A2-Central™

formerly
Open-Apple

July 1990
Vol.6, No.6

ISSN 0885-4017
newstand price: $2.50
photocopy charge per page: $0.15

A journal and exchange of Apple II discoveries

Picture this

At the 1989 World Science Fiction Convention, there was a panel entitled "*TheWatchmen*:–best other form of '87". For those not familiar with the leading edge of science fiction and fantasy, *Watchmen* was a landmark "graphic novel"; a combination of the best of comic book art and novella-style writing. The graphic novel has been considered a "new form"; the literary content exceeds the depth of a comic book, but it's not strictly a literary work because the illustrations are integral to the form.

There had been several graphic novels before *Watchmen* and it wasn't the first to cause controversy (a Batman graphic novel, *The Dark Knight*, had previously been nominated for one of science fiction's coveted Hugo awards in a *non-fiction* category because it didn't fit any of the existing literary criteria!). As presented *Watchmen* was such a departure from our current definitions of "literature"; the text alone would qualify as a novel in size and scope, but the graphic content is crucial to the presentation of the text. Demonstrated brilliance in the execution of an idea makes the idea itself valuable; *Watchmen* is interesting both as a work of science fiction and as an example of a new form of communication.

In the computer arena, the buzzword is "multimedia". Multimedia combines various visual and sonic elements to communicate an idea. Instead of just a book or a videotape or a compact disc or a computer program, some or all of those elements are combined to create a work that is something more than the sum of its parts. (*Watchmen* might, using this terminology, be called a "multimedia novel".)

Like most new technologies, there has been a lot of hype surrounding multimedia; we've heard suggestions that it is the salvation for our educational woes and that it is the ultimate end-user technology. Like any technology, successful application of multimedia techniques will tell more about its true value than the sales emphasis that is placed upon it. There may also be a lot of inconsequential uses.

So far, the elements for computer-based hypermedia have been a controlling program to mix the various computer-generated graphics, sound, and text elements, and peripherals to augment the computer's features, such as video monitor systems (for display elements), CD-ROM (as a source of computer data, sound, and, images), sound systems (from simple speakers to stereo systems to synthesizers), and so on. Specialized computer applications like Apple's HyperCard have appeared to make the job of creating a hypermedia symphony easier for the "average" user, who is not familiar with a programming language in the classic sense.

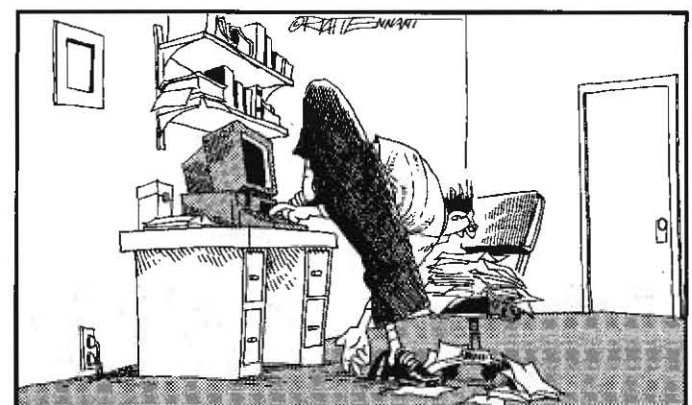
Videotape players can also be controlled by a computer using an interface called the *VidClip* (APDA sells a "*VidClip* Videotape Control Toolkit for the Apple IIgs Developer", part number T0360LL/A, for $79). But shuttling along a videotape to find the sequence you want is slow; this is one reason computers don't generally use tape for data storage. Instead, they use the now-familiar disk drive.

But video is definitely an emphasized part of multimedia. If you want to use a video segment in a multimedia production, what can you do? The answer is you use an interactive video disc; the surviving technology in that realm is Pioneer's LaserDisc.

Laser video disc players have been with us for years. The early models, among them the Pioneer VP-1000, retailed for about the price of their VCR contemporaries. Laser disc software was rare in those days, as it is for any start-up technology, but some videophiles bought players for the advantages they had over the videotape recorders of that time: high-fidelity stereo sound, sharper video reproduction, and the comparatively low cost of laser discs. For example, the cost of *West Side Story* on laser disc at that time was approximately half as much as the video tape release.

The perceived disadvantage of the laser disc player was that it was a "read only medium"; VCRs allowed taping other video sources for playback later, including broadcast television. VCRs heavily outsold video disc players and after the demand for pre-recorded tapes rose enough to offset start-up costs, the price of tapes began to fall drastically. Stereo VCR's began to appear, eventually incorporating high-fidelity sound. Meanwhile, there were quality control problems in the manufacture of laser discs, and their availability was limited. VCRs crowded laser disc players into the dark corners of the homes of hard-core video enthusiasts. Laser disc players survived on the basis of this "videophile" market, and due to one other advantage: video discs could be accessed interactively at very high speed, which made them useful in training and education. The adaptation of laser discs to interactive training probably sustained the medium when the consumer market couldn't.

The laser disc playback mechanism rotates the disc at a top speed of 1800 RPM. The sequences of images are recorded in one of two speed-related formats: CAV (constant angular velocity) where each image occupies the same number of degrees along a laser disc "track" (the disc is rotated at constant speed), and CLV (constant linear velocity) where the image occupies the same linear distance along a track and the disk speed is slowed as the playback beam reaches the outer tracks of the disk (laser players play from the inner portion of the disc outward). More images will fit on the side of a disc in the CLV mode (about twice as many), so the maximum playback time per side of a CAV disk is about 30 minutes while the maximum playback time of a CLV disc is 60 minutes. A disk can have two sides, so the maximum total real-time video information that can be stored on a disk is 1 hour (CAV) or 2 hours (CLV).



SO POORLY DOCUMENTED IS THE SOFTWARE THAT ROY IS BETA TESTING, THAT HE FAILS TO NOTICE THAT THE GAME RULES TO "TWISTER" HAVE ACCIDENTALLY BEEN INCLUDED.

CLV is obviously used when the duration of playback per disc is an issue. But CAV's organization of one image per track facilitates several types of frame-by-frame effects, including still frame, variable speed playback (in forward or reverse), frame by frame search, and so on.

CAV laser discs can store up to 54000 frames per side; assuming you can place 1000 characters of written information in each frame, you can store the equivalent of several (over 300, figuring 70,000 words per book and an average of 5 letters per word) books on that single disk.

All laser disc players were designed with the ability to detect a "table of contents" area on the disc which can contain an index of available "chapters" on the disc. By pressing a few keys on the player (or a remote control), you can search for a specific chapter or frame on the disc. This made possible interactive video, commanded with a relatively simple pushbutton control. So in addition to the normal "real-time video" capability, laser disc players allowed the production of interactive games, video encyclopedias, and so on, where you used the chapter and frame search features to navigate the disc's contents. Arcade fans may remember the game *Dragon's Lair*, where a laser disc player was integrated into the game console, and a player's commands via the game console caused the corresponding section of video to be played. The effect was that the video sequence played reflected the actions of the arcade player, giving a spectacular real-time feeling to the game. Another application is an interactive display such as a shopping mall directory that can accept keypad commands to display the description and location of a specific store.

The natural extension of these ideas is to use a more intelligent remote control: a computer. Pioneer, Sony, and other companies sell "industrial" laser disc players which can either use their own internal circuitry to accept programmed playback (the players include text generators to allow displaying comments overlaid on the video image), or that can be interfaced to a an external controller or computer. The combination of laser disc player and programmable controller allows the laser disc to become the audiovisual amanuensis of the computer for business or educational presentations. The key problem, as for CD-ROM, is to get the disc for your specific presentation produced, if such a disc does not exist. There are currently several "interactive" discs in production, from general interest discs such as the *KnowledgeDisc* to special interest volumes such as the NASA Discovery discs. A few of the publishers are Optical Data Corporation (30 Technology Drive, Box 4919, Warren, N. J. 07060, 800-524-2481 and 201-668-0022), Videodiscovery (P.O. Box 85878, Seattle, Wash. 98145, 80054V-DISC, and the Voyager Company (1351 Pacific Coast Highway, Santa Monica, Calif. 90401, 213-451-1383).

In addition to producing a line of special interest volumes, some with associated *HyperCard* stacks to automate their investigation, Voyager has a line of high-quality reproductions of popular films known as the Criterion Collection. Criterion movies cater to videophiles by reproducing the movie in a format as close to the theatrical release as possible, including the use of wide-screen formats. (Using the wide-screen format presents the film as a "strip" across the center of a standard TV screen, with empty bands above and below the image. This "letterboxing" technique has met with some resistance among mass-market consumers of videotapes, but many videophiles feel it is the only acceptable way to present a wide-screen movie on home video.) In addition, the Criterion disks often are produced in the CAV mode, and may include a commentary soundtrack (in addition to the movie soundtrack), additional scenes not appearing in the final "cut" of the movie for theatrical release, historical and background information on the movies, and so on. These extras come at a price; the commercial mass-market release of *West Side Story* on laser disc is $39.98. The Criterion CLV release (two discs) is $59.95, the CAV release (with extra material) is a stiff $124.95. These are products for connoisseurs.

Currently, there are several Apple II compatible products that support developing "mediaware" that uses video discs. For the 128K Apple IIe and IIc there is version 2.6 of *Tutor-Tech*, which supports interfacing to several of the industrial video disc players (Pioneer models LD-700, CLD-909, LD-V2000, LD-V4000, LD-V4200, LD-V8000 and their Hitachi, Magnavox, Mitsubishi, and Sony counterparts) and Scholastic's *HyperScreen* (reviewed elsewhere in this issue) which supports the LD-4200. For the IIgs, there is of course *HyperStudio*, which includes an XCMD (external command) for the Pioneer 4200 command set. Both systems can also incorporate use of the Apple Video Overlay Card for combining video disc and computer video on a single monitor.

Bringing it home. There is an exiting new class of laser disc players out that makes playing with interactive video more of a home hobbyist occupation than something for videophiles only. The representative current Pioneer models are the CLD-1070 ($495), CLD-2070 ($795) and the CLD-3070 ($1195); these players can play compact discs as well as video discs.

These players have a "Control" interface that is used to integrate the player into a home stereo/video system using a single "master" remote control. By replacing that remote control with a computer interface that acts like the remote control, you can have the computer control an entire video and audio orchestra, with an interactive video disc player at the center. All you need is the device to make the stereo/video system think the computer is the remote control.

Visual Database Systems manufactures such a device: the *SIA III Serial Interface Adapter*. Visual Database Systems is primarily a development house and is not prepared to do end user support for the consumer market. Instead, they generally sell the *SIA III* to companies like Voyager with the expectation that the company they supply will create and support a product for the consumer market using the device. Voyager sells the interface with Macintosh software as *The Box* ($149.95) to allow their Criterion *HyperCard* stacks to be used with various consumer-level players, but does not have equivalent IIgs software. Our interest, then, was seeing if we could create tools to allow using the *SIA III* with an Apple II.

Before we talk about the interface itself, you should realize that the consumer-level players are not as sophisticated as the industrial players in terms of their innate interactive abilities, but the computer can provide most of that functionality. You also give up some of the frame-access speed of the high-end industrial players that are sold specifically for interactive use. But with the newer consumer players, you gain other features that make them more justifiable for stand-alone purchases.

Let's look at the $500 Pioneer CLD-1070, which includes the player, stereo and video connecting cables, and cordless remote control. This player can access all current laser media formats except the data portions of CD-ROM. This includes the 3 inch "CD single" (audio), 5" CD (audio), 5 inch CD-video (up to a five minute video track with up to an additional 20 minutes of CD audio only), and 8 and 12 inch video disc formats. Current industrial players can only access the video disc formats.

Video discs can contain digitally recorded audio tracks in addition to the standard "analog" tracks, and these are reproduced via the same type of circuitry used in CD players (thus the player acts as a CD player with remote control). You can selectively listen to either the stereo analog or (if available) digital soundtrack, or to either the left or right channel of either. This means you can have two individual stereo channels (digital or analog) of audio information, and either or both of those channels can be split into two separate monophonic tracks.

As mentioned earlier, video discs can also be encoded with a "table of contents" to allow defining the start of "chapters" on the disc. Using the analogy of chapters in a book, each of the video disc chapters consists of a range of frames (correlating to the pages of a book). Chapters are available, if the table of contents exists, on either CAV or CLV disks, but only on CAV disks are the individual frames accessible via indexing. On CLV discs, you can elect to search for a chapter and the player will attempt to locate it and play it. On CAV disks, the player will locate the chapter and return to the mode it was in (play, pause, still frame display, etc.) when the search was requested. Or the index may contain an indication to have the player stop on the leading frame of the chapter (to allow a manual "frame advance").

All of these functions are supported on the $500 player, and can be accessed from the remote control. In addition, the player supports some simple programmability functions, such as selecting only specific chapters to play, the order in which to play them, and repetition of all or a portion of the disc. More expensive players add features, of course; for example, the CLD-3070 adds a "jog" ability which allows

still-frame displays on CLV discs. Some players have a "marker" feature which allows simulating the ability to play a segment of a disc.

The *SIA III* takes ASCII data from a computer's serial port and generates a pulse stream that can be fed into an "SR" (for "System Remote") control port. If you're not familiar with the concept, the SR ports on some component systems allow "daisy chaining" the system together so that a single master remote control can control all of the components in the system. The *SIA III* accepts single character ASCII character equivalents for some commands (such as "P" for "Play"). Those that don't have a single character equivalent can be sent as a hexadecimal byte by prefixing the byte's two-digit ASCII representation with a "/"; for example, "/1E" sends a hexadecimal 1E (decimal equivalent 30) byte to the SIA III.

The SR mechanism uses a prefix code to select the "listening" device; you send this code to tell a device that it is supposed to respond to the codes that follow. The device will continue to accept command codes until a new prefix other than its own is received, when the newly specified device (if it exists on the chain) should start responding. These prefix codes are manufacturer and device specific to prevent interactions between the devices individual remote controls; for example, you don't want your JVC videocassette remote to turn on your Pioneer LaserDisc player when you're trying to view a tape. When integrated into an SR system, the "master" remote control (associated with the system controlling device that all other devices are connected through, such as a control amplifier) includes the ability to send a command to switch among the devices on the chain so the individual remote controls can be boxed up and stored (in fact, using the SR connection disables the remote control for any devices other than the system controller). The *SIA III* includes a "decoding" ability that can be accessed by making a special cable so that you can attempt to read and display the codes for other SR remote controls (most of our older non-SR remotes didn't even register).

Using the *SIA III* on the Apple II is absurdly easy, once you figure out that it requires data to be sent with the high-order bit clear. You can communicate with the *SIA III* with a terminal program (we used *Talk is Cheap*). Since the *SIA III* has its own small internal buffer, you can even send commands to the interface with an Applesoft program. Our main loop to gather the ASCII command from the keyboard and send it to the device looks like this:

```
1000  REM — Issue SIA commands --
1005 D$ =  CHR$ (4)
1007  GOSUB 10000
1010  TEXT : HOME
1020  VTAB 23: HTAB 1: CALL  - 868
1030  INPUT "Command?: ";A$
1035  IF  LEFT$ (A$,1) = "." THEN  GOTO 1090
1040  PRINT D$;"PR#2"
1050  PRINT  CHR$ (1);"12B0D0P": REM 4800baud, 8N1
1060  IF  LEN (A$) THEN  GOSUB 2000: REM send command
1070  PRINT D$;"PR#0": REM back to screen
1080  GOTO 1020: REM loop until "quit"
1090  HOME
1100  END
```

Line 1050 sets the serial port to the requirements for the SIA III: 4800 baud, 8 data bits, no parity, and one stop bit. Line 1007 calls a subroutine to install a machine language routine that strips the high bit off an ASCII character before sending it to the output device; we need this because Applesoft sets the high bit when PRINTing characters:

```
10000  REM --- Install ML routine ---
10010  FOR EYE = 768 TO 774: READ THUMB: POKE EYE,THUMB: NEXT EYE
10030  DATA 169,0,32,237,253,24,96
10098  RETURN
```

The subroutine used in line 2000 sends the command string to the *SIA III* via the serial port one character at a time through the machine language routine:

```
2000  REM — send A$ by char —
2010  FOR I = 1 TO  LEN (A$)
2020 : POKE 769, ASC ( MID$ (A$,I,1))
2030 : CALL 768
2040 : NEXT I
2050  RETURN
```

It is possible to use the *SIA III* with present Apple II multimedia software. *HyperStudio* supports the issuance of the codes through external commands (XCMDS) that can be written and incorporated into stacks; *HyperStudio* 2.1 includes a sample XCMD for sending commands to the serial port that can be suitably modified. *Tutor-Tech* allows sending ASCII sequences to a serial device, but we haven't have any luck (yet) with *HyperScreen*, which appears locked into the LD-4200 commands (the program may expect a return status which the *SIA III* can't supply). We've also got a (very rough) IIgs NDA running that functions as a remote control using GS/OS device calls to the serial driver that can be used from desktop applications.

All is not perfect. The *SIA III* cannot receive status information from the device from the SR port and will always return an indication that the last command was received and executed. If you send a command to a device there is no way to tell if the device received it and was able to act upon it, or whether the device has completed a previous task. There are also the limits originating in the devices themselves; for example, the CLD-1070 does not have the capability to play only a specified segment of a disc once (for example, "play frames 1000 to 2000 and pause"), a capability removed from the earlier equivalent model, the CLD-1030. The current CLD-2070 and CLD-3070 models do have the ability to play to a preset marker and stop which can be used to simulate the "play a segment" function, but there's no guarantee Pioneer will retain the ability in future revisions.

But we still see advantages to the consumer players; they are less expensive and they play a wider variety of discs. The sales appear to be rising for these players just as another "playback only" medium, compact discs, has exploded. As the likelihood rises that the average computer user may also have a laser disc player at home, maybe the expectations for the incorporation of "commercial" features into the "consumer" players will rise.

Despite the limitations of the consumer video disc players versus the industrial versions, here is what we see as the truly interesting capability of these players in combination with the *SIA III*: imagine a couple of video monitors, a couple of video disc players, and a premier audio system connected to a computer via the *SIA III*. The computer could issue commands to raise and lower the audio volume, change the balance, change the video or audio sources and target devices. With the proper controlling software, the computer would become the orchestrator of a multimedia show.

A more modest system could consist of your Apple II system, a consumer laser disc player, the *SIA III*, a hypermedia product, and a monitor for the player (or a Video Overlay Card to overlay the player's output on your current monitor). For example, I'm using the CLD-1070 connected to a stereo (if you want external speakers on the IIgs, the IIgs would need its own set) and a Video Overlay Card as I type this; when I'm not experimenting with the video features, I can pop a CD into the same player and listen to music. A more practical example might be in a music class; a hypermedia production could use a video disc describing aspects of composition and have the student stop at points to insert various CDs into the player so that the software could play specific examples; something you couldn't do on the current commercial players. Of course, it would help if Pioneer would re-instate a way to play only a specified segment of the disc; for now, we're trying to find a way to simulate that function by software control

Even with all these features, the utility of the player as a computer peripheral depends on the availability of laser disc software applicable to your use, or justification for producing your own custom discs. The current advantage of laser disc over CD-ROM players is that the laser disc can easily be justified in terms of stand-alone use. (If someone ever adds CD-ROM support to a player at a consumer cost, it will be Nirvana.) Interfacing to a computer just adds a new dimension, and new dimensions expand the creative playground.—DJD

HyperScreen

HyperScreen is a (single high-res) graphics-based and menu driven hypermedia system designed for the 128K Apple IIe or IIc. *HyperScreen* comes in two forms: a $99.95 "teacher edition" with a detailed instructional manual (this is the package we received for review) and a $32.95 "trade edition" with the same program but a less detailed manual for home use. *HyperScreen* is available from Scholastic Software, 2931 East McCarty Street, P.O. Box 7502, Jefferson City, Mo. 65102, 800-541-5513 (800-392-2179 in Missouri).

Like other hypermedia programs, *HyperScreen* is based on the creation of a "stack" of graphic screens on which "buttons" may be installed. When creating the stack, you select the action that will occur when a button is selected; available actions are transferring (linking) to another screen (within the same stack), linking to another stack, displaying "pop-up" text, playing a sound, entering text, operating a video disc player (the Pioneer LD-4200 is specifically supported), or executing a "multi-button" sequence.

When creating a new stack, you can use the built-in graphics tools to create a graphics backdrop you feel is suitable to layer your control buttons on. The available tools are a brush for drawing "freehand" curves (the brush style is selectable from a set of options), a line tool for drawing straight lines (the width is selectable), a "frame" tool for drawing rectangular outlines, a "block" tool for drawing filled rectangles, "ring" and "oval" tools for drawing hollow and filled circles and ovals, a "flood fill" tool for coloring existing objects, a "type" tool for text, and tools for importing full backgrounds and clip art. The *HyperScreen* disk comes with a selection of backgrounds and clip art samples ready to use. You can select the type of "opening transition" you want the screen initially displayed with from 15 options such as "wipe down", "wipe left", "dissolve", and so on. Items can be drawn using a mouse, joystick, touchpad, or even the keyboard.

HyperScreen also provides for "text-only" screens that contain only text and a border; these are still displayed using the graphics screen, but use a more compact form of storage. Text screens allow the use of a built-in editor and text import facility, but they do not allow the use of graphic images on the same screen.

Once you've created the background, buttons are added by selecting the "Set Buttons" item from the "Edit" menu. You drag the button to the position on the screen and the size that you want; *HyperScreen* will name the button "Button n" where "n" is the number of the button. Once the button is created, you can select "Set Button Info" to choose the button action. To link to another screen, stack, or to play a sound file, you pick the name of the object from a list that *HyperScreen* displays. For a multi-button action, you enter the names for up to five other buttons on the same screen to be actuated in sequence. For pop-up text, you enter the text you want displayed. For text entry, you enter a pattern to match and the screens to branch to if the entered text matches or does not match. For video control, you enter the port (slot) you want the command sent to and the text of the command. If you have an Apple Video Overlay Card, you can use the Video Key Color option of *HyperScreen* to set the key color for a screen so that it can be overlaid with the video image.

You can test the button functions through a "preview screen"

Ask (or tell) Uncle DOS

Reader Kevin Thornton called to tell us that the Finder's Icon Info "calculate" function doesn't include "hidden" files in it's file count and storage use (by files) information. Hidden files are created by setting a specific flag bit in the file's "attributes" byte; the most common hidden files are Finder's data files (FINDER.DATA, FINDER.ROOT, FINDER.DEF). This normally won't be a problem, but if you have a program that reports "too many files" when Finder reports a number within the stated capabilities of the program, this may be the reason.

Another caller who had looked at the changes in the ROM 03 IIgs disk port says the changes reflect an attempt to reduce noise problems on the disk bus, and also commented that he didn't know of any current third-party 5.25 drives specifically advertised for use on the IIgs disk port (many are Apple IIe drives installed using an adapter). We'd therefore conclude that our first theory was correct, and that Apple made the changes without intending to "obsolete" any drives. We'd still like to know if there really are problems outside of the few reports we've heard; if you are having problems with a third-party 5.25 drive on the IIgs, let us know.

After the letter last month commenting on the use of the Leadman power supply with 220 volts, Tom Vanderpool looked carefully at the power supply circuit board again and found the magic incantation: you take a plug loose from the post labelled "110V" and move it to the one labelled "220V" (with the power off and supply unplugged, of course!). Again, we don't have any way to actually test this here, but we thought we'd pass the information along.—DJD

DMA confusion

I read with interest your article in the June ***A2-Central*** concerning the new *Apple High-Speed SCSI Card* and DMA compatibility. I had been having a devil of a time getting my new card to work. Each time I booted the card it would crash. After many tries and lots of telephone calls I got my upgrade for my *TransWarp GS* from Applied Engineering and eliminated all other possible cards. So I, at Apple's suggestion, concluded that the SCSI card was bad and returned it.

While waiting for a replacement to arrive, I read the June ***A2-Central***. In your article you said that you had not been able to use any card with more than four banks of chips. That rang a bell. My *GS-Ram Plus* has five megabytes on board. I had been repeatedly told by Applied Engineering that the card should work in that configuration. When the new SCSI card arrived, however, I pulled the last bank of chips (after again crashing on boot as a test) and the system worked!

Something kept nagging me, however. Applied Engineering said that the SCSI card should work with the five meg *GS-RAM Plus* but the card acted like those you mentioned. I finally called Applied Engineering and got gently obstinate. Although in all my prior conversations with them I had said that I had a five meg board, they never told me that for the board to work in that configuration any banks beyond four must be configured as a RAM disk. Sure enough, I got home tonight and replaced the fifth bank of chips and reconfigured my RAM disk from 800k to 1 megabyte and the board works fine in DMA mode.

The first moral of all this is, of course, to keep asking questions. More importantly, believe your vendor but *still* ask lots of questions. Finally, in a bit of irony I now have more usable memory than before. This is because I usually keep an 800K RAM disk in the configuration. That means that each time I booted with four megabytes *on* the motherboard 800K above and beyond the needs of the operating system was used. In short, I had upon booting 1655K used and 2596K available with a four megabyte board. With a 1024K RAM disk and five megabytes on the *GS-RAM Plus* I now have upon boot 1879K in use and 3496K available. From this I conclude that the RAM disk is set aside first in the boot process and that the SCSI card can and does then address the next four banks on the *GS-RAM Plus* in DMA mode.

All of this may be a bit academic. If any of your subscribers, however, are like me and typically use a RAMdisk of less than 1024K as part of the system configuration, they may be able to pick up added free memory by expanding their *GS-RAM Plus* to five meg with a 1024K RAM disk.

J. Latta
Evanston, Ill.

When you allocate a RAM disk, the memory for the disk is taken from the "upper" portion of the IIgs's expansion memory. The current crop of Applied Engineering memory cards have been designed to be DMA compatible,

option, and you can export the background or a clipped image from the screen for use in other screens. You can also print the screen to a selected list of printers.

Overall, *HyperScreen* is capable. It even has one strength (other than it's ability to work on the IIe and IIc) compared to *HyperStudio*: you can easily edit the action of a button (currently, to do this in *HyperStudio* you have to re-define the button).

But...Scholastic's vice-president Peter Kelman published an article in the February 1990 *Apple Direct* advising developers to "Program for 128K of onboard memory" on the Apple II and to write more advanced applications for the Macintosh, discounting the IIgs entirely. *HyperScreen* was mentioned in the article as one of several "IIe desktop-presentation tools that you'd be amazed to find on a IIgs". I can't agree with that opinion; I would definitely consider *HyperScreen* for my IIe, but *HyperStudio* is obviously the stronger hypermedia product, utilizing the IIgs super high-res graphics screen, extra memory, better sound tools, animation, and so on. *HyperStudio*'s XCMD feature provides more expandability than *HyperScreen* currently supports. Scholastic has to decide if it feels challenged by that or not, and customers that use a IIgs as their primary machine must decide if they are willing to work with a product that does not take full advantage of *their* computer.--DJD

Miscellanea

AppleFest's move to New Jersey netted them about 12,000 in attendance, a smaller figure than previous years. Andy Calkins of Exposition Management, Inc., says it appears likely the spring AppleFest may be back in Boston next year. AppleFest/ComputerFest for this fall will be September 14-16 in San Francisco; Exposition Management says exhibitor sign-ups for the fall 1990 and spring 1991 shows were strong. For more information, call 800-262-FEST.

More on using foreign characters with AppleWorks: Beverly Cadieux of KingWood Micro Software, 3103 Lake Stream Drive, KingWood, Texa 77339, 713-360-5013, sent us a note on *Ultimate Fonts*, a character conversion program for AppleWorks using *TimeOut SuperFonts*. The product works with *TimeOut UltraMacros* to convert text like:

```
Si', yo hablo E'span~ol!
```

to the *SuperFonts* form of:

```
S<x2>R<x1>, yo hablo <x2>C<x1>spa<x2>V<x1>ol!
```

which prints (using SuperFonts) as:

```
Sí, yo hablo Español!
```

This also comes in handy for various scientific and math typesetting problems, like "Ångstroms". *Ultimate Fonts* is menu driven and includes help screens to remind you of the translation formats. It's $30 plus $2.50 shipping.

As a follow-up to last month's article "Sculley committed to Apple II": Barney Stone of the Apple II Developer's Association notified us that Ralph Russo, a seven-year Apple veteran, has been appointed as director of the Apple II product line reporting to John Sculley through Don Casey. That's only one "layer" removed from the top, and this is the highest visibility that an "Apple II only" executive has had at Apple in several years.—DJD

---

*but (without ancillary hardware) cards that use four rows of memory (the **GS-RAM** and the **GS-RAM Plus** among them) may only support DMA transfers in the first four banks of expansion memory due to the design of the IIgs memory expansion slot. This corresponds to the first megabyte on the **GS-RAM** and the first four megabytes on the **GS-RAM Plus**.*

By expanding the memory used by the RAM disk to encompass everything except the four banks that can accept DMA transfers, the operating system will attempt transfers only into the compatible memory on the card and everything works again. If you add another megabyte of memory to the card (a total of six megabytes), you'll need to increase the size of the RAM disk to two megabytes in order to keep things working.—DJD

Don't hang up?

In your June issue you wrote about the new DMA SCSI card by Apple. Here are my two bits of information:

My configuration is an Apple IIgs with 2.25 megabytes of memory, a *TransWarp GS* in Slot 3, two 3.5 drives connected to the disk port, a Disk II controller in slot 6 and either SCSI card in slot 7. The hard disk is a Seagate ST177N with 40 millisecond average access time. It had a SCSI ID of zero at first. I boot a system configuration with 190K of desk accessories, the desktop picture, and "break cursor init" files.

With the Revision C card and the proper drivers installed, I got the times listed in column 1 of the table (all times are in seconds). This was with an interleave of 1:7.

	1	2	3	4
boot to Finder	35.6	40.3	39.9	31.2
AWGS (all modules)	24.7	23.7	18.1	N.C.
load 405K file	18.8	17.5	15.3	N.C.
AppleWorks 3.0	16.4	12.5	10.8	N.C.
330K database	25.8	36.3	23.6	N.C.

After changing to the DMA SCSI card and installing the drivers for that card, I got the results which are listed in column 2 of the table—still with an interleave of 1:7. Now I reformatted the drive with an interleave of 1:1—that was even worse than before. So I reformatted a second time with an interleave of 2:1—that gave the results in column 3 of the table. Then I changed the SCSI ID of the drive to 6 and got the results of column 4 ("N.C." is "no change" from column 3). But with a SCSI ID of 6 ProDOS 8 needs about 4 seconds to identify the drive— with the SCSI ID of 0 GS/OS took about 4 seconds before it started to boot. A further annoyance is that the card hangs the system if the drive is not on. You have to power up the drive if you want to boot from a disk. It needn't be ready; just powered up.

Udo Huth
Wittmar, W. Germany

We also have been using 2:1 on our hard disks. Some drives, such as the larger Conner or Quantum mechanisms, include a "track cache" in the drive that would probably allow the best performance at a 1:1 interleave (the Quantum drives will always use 1:1; they don't accept a change in interleave from the host computer). Our SCSI card is set to an ID of 7, and we have noticed that the drive does seem to be recognized faster when set to a higher SCSI ID (however, the difference we observed with ProDOS 8 was only a few tenths of a second).

*The "hang" you're seeing when the SCSI device is powered down is a "feature" of the true SCSI nature of the DMA SCSI card. The card supports arbitration (the ability to wait for a device that's in use to become "free"), which means if a device isn't ready when the card tries to identify the device, the card will wait. Part of the effect is that if the device is completely off-line, the card goes into an "infinite wait" situation. The **Cirtech SCSI Interface** (which also supports arbitration) react similarly; it's the trade-off for arbitration support.—DJD*

Cheap laser printer at last?

One of our morning papers is reviewing the new Hewlett-Packard *LaserJet III*. Among other things it speaks of a Macintosh version of the printer, which will come with an AppleTalk interface and a *PostScript*-compatible printer cartridge.

Could you please tell me if this printer, with such an interface, can be used with an Apple IIgs?

Peter Bourke
Waverley, N.S.W.

*Several "modular" laser printers have appeared on the market recently that combine a **PostScript**-compatible interpreter with an AppleTalk interface for use with the Macintosh. We don't have any direct experience with these printers yet, but if they are plug compatible with the Macintosh (that is, if the printer will plug into the Macintosh and work with the current Mac LaserWriter driver) then we see no reason why they would not work on the IIgs.*

*The lowest cost combination we have seen advertised is the Hewlett-Packard **LaserJet IIP** with the additional interfaces. If you are price shopping, also check the memory requirements for the conversion; you may find you need to expand the printer's basic memory to use the extra interfaces. If anyone gets such a combination working, we'd like to hear about it.—DJD*

A bit (and a character) lost

In *AppleWorks GS* the German umlauts are represented by 8-bit ASCII values. If you transmit a file containing such umlauts (or any other

8-bit ASCII value) either with *FreeTermGS* or the communications module of *AppleWorks GS* these characters are simply omitted from transmission. Do you or other readers know of a way to patch either *FreeTermGS* or the communications module of *AppleWorks GS* so that they are capable of transmitting 8-bit ASCII values?

Udo Huth
Wittmar, W. Germany

We got the same results in a test, but don't know of a cure. We hadn't expected the problem since ***AppleWorks GS*** *does save 8-bit values to a ProDOS text file when you request to save a word processor document as an ASCII text file.—DJD*

Wipe it clean

How can I "wipe" (using Glen Bredon's term) a volume from within a BASIC program? I am specifically interested in clearing my RAM disk from *ProSel* without having to boot up *Cat Doctor*.

Kelly J. Jones
Seattle, Wash.

The easiest way to "wipe" a RAM disk would be to check to see if the disk is empty (you'd usually, but not always, want to do this) and, if it's all right to erase, issue a ProDOS "format" call to the driver for the disk. For a normal disk, this might cause a time-consuming low-level format, but for a RAM disk the format will be quick.

To check to see if the disk is empty from Applesoft, you can BLOAD the directory and check to see if the file count is zero:

```
PRINT CHR$(4);"BLOAD /RAM,A$2000,TSYS"
IF (PEEK(8192+39) + 256 * PEEK(8192+40)) THEN
   PRINT "/RAM NOT EMPTY!"
```

The format call has to be performed from machine language using the ProDOS ROM code conventions described in section 6.3.1 of the ***ProDOS 8 Technical Reference:*** *there is no provision for formatting a ProDOS device from Applesoft itself.*

To issue the format call for a ProDOS block device, you can use the following sample:

```
        PHP             save status
        SEI             lock out IRQ
        LDA  #3         format command
        STA  $42
        LDA  #%10110000 slot 3, drive 2
        STA  $43        device for format
        STA  UnitNum     and for online
        LDA  #$00       define buffer
        STA  $44         at $1000
        LDA  #$10
        STA  $45
        LDA  $C08B      enable LC RAM
        LDA  $C08B       bank 1
        JSR  $FF00      (for /RAM only!)
        BIT  $C082      ROM back on-line
InstErr STA  ErrRetrn
        PLP             recover status
        RTS
ErrRetrn DS  1          for error code
```

Four zero page locations are used. $42 holds the ProDOS block device command (3 for a format). $43 holds the unit number; the bitwise definition is "DSSSxxxx" where "D" is "0" for drive 1 and "1" for drive 2 and "SSS" is a slot number from "000" (0 decimal) to "111" (7) and we've entered the binary equivalent for the /RAM volume as the slot 3, drive 2 device. $44 and $45 hold the address of a 512 byte buffer for use by ProDOS; we've picked $1000 arbitrarily. (We locked out interrupts during the format operation because the Apple II, II Plus, and unenhanced IIe monitor ROMs use location $45 during an interrupt.)

To format the device, we need to JSR to the address of it's ProDOS 8 driver. We can get the address of the driver by looking at the device vector table beginning at $BF10 in the ProDOS 8 global page; your program should provide for doing this after determining the correct location of the device you wish to format. The table consists of a series of (two-byte) addresses for the "drive 1" devices of slot 1 through 7, followed at $BF20 by the same list for "drive 2" devices. In our example /RAM is the slot 3, drive 2 device and its vector is at $BF24-5; the value there is $FF00. This location is inside ProDOS on the language card, so in order to access the driver we select and write enable the language card RAM, call the driver (performing the format), and then re-enable the motherboard ROM.

We can get away with this because we know the /RAM driver supports formatting. For slot-based RAM disks that have ROM-based drivers (not in the "language card" area), you don't need to switch the language card space, but you do need to check the ROM identification bytes for the device to insure it is an "intelligent" disk controller that supports the format call. To do this, verify that (for a device in slot "n") that $Cn01 equals $20, $Cn03 equals $00, $Cn05 equals $03, and $CnFF equals $FF (notice that the Disk II interface has a $00 here, and does not support the intelligent controller features). If all these match, then the status byte at $CnFE will tell you if the device supports formatting (bit 3 will be set if formatting is supported)

The only way we know of to "erase" a disk volume without actually formatting it (or manually deleting each file on the disk) is to use block-level commands to read the volume directory, purge all file entries, re-write the "blank" directory, and then use a similar sequence to update the volume bitmap's record of used space on the disk. That's beyond the scope of something that we can duplicate here because it would require explaining the directory and bit-map structure in detail, but the structure is explained in Appendix B of the ***ProDOS 8 Technical Reference.***

*For disk devices installed under GS/OS, there is a GS/OS specific command (**EraseDisk**) which handles "erasing" versus "formatting" a device.—DJD*

Don't touch that memory!

I'm developing a potentially large utility program that is supposed to run concurrently with BASIC.System and Applesoft. In examining my real estate option (looking for an out-of-the-way place for the code to live) I'm thinking that the upper reaches of auxiliary memory would work pretty well. However, the memory map in the *ProDOS 8 Technical Reference* manual (page 119, March 1987 printing) declares the area from $E000 to $FFFF, and most of one of the $D000 banks, to be an "Other used or reserved area".

This doesn't necessarily deter me despite the fact that my knowledge in the area is limited, as the only system use of this memory range that I can think of is by the ProDOS /RAM volume, which is certainly expandable. (In fact, I seem to recall Beagle Bros *Extra Variables* program using this area quite freely).

Can you think of anything that happens in this memory which would make it a no-no to use?

Eliot Lifeson
Fair Lawn, N.J.

There are two reasons you shouldn't use memory (or other system resources) Apple has marked "reserved". First, it may already be in use by a product that has arrived since the last printing of your reference manuals; for example, part of that auxiliary memory language card space is reserved for AppleTalk support. While this may not seem of immediate consequence to you specifically, writing software that is incompatible with AppleTalk will be a factor for others. (We are sensitive to this because we've had to abandon more software for AppleTalk incompatibilities than any other reason.)

Second, Apple reserves resources so that they will have a designated place to expand the system when necessary without "breaking" current software. If Apple had to expand the size of ProDOS or BASIC.System by using main memory, for example, everyone who had programs pushing the edge of available memory would scream. All this is somewhat tactfully explained in ***ProDOS 8 Technical Note #26,*** *"Polite Use of Auxiliary Memory".*

That /RAM disk itself is an acceptable place to expand, and there is even a prescribed method for grabbing its space. Section 5.2.2 of the current ***ProDOS 8 Technical Reference*** *Manual documents the conventions, including sample source code.*

ProDOS 8 Technical Note #8 *mentions that ProDOS 8 versions prior to 1.2 do not have the capability of removing the volume control block (VCB) which ProDOS allocates internally when /RAM is installed. This means that if you disconnect /RAM, you will only have the capacity for a maximum of 7 volumes on-line as opposed to the 8 volumes ProDOS normally supports. This was fixed in v1.2 and you can free the VCB by performing an on-line call to the device* ***after*** *disconnecting it; MLI error $28, "No device connected", will be returned, and the VCB will be freed.—DJD*

Squashed mouse

Why is my mouse disabled after entering and returning from the Control Panel? The program doesn't seem to make a difference.

Andrew Wehling
St. Louis, Mo.

I don't know, but my ROM 03 IIgs also does the same thing and it is very irritating. It also points out a weakness in Apple's Human Interface Guidelines: when the mouse dies and you try to quit without destroying the work you did in the previous session, the only keyboard options you usually have to deal with the current document are "Return" (to save the new copy) or "Escape" (to return to the program). If the mouse dies when the document in memory is in a worse state than the one on disk, the only option is to use reset to reboot, which (considering you might catch GS/OS's cache operation in a bad state) isn't exactly ideal, either.—DJD

DIF conversions

I am looking for somebody who could have made an adaptation of your program from "What's the DIF" (***A2-Central***, April 1989) in order to retrieve *PFS:Plan* data in DIF format. (In order to import data into AppleWorks, for instance.)

J. P. Busquet
313 Rue Lecourbe
Paris 75015
France

Large print

Is there any graphics program out there that allows you to draw, edit, and print large format graphic images (ones that would require several normal 8.5 by 11 pages to be pieced together once the printing is done) without sacrificing resolution; in other words, without just doubling, tripling, etc., pixel size? I own a IIgs and an ImageWriter II.

J. R. Ginnow
Middletown, R.I.

In order to scale a drawing without resolution loss, the graphic has to be defined mathematically (with equations describing curves and so on) rather than by a "bit map" of the image. The IIgs (and Macintosh) each contain the ability to store such definitions using their respective Quickdraw commands.

A QuickDraw-based printer driver can accept these commands directly to generate the final document, scaled to the size of the print area. But QuickDraw printers are rare; a more common solution is to use a "computer independent" page-description language like ***PostScript****, such as the LaserWriter uses. The IIgs (and Mac) LaserWriter drivers convert the QuickDraw description of the document into* ***PostScript*** *commands. These can draw on any* ***PostScript****-compatible device using the device's inherent resolution (unless the image being generated is a bit-mapped image to begin with). For example, we print the proofs for this newsletter on a LaserWriter with 300 dots per inch (DPI) resolution, while the printer masters are generated on a (****PostScript****)* ***Linotype*** *phototypesetter at 1200 DPI (versus the LaserWriter's 300 DPI) using the same source file. If you had a large enough* ***PostScript*** *printer, you could print the document any size you liked.*

A super-large printer is impractical (the cost would be prohibitive) for most users, so the alternative is to divide the document into "tiles" and print each tile separately. After printing, the tiles" could be assembled into the complete image. The normal printer drivers do not handle this type of printing themselves, but MECC (3490 Lexington Ave. N, St. Paul, Minn. 55126, 612-481-3500 or 800-228-3504) sells a program called ***Designer Prints*** *that can import graphics and print images up to 10 by 10 (8.5 by 11 inch) pages in size using your selected IIgs driver. It is copy-protected, and requires GS/OS and at least one megabyte of memory.—DJD*

BASIC networking

I am the District Computer Coordinator and need some help with an AppleTalk network. I am trying to write several programs in BASIC to make the network work better. One of these is to get rid of the temporary files that are left behind when students do not log off the network. I have the program written but it will only work on legal file names. Since the files and the names are kept on our Mac SE/30, there are names that are not legal ProDOS names. Spaces and other illegal characters are allowed and converted to "?". Is there any way that I can delete these files without going to the block level of the hard disk? Could I find the place where the prefix of the disk and directory is stored and change it in memory? If I did this would ProDOS examine the prefix as well as the file name when I delete it? The following is an example of the type of file that I am looking to get rid of:

```
/HS/USERS/<ANY.USER>/SEG.A10000.34
/HS/USERS/DANIEL DILL/SEG.A10003.2
```

The illegal part of the name is always the second last part and never the last portion of the name. Any suggestions?

Daniel Dill
Newton, Wisc.

If you are on a IIe locked into ProDOS 8, there isn't anything you can do about the filenames from Applesoft (and ProDOS) since the illegal characters won't make it past the operating system. It would require using the ProDOS MLI AppleTalk command ($42) and talking to the server directly through the Apple File Protocol documented in ***Inside AppleTalk*** *(Addison-Wesley) and* ***The AppleShare Programmer's Guide for the Apple IIgs*** *available from APDA (don't let the title throw you off; the ProDOS 8 stuff is in there, too).*

But the "easy" solution is to do your file administration from a IIgs, and using GS/OS (rather than ProDOS 8) to delete the unwanted files. GS/OS uses the AppleShare File System Translator (FST) to communicate with the AppleShare volume, and the FST and GS/OS are prepared to deal with the Macintosh filenames. Assuming you have adequate access privileges to do it, you can simply issue a GS/OS "Destroy" command to remove the offending file.

If you have control over the assignment of user names, you can eliminate some of the problems by asking users to enter their user names as valid ProDOS names; for example, "Dennis.Doms" (with a period) rather than "Dennis Doms' (with a space that ProDOS can't digest). Using the ProDOS-legal form will create user folders that are accessible from ProDOS 8, since the user name is used in the pathname. That will help for all except the "<Any User>" designation that the server defines (we haven't found a way to alter it).—DJD

How much memory?

How many bytes equal one megabyte? I have two conflicting theories.

Method 1: 40992 blocks (number of blocks on my 20 meg hard disk) times 512 bytes per block equals 20,987,904 bytes, divided by 20 bytes per megabyte gives 1,049,345.2 bytes per megabyte.

Method 2: 1024 bytes per kilobyte times 1000 (to find bytes per megabyte) gives 1,024,000 bytes per megabyte times 20 gives 20,048,000 bytes (for 20 megabytes) divided by 512 (to find the number of blocks) gives 39,156.25.

What's wrong?

Rick Koch

There are 1024 times 1024 or 1,048,576 bytes per megabyte. (The 1024 comes from the power of two that is closest to one thousand; two to the tenth power equals 1024.) So we'd calculate your hard disk capacity this way: 40992 blocks times 512 bytes/block equals 20,987,904 bytes, and dividing that by 1,048,576 gives us about 20.02 megabytes. Don't worry about the slight difference; the hard disk may not be exactly 20 megabytes in size.—DJD

RAM chip designations

Please explain the terms "dynamic RAM" and "static RAM", and the numbers "64K x 1, 120 ns", "64K x 4, 100 ns", etc.

I'm guessing the "xxx ns" refers to how fast (or slow) the chip operates or can be accessed in nanoseconds but the "x 1", "x 4", is Greek to me.

If I were to buy a 0K memory board for my IIe or IIc, how would I know or determine which memory chips to buy?

S. Dinnel
Santa Maria, Calif.

"Dynamic" and "static" refer to the RAM chip's ability to retain information. Either chip will accept and retain data in normal operation, but dynamic RAM's will "forget" the information stored in them unless the information is accessed within some minimum time frame. To make sure the information isn't lost, systems using dynamic RAM must supply a "refresh" signal to the chips within the specified interval to preserve the information. Static RAM will retain the information without requiring any "refreshing" of the data. When needed, the refresh is implemented in hardware; all of this is transparent to you as a user.

The numbers describing the chip are the size of the chip in bits followed by the number of bits that can be accessed when an address is supplied to the chip. For example, "x 1" means that for a specific address only one bit can be accessed, "x 4" means four bits can be accessed. To get the proper number of bits for your computer's processor, the chips can be wired in parallel; for example, wiring two "64K x 4" chips in parallel would allow you to access 64K bytes (a byte is 8 bits) of memory. Half (four bits) of each byte would come from each of the two chips. Again, all of this is implemented in hardware.

As you suspected, the speed value tells you the fastest rated speed of access for the chip in nanoseconds (billionths of a second). You can usually substitute a faster chip for a slower one, but not the reverse; for example, a 120 ns chip should work where a 150 ns chip is specified, but not the reverse.

Many of the chips look alike, so you can't tell by looking at the sockets on an "empty" memory card what types of chips it uses. The information supplied with your memory card should tell you the proper type and speed of chips to use with it, and that's what you should refer to.—DJD

Documentation dilemma

I've some experience I would like to share with you and your readers.

Recently I attempted to publish some weaving software, but I made two major errors. I released the templates without the author's final approval and I did not supply sufficient documentation which, under agreement, I was to provide. The first mistake was stupid, it was

irresponsible of me to risk another person's reputation. The second mistake, the lack of documentation, is what I wanted to talk to you about.

It seems that I did not have sufficient understanding of the processes involved in the weaving processes that use the templates' answers and I now understand that I do not want to do that much homework. I told the author so and I have declined to publish. So we will be friends again and the templates will get a rest.

It seems that in the Apple II kingdom there are many fine programs without adequate documentation and the lack of documentation of some programs is downright infuriating. I'm talking about successful programs by clever programmers. Of course, this goes with the territory. The Apple II is like a giant erector set and each person can create astonishing new tricks with the billions of on/off switches without having to apologize for their status in life. I bought into the II because I'm a techno-nut (my past career was as Chief Engineer in the Merchant Marine), and I don't want to be spoonfed by a Mac or have to deal with the hyperbole of MS-DOS. Unfortunately, some programmers seem to think that the rest of the world should understand what little they cryptically say and their documents reflect this snobbism. Sadly, at one of the nation's top engineering schools located near me, the students (and not a few faculty) regard English as a second language.

I wish that the guilty programmers and publishers would take the time and efforts to ensure that their products do have thorough documentation and make sincere efforts to address that specific problem. I know now that they work under tremendous pressures but that is no excuse for leaving a job half done. I would sooner wait for the finished product than get a "kit" program. From now on I will discontinue using those packages whose documentation is the pits; where the author assumes that I can intuit meaning from code words; where the index, if it's got one, is only jargon.

Let me not leave on a sour note. My thanks to all of you who do go the last mile and beyond. Your efforts should not be taken lightly and my hat's off to you and I will patronize you.

Quentin Packard
Troy, N.Y.

You have hit on part of the problem in your letter: good documentation is hard work. Some companies may find themselves in the same predicament you did; they have the program completed, but are faced with investing more time and money in preparing the documentation. If the company has a financial need to get the product to market quickly to generate operating revenue, the documentation may be the part of the package that "slips". The "chicken and egg" syndrome here is that better documentation may make the product sell better, but the company may question the need to invest a large sum in the manual until they know the product will sell enough to justify the extra expense.

Translating a highly technical concept to "plain English" is also sometimes difficult. Many computer users claim to have an aversion to "jargon", but it does make sense that a certain amount of a new vocabulary will have to be learned in order to cope. One problem we've seen is that some users want to be "spoonfed" when there ***is*** *a learning process involved that involves some effort. It may even be difficult, though good documentation (and a good user interface design) should make it as painless as possible.*

I'll admit to finding it hard work to write at an "end user" level; I didn't skimp on English courses in high school and college even as a science major because it was emphasized to me that the ability to communicate your ideas is at least as important as the ability to formulate them. The problem I perceive is that the longer you work in a field the more inured you become to the jargon; sometimes you use a term without realizing that it's meaning isn't plain to everyone. Editing your own writing is a rather sobering experience, because you have to make the additional effort to avoid that trap.—DJD

Lost art?

Something that seems to be lost of late in the Apple II world is all the amateur programming enthusiasm there used to be back when Applesoft was the thing. That alone used to sell a lot of Apples. Seems like we need a new folk programming language that will access the power of the new machines but also be simple enough for everyday users. Not that such a language doesn't exist, but no one seems to have embraced it and made it popular.

Martin D. Paquette
Sebastopol, Calif.

For the Apple II, Applesoft became the standard because it was included with every machine (not an insignificant point) and it was easy to learn. Another factor was that, for a time, Apple included the ***Apple II Reference Manual***, ***Applesoft Tutorial***, ***The DOS Manual*** *and* ***Applesoft BASIC Programmer's Reference*** *with the Apple II. This stopped with the IIe, though Apple did start supplying a short introduction to Applesoft (****A Touch of Applesoft BASIC****) with the IIgs when it was introduced in 1986. But there was no native IIgs language supplied with the system; Apple elected to let the market decide.*

*Apple did release a IIgs development environment contracted from The Byte Works (****Apple Programmer's Workshop****) with an assembler; The Byte Works, in turn, sells a substantially similar product as its* ***ORCA/M*** *assembly language development system. Apple has also released tools to "enhance" this environment, such as the resource compiler and decompiler* ***REZ*** *and* ***DeREZ*** *(part of the $50* ***Programming Tools and Interfaces for APW*** *package from APDA, part # A0228LL/A). APW Product Manager Tim Swihart revealed in a GEnie real time conference on May 28 that the anticipated version 2.0 of the environment will be released solely as a Byte Works product; Apple will continue to supply IIgs development tools, but they decided having two forms of essentially the same environment on the market (one labelled* ***APW****, one labelled* ***ORCA/M****) was not desirable.*

Therefore, it is left mostly to potential programmers to decide what languages to use. We've found that many potential programmers get confused just deciding which language to use. We're not immune to the confusion; one of our obstacles in trying to cover IIgs programming is that we have to decide what language to present the program in, and it's likely we'll have to make some effort to teach elements of the language, because it will probably not be BASIC. IIgs programming uses a ***lot*** *of data structures which BASIC is not able to handle conveniently. (My personal choice would probably be C, with Pascal a close second.)*

The other hurdle is that the environment surrounding personal computers has changed remarkably in their lifetime, from primarily a "hacker" audience to a "user" audience. Much of the user audience is unconcerned with the mechanics of the system. And the truth is that we get relatively few programming questions these days, and the ones that we do get are often complex or obscure enough that we're not sure what the general interest in our reply would be.

We may "experiment" with a sample IIgs program or two in the next few months to see what the interest is. We'd like feedback on the articles, good or bad, when they appear.—DJD

Which way did he go?

Any chance of a regular editorial column by Tom Weishaar?

Isaac Molho
Garden Grove, Calif.

Tom slips in an article now and again (such as "Personnel moves rock Apple" in the March 1990 issue); keep watching.

Currently, Tom is working on trying to get our mailing list system ported over to GS/OS as well as implementing an internal electronic mail system. Tom says by the time he's done, he'll probably be sick enough of programming to look forward to more writing.—DJD

A2-Central™

[redacted]

Edited by:

Dennis Doms

with help from:

Tom Weishaar	**Sally Dwyer**	**Dean Esmay**
Joyce Hammond	**Jeff Neuer**	**Jay Jennings**
Tom Vanderpool	**Jean Weishaar**	

A2-Central,—titled ***Open-Apple*** through January, 1989—has been published monthly since January 1985. World-wide prices (in U.S. dollars; airmail delivery included at no additional charge): $28 for 1 year; $54 for 2 years; $78 for 3 years. All back issues are currently available for $2 each; bound, indexed editions of our first four volumes are $14.95 each. Volumes end with the January issue; an index for the prior volume is included with the February issue.

The full text of each issue of ***A2-Central*** is available on 3.5 disks, along with a selection of the best new public domain and shareware files and programs, for $84 a year (newsletter and disk combined). Single disks are $10.

Please send all correspondence to:

A2-Central
P.O. Box 11250
Overland Park, Kansas 66207 U.S.A.

ISSN 0865-4017

Printed in the U.S.A.

GEnie mail: A2-CENTRAL
Voice: 913-469-6502
Fax: 913-469-6507