page/rom-inator-resurrections

Apple II Disk Browser II



Denis Molony, the author of Apple II Disk Browser, has come up with a nifty tool for Apple II users who want to get to the guts of their floppy disk images in a hurry. Apple II Disk Browser is a tool which allows you to flip through several of your disk images at a time and see the actual contents of those images right down to the sector level.

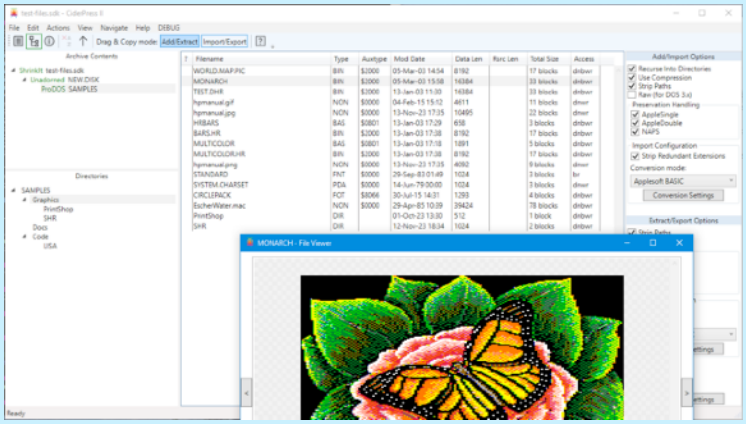
Apple II DiskBrowser II is a major rewrite of DiskBrowser, using JavaFX and the new libraries (AppBase, AppleFileSystem and AppleFormat).

The goal is for DiskBrowser][to retain all the file display formats of DiskBrowser, but with a better interface and a more maintainable code base.

The complete source code for Apple II Disk Browser has also been made available by the author.

Download Apple II Disk Browser from:
https://github.com/dmolony/DiskBrowser2

Ciderpress II 1.1.0



CiderPress II is a software tool for working with vintage Apple software, specifically that used on the Apple II series of computers and early Macintoshes. It is the successor to the CiderPress utility.

Key features:

- Supports a variety of vintage archive formats, as well as file attribute and resource fork preservation in ZIP archives.
- Supports many disk image formats and filesystems, including multi-partition disk layouts.
- Works directly with physical hard drives and removable media.
- Provides file converters for many document, graphic, and code files.

The tool is available with a graphical user interface (GUI) and a command-line interface (CLI). These have roughly equivalent feature sets. The CLI runs natively on a variety of systems (including Windows, Linux, and macOS), while the current GUI runs only on Windows.

Download *Ciderpress* II from: https://ciderpress2.com/

Minnesota Educational Computing Consortium
Software Repository

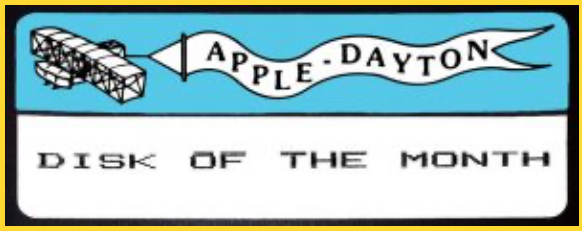
Featuring the full history of MECC and their software right from the officials and authors from the original company.

https://www.mecc.co/new

AppleCommander 13.0

AppleCommander is a disk image file manipulation package which allows conversion of files between formats as well as the export of files. AppleCommander is available as a JAR file and runs on most major platforms that run Java.

Visit Dr. John B. Matthews site:
https://github.com/AppleCommander/AppleCommander



The Apple Dayton User Group Disk of the Month Library

Tons of software, free to all Apple II users. Download the library today!

www.callapple.org/software2/appleii/apple-dayton/

The A.P.P.L.E. Apple II Games Library

All of your favorite Apple II games from the modern era all in one place

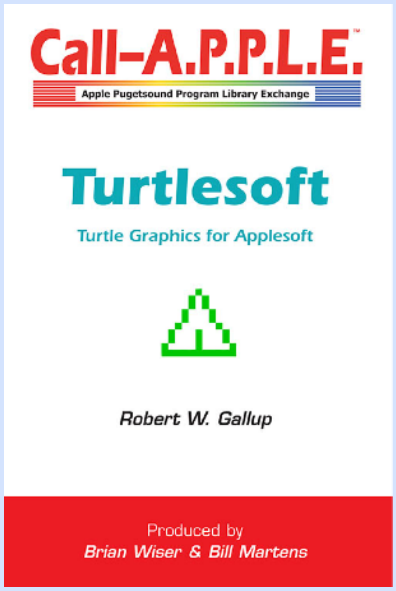
www.callapple.org/games/

FORTH INTEREST GROUP

fig

https://www.forth.org

for iOS – Just $1.99



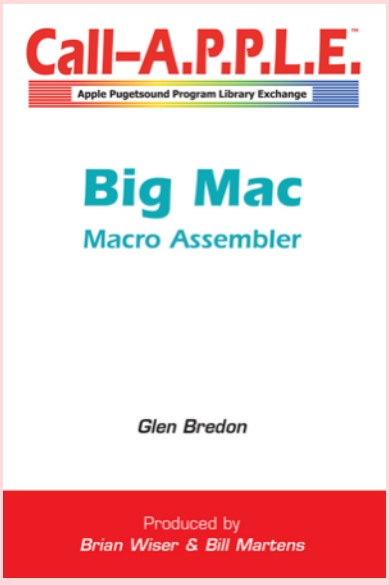
Turtlesoft

DOS & new ProDOS version free for Members:

www.callapple.org

Manual for Everyone

www.callapple.org/books



Big Mac is an advanced editor-assembler for Apple II computer programmers. This book includes documentation for *Big Mac: Macro Assembler*, along with *Big Mac.LC*, *Symbol Cross-Reference*, and *Symbol Symon* that expand its capabilities.

Book and Software at:

www.callapple.org/books

Huibert-Aalbers Software

Jigsaw!

A very simple game, written in less than a week. It was never meant to be published but my editors at Britannica liked it and convinced me. *Jigsaw!* sold over 100,000 copies on multiple platforms.

LaserForce

A 3D high-speed action game for the Apple IIGS with sound using the Ensoniq chip.

SoundSmith

This is probably the most rewarding program I have written. *SoundSmith* became an instant hit. People had purchased their Apple IIGS to enjoy their Ensoniq chip and there wasn't a decent music application. *SoundSmith* filled that gap.

AZERTY

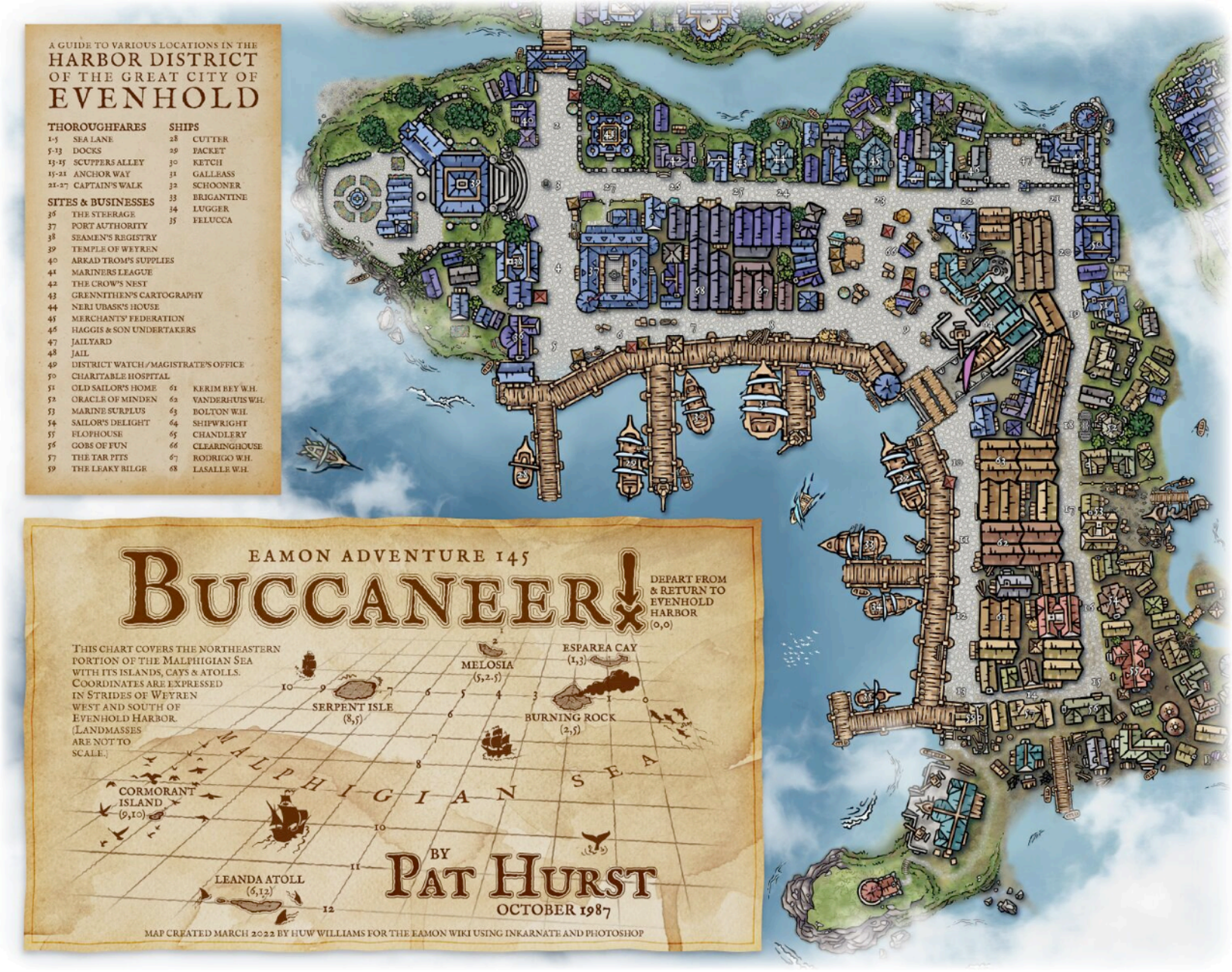
A simple NDA that allowed French IIGS users to fully use their keyboard when working with GS/OS applications.

Jigsaw Deluxe

An improved version of my best selling game, *Jigsaw!* This application was lost for almost 30 years and is now finally available for download.

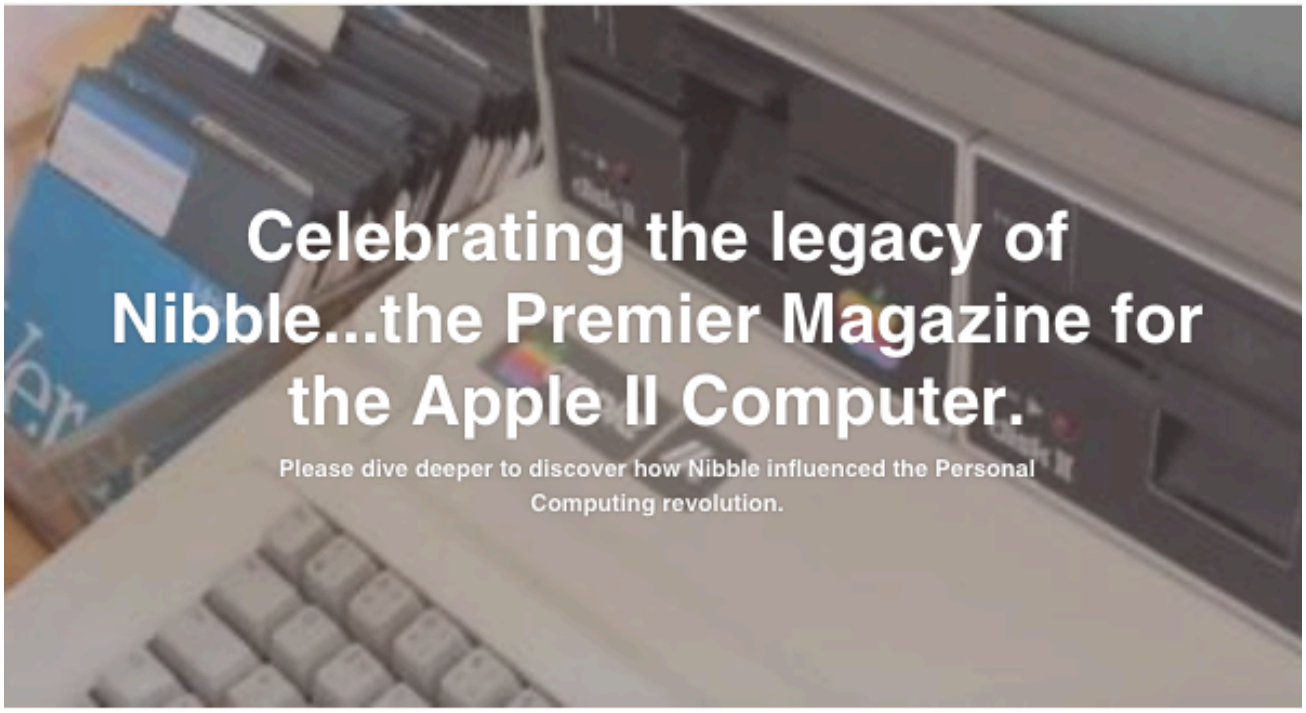
Get These Titles from:

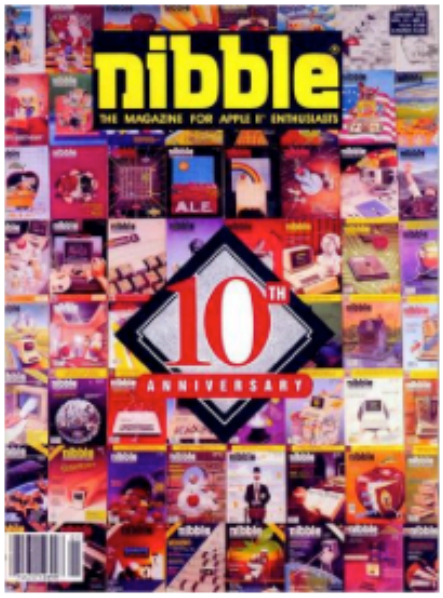
www.huibert-aalbers.com/AppleIIgs/Apple2gs.html



nibble

Home Catalog





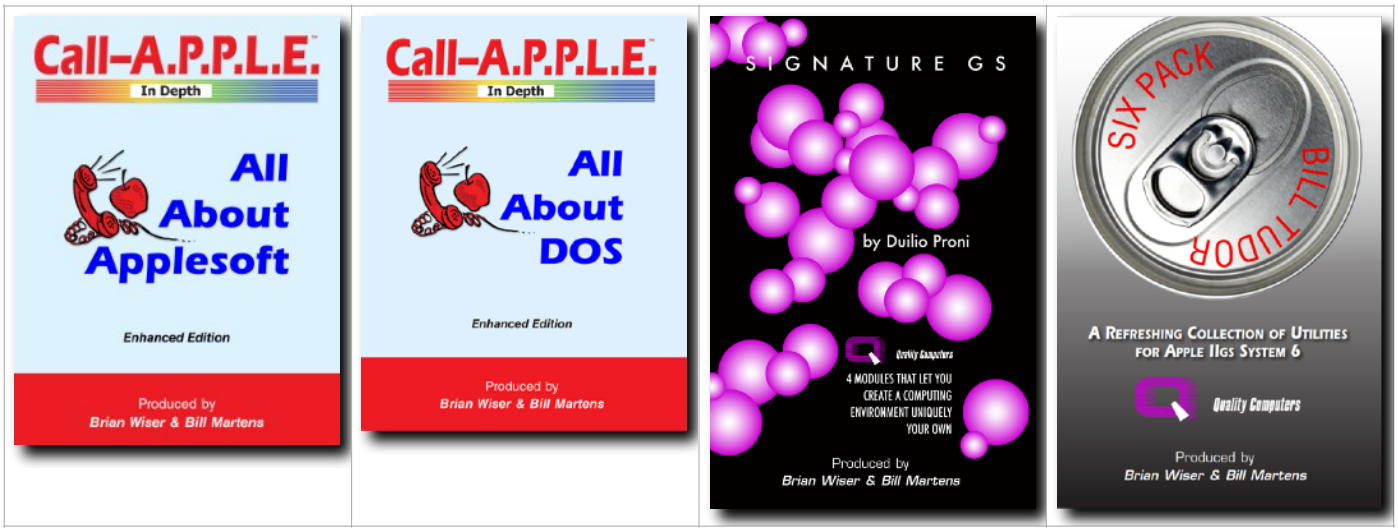
Welcome to Nibble Magazine...

WELCOME! It's hard to believe that more than thirty years after the Apple II came on the scene, there are still enthusiasts gathering the bits and pieces that still linger from the 1980's. My father, Mike Harvey, made Nibble Magazine his passion. With the help of a lot a contributors, friends, and family he was able to contribute to the Apple II (and Mac and PC) community for over 12 years.

Feel free to explore and reach out to us directly, as we want to make this site a destination for all looking to recapture the excitement, nostalgia and wonder of the first significant PC---the Apple II. The magazine, programs, and books that Nibble published are all available here for purchase as well.

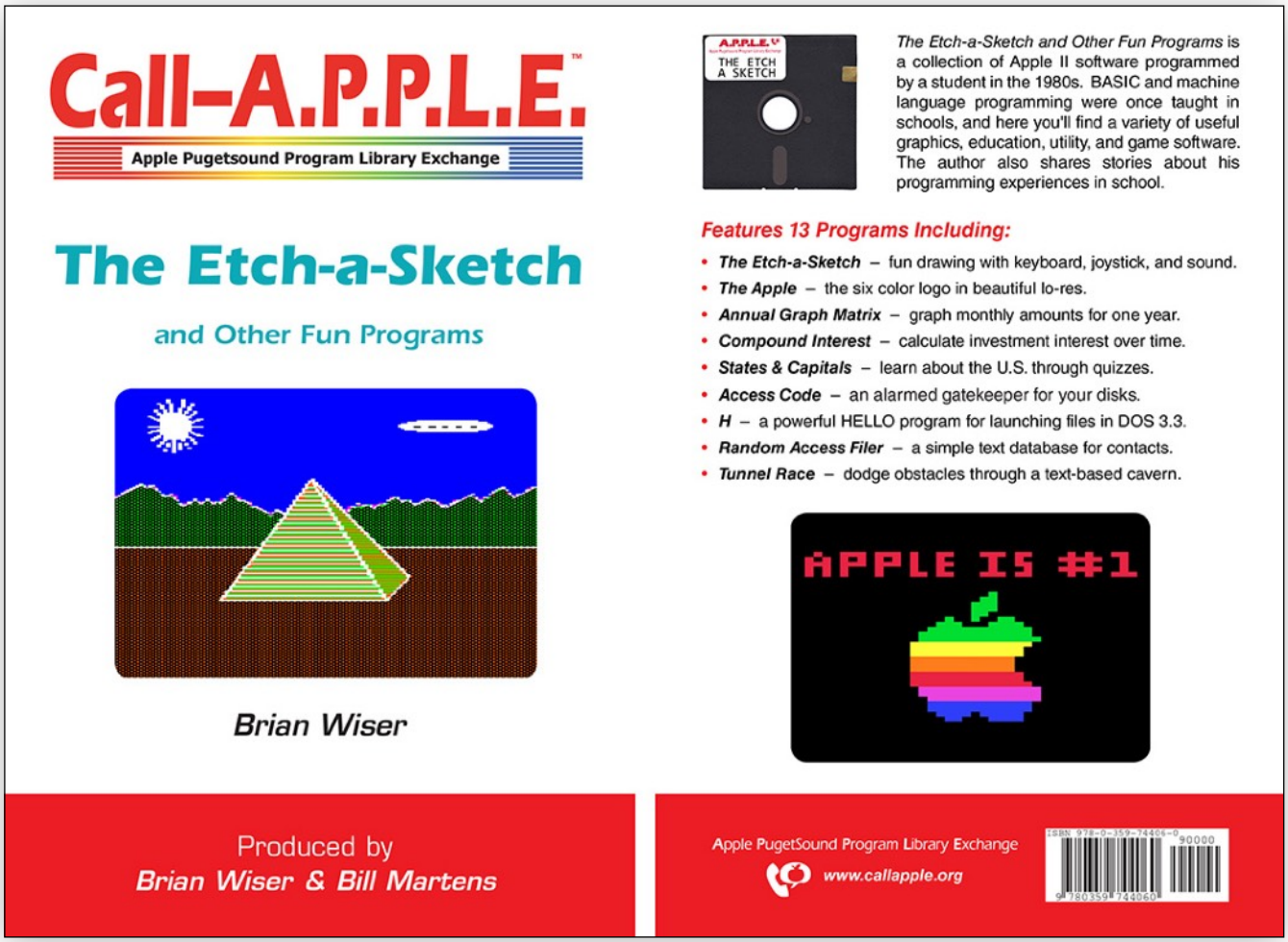
Get your copy of the complete ***Nibble*** **magazine**:

www.nibble-magazine.com



Available at: www.callapple.org/books

Complete with all programs on FREE floppy disk image



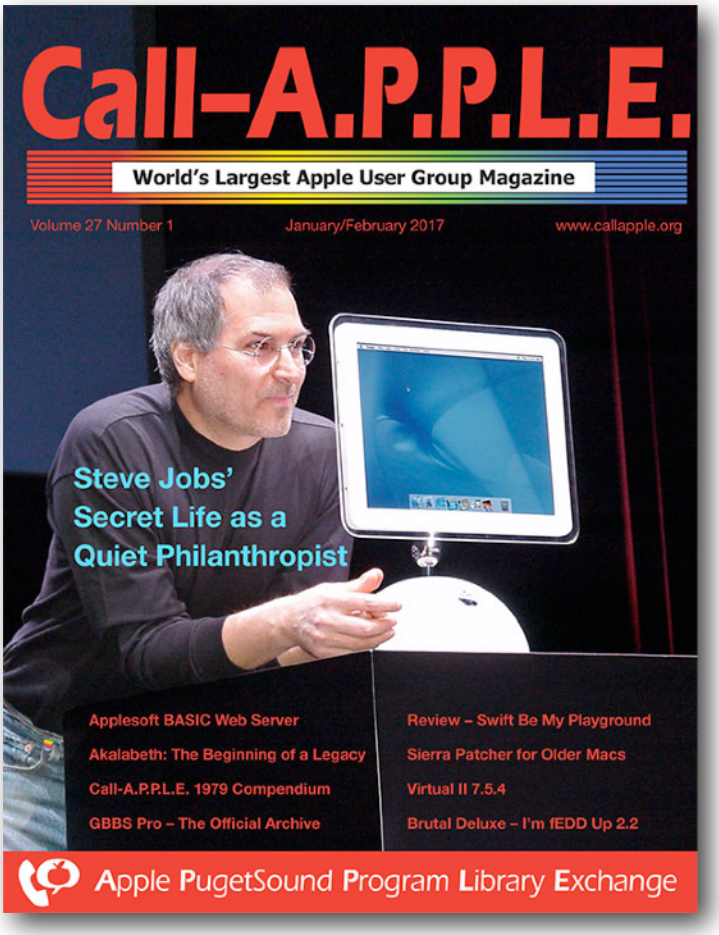
Article Submission Guidelines

Do you have content you'd like to share with the A.P.P.L.E. community?

Call-A.P.P.L.E. is always looking for new and interesting ideas and articles related to Apple, Linux/Unix, technology of general interest, technical or editorial, modern or retro.

All submissions must be original works of the person submitting and free of distribution restrictions. Please email your proposal to: editor@callapple.org.

By submitting materials to us, you agree to give A.P.P.L.E. the royalty-free right to publish and reuse your submission with your name in any form in any media. We reserve the right to edit, change, or not use anything submitted to us.



Join A.P.P.L.E. Today!

Access every legacy PDF issue of Call-A.P.P.L.E. magazine!

With your membership, the new, current issues of the magazine are included along with access to many back issues, articles, books, software and discounts for products.

Only $27.95 Per Year!

callapple.org/members

Primary Sites:

Apple Pugetsound Program Library Exchange (A.P.P.L.E.)
https://www.callapple.org

A.P.P.L.E. – New Books
https://www.callapple.org/books

Apple Archives
https://www.applearchives.com

Virtual Apple][
https://www.virtualapple.org

Gamezyte
https://www.gamezyte.com

Applied Engineering
https://ae.applearchives.com

Beagle Bros Official Repository
https://beagle.applearchives.com

Minnesota Educational Computing Consortium
https://www.mecc.co

WOZ Speaks
https://wozspeaks.callapple.org

GBBS Pro – The Official Archive
https://gbbs.applearchives.com

Vintage is the New Old
https://www.vitno.org

Additional Sites:

Take-1 Movie Site
https://take1.applearchives.com

Willamette Apple Connection
https://wac.callapple.org

Digisoft Innovations
https://digisoft.callapple.org

Australian Apple Review
https://aar.applearchives.com

Computist Project
https://computist.applearchives.com

A2-FS1 Flight Simulator Site
https://fs1.applearchives.com

A2-FS2 Flight Simulator Site
https://fs2.applearchives.com

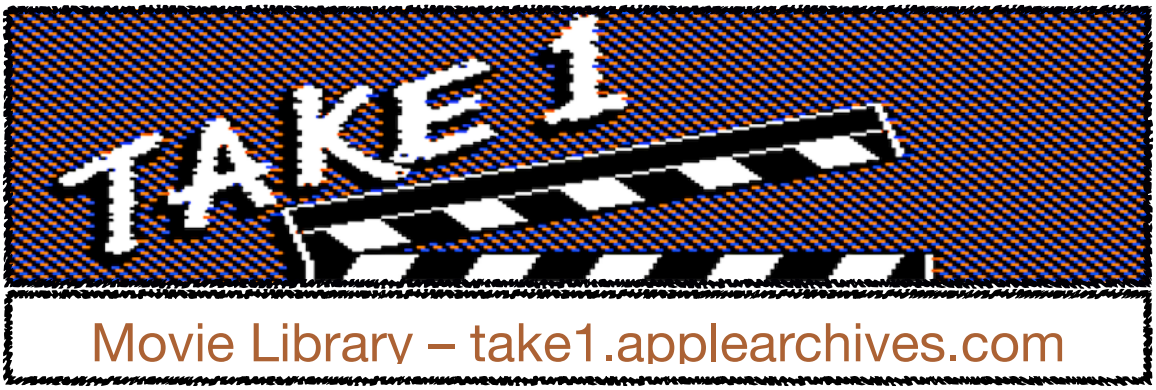
A2-PB1 Night Mission Pinball Site
https://pb1.applearchives.com

Ancestorworks Repository
https://anw.applearchives.com

The Apple /// Resource
https://apple3.applearchives.com

Apple II Hackers
https://hackers.applearchives.com

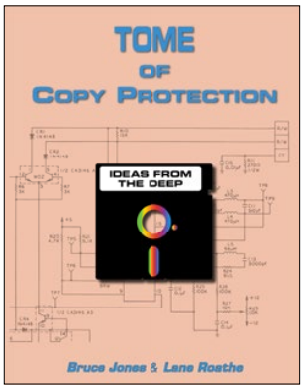
Terry Allen's Apple II Page
https://apple2.callapple.org

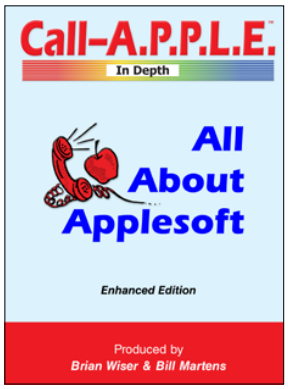


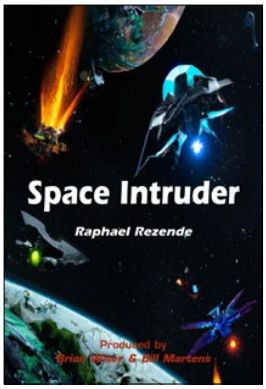
Call-A.P.P.L.E.™

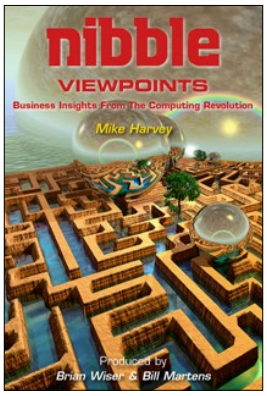
World's Largest Apple User Group *Since 1978* www.callapple.org

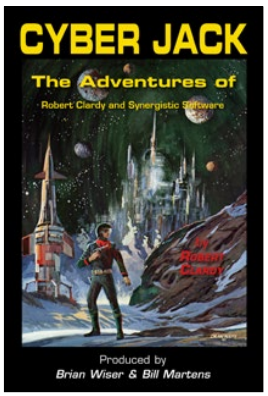
Read our Books & Magazines – Play Games – Join our User Group

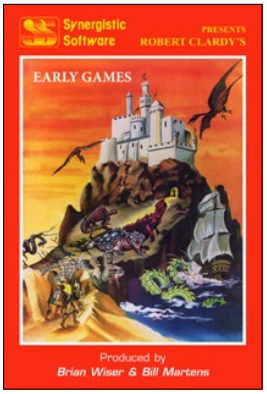


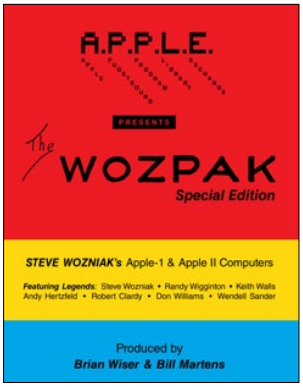


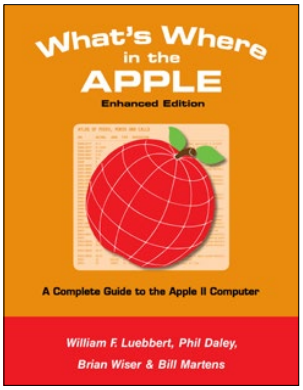


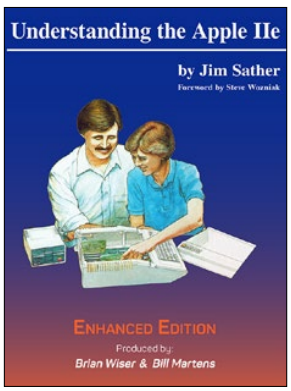


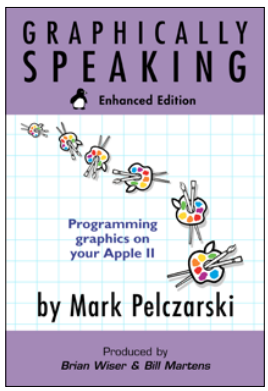


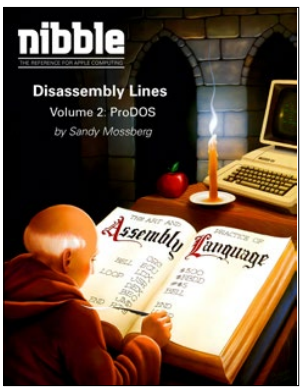












^----- CallAPPLE.org/books -----^

v----- Our Sites -----v

Retro & Current *NEWS*
callapple.org

AppleArchives.com
The Best Vintage Apple & Mac Websites

BEAGLE BROS
EST 1980
beagle.applearchives.com

VIRTUAL apple II
VirtualApple.org

mecc.co
VITNO.ORG

APPLIED ENGINEERING
ae.applearchives.com