A2-Central™

formerly **Open-Apple**

August 1992
Vol.8, No.7

ISSN 0885-4017
newstand price: $3.00
photocopy charge per page: $0.20

A journal and exchange of Apple II discoveries

Feeling synful

by Dennis Doms

One of the strongest features of the Apple IIgs, one that other computers like the Amiga or the Mac still can't match, is the capability of its Ensoniq sound synthesizer chip.

Apple has kept this capability well hidden. The beautiful sound of the Ensoniq was somewhat hampered by the lack of stereo output (though stereo can be added to the IIgs inexpensively) and a pretty pathetic excuse for a speaker (the internal one). To complicate matters, Apple Corp (the record label spawned by the Beatles) and Apple Computer became embroiled in a lawsuit over the use of the Apple name in conjuntion with the IIgs; Apple Corp felt this was dangerously close to use of their trademarked name in a musical context and therefore an infringement.

Recently the suit was settled; the terms were not disclosed publicly. Whatever the agreement, apparently Apple felt the situation had cooled enough to release a strong acknowledgement of the sound capabilities of the IIgs. The result was an extra disk added to the distribution set of System 6: synthLAB.

synthLAB is actually an application that demonstrates the MIDI Synth toolset. "MIDI" stands for Music Instrument Digital Interface, which is a standard in the music industry for communication between musical instruments. As supplied, synthLAB lets you play prerecorded songs that are actually sequences of MIDI instructions saved in disk files. But there are hidden capabilities that can be further realized by adding a sound digitizer (to record new instrument sounds) or an external MIDI device such as a keyboard (to play back prerecorded songs or to play and record new songs into synthLAB).

However, even those who lives revolve around instrumental music ~~may find that discovering~~ what MIDI Synth and synthLAB can do is a lot like solving a mystery. First, realize that the capabilities of the IIgs are actually twofold: the IIgs can be a sound synthesizer controlled *by* an external device or the IIgs can be a MIDI controller *for* an external device. The reference document supplied with synthLAB provides a good funtional overview of using the program but refers to other documents for most of the underlying theory of operation. To understand the IIgs's sound capabilities (and what synthLAB can do with them) you need to know a little about what a synthesizer is. And, on the other hand, to understand what synthLAB can do with an external MIDI device, you have to know a bit about MIDI. In both cases, it turns out the concepts are complicated enough that a "bit" is still a lot; we'll try looking into synthesizers this month and MIDI (as well as some other sound software) next month.

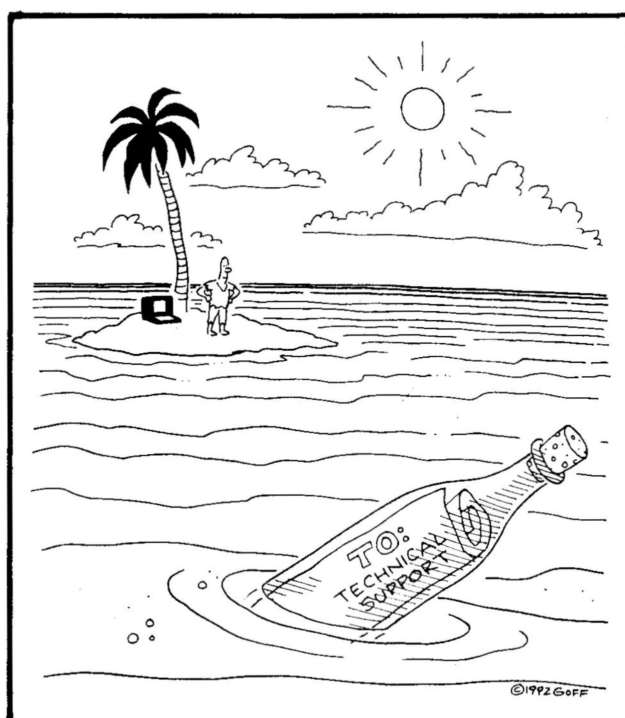
As a musical instrument, a synthesizer is an electronic device capable of creating a wide variety of sounds suitable for musical use. The sounds may be intended to mimic familiar instruments (flutes, percussion, organs) or less familiar sounds (such as a conceptual "UFO landing") created "from scratch".

A musical tone can be divided into three component qualities. One is *timbre*, the general way the tone sounds (timbre is what makes a steady flute tone sound different than a steady saxophone tone). Another is *pitch*, whether the sound is high (think "soprano") or low (think "bass"). Finally there's *amplitude*, how loud or soft the sound is.

Most synthesizers are designed as voltage-control devices. A voltage is varied over time and is combined with other voltages in the synthesizer. In order to create sound, the sum of all the interactions is eventually fed into an electromechanical device that changes the output voltage into moving air that the ear can detect—the familiar loudspeaker. The manner in which the speaker moves (and the sound it produces) is determined by the way the voltage changes.

The feature of this dynamic voltage that determines the timbre of a sound is the *waveform*, or the manner in which the voltage varies repetitively over short periods of time ("short" is a relative term, but the audible range is usually considered to be from 20 to 20,000 repetitions a second). Some of the common basic waveforms used in synthesizers are sine waves, triangle waves, and square waves (and variations sometimes called "pulse" or "rectangular" waves) (see Figure 1, next page). Each of these waveforms has a distinctive sound when converted to an audible form.

A sine wave causes a relatively smooth back and forth motion of the speaker with a steady pure tone. A triangle wave involves abrupt changes of direction that cause a more "raspy" sound. Square waves, where the speaker is abruptly thrust from one extreme to the other, are even raspier. You may think "raspy" is bad, but an unadulterated sine wave is such a pristine tone that it doesn't have much to interest



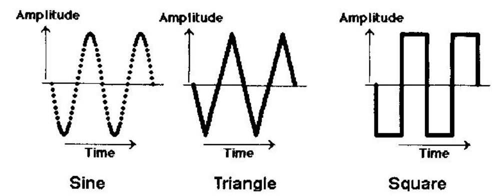


Figure 1. Some simple waveform shapes.

your ear; as a way of comparison, you would probably soon get tired of seeing pictures painted only in pastels.

Actually, all complex waveforms can be expressed as the sum of a series of sine waves (but we're talking *a lot* of sine waves), that give the sound more character. Adding waveforms to create new waveforms is called *additive synthesis* and is commonly used by many synthesizers.

Pitch is an easier quality to categorize physically; it's the frequency of repetition of the waveform. Musicians assign letters to pitches, 440 cycles a second is an "A" below middle "C". But in order to avoid music theory, let's just say that the faster the waveform repeats, the higher the pitch of a note; a soprano sings at a higher pitch than a bass.

Amplitude is easier yet, it reflects how loud the sound is. The greater the voltage variation of a waveform the more movement the speaker undergoes and the louder the generated sound is.

If all you needed to know to duplicate a sound was its waveform, frequency, and amplitude, the most complicated part of the synthesizer would be the waveform generation. But this capability has been around for a while; organs (electronic or not) usually have the capability of simulating these three qualities fairly well; this is why you'll see organ controls often labeled with the names of other musical instruments even on "antiques."

What makes a synthesizer special is a catch not mentioned in the above description: these three qualities often don't stay constant over the duration of a sound. This is true especially if there is a "percussive" element to the sound; some action that varies the quality of the sound depending on how vigorously it is initiated or maintained. When you strike an organ key the sound usually starts abruptly and maintains itself; this is fairly unusual for a musical instrument.

The piano, which creates sound by striking a string to cause it to resonate, was always a tricky instrument to duplicate with a synthesizer. When you strike a piano key, the sound begins and sustains until you release the key, at which point it dies out. But there is a lot of variation available to the piano player, who can strike the key quickly or slowly (forcefully or gently), let it sustain only a short time or as long as the key is held down (or longer, by using the sustain pedal), and who can release the key quickly or slowly. The vibration of the piano string producing the sound also changes subtly over time, making a realistic piano hard to duplicate.

Synthesists refer to the way the sound changes over time as the *envelope* (think of it as enclosing the shape of the sound). Synthesizers have to incorporate a way to simulate these changes and usually do it by including envelope generators that shape the sound over time. The envelope also has to be synchronized to the generation of the sound and may control combinations of the amplitude, timbre, and pitch, though the amplitude is often the most noticeable quality controlled.

A common model is the ADSR (attack, decay, sustain, release) envelope. (A sample might look something like Figure 2.)

The "attack" phase is the initial response, usually defined by the amount of time the envelope takes to reach full voltage after the sound is initiated. The "decay" phase is usually defined by the amount the envelope voltage is allowed to change over time (usually to a lower amplitude). The "sustain" is the level the voltage will maintain if the sound is sustained. The "release" is the time it will take the voltage to drop to zero (silence) once the sound is no longer being sustained by the operator.

Since the "operator" may be something as familiar as a human depressing a key or as mechanical as a controlling signal coming from a computer, the variations in the possible implementation of a synthesizer can be very complex. But this should outline enough of the concepts to take a peek at how synthLAB uses the Ensoniq chip in the IIgs to generate instrument sounds.

Into the digital domain. The operation of the Ensoniq varies from our theoretical synthesizer in one major respect: the waveform is generated digitally rather than by summing a series of waveforms. But it can be thought of as a more complicated version of the model.

The Ensoniq chip functionally consists of 32 oscillators that can produce sound from waveform information. The digital waveform is defined as a series of (byte-sized) values representing effective amplitudes from -127 to +128 (there are also some other Ensoniq complexities, too primitive to control from synthLAB, that I'm going to avoid here). Don't panic at the "limitation" of 8-bit values in our increasingly 16- and 32-bit computer world. Eight-bit synthesis is commonly used in both electronic musical instruments and as the basis of most computer sound cards (and 16-bit sound cards just now appearing are not cheap; one for an MS-DOS PC sells for $799 at our local computer superstore). The IIgs is still ahead of most of the field.

It is possible to construct a wavetable for a complex waveform by simulating additive synthesis, adding (literally, though averaging may also be employed) together the synchronized amplitudes of simpler component waveforms (sine, square, and so on). Some musicians accustomed to "programming patches" (designing instrument sounds by changing synthesizer parameters) might like to see a such a "virtual synthesizer" created on the IIgs. But instead of trying to create a saxophone sound from scratch, these days some musicians prefer to record what they feel is an appropriate saxophone sound directly into a wavetable (a technique called *sampling*) and then fine-tune the sound by varying the envelope parameters. This is the way most IIgs users will probably approach the Ensoniq — using programs that employ sampled waveforms to create the "instruments" to be used. (But since it is a computer, it would be no great leap to let the IIgs simulate additive synthesis, too.)

Using digitized instruments, the timbre is defined by the waveform as represented by the table. The pitch is varied by changing the speed

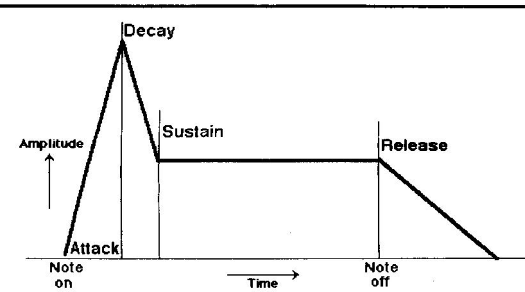


Figure 2. A simple ADSR envelope.

at which the waveform is played; if you run through the sets of values faster, the pitch of the note played will be higher — just as speeding up a tape recorder on playback makes voices sound squeaky. The amplitude is determined by the maximum and minimum values within the waveform table; an average range of +64 to -64 is softer than +100 to -100.

synthLAB doesn't let you modify the wavetable itself, but it does allow combining the oscillators to use multiple instruments through the underlying MIDI Synth tool. As designed, MIDI Synth can utilize up to seven instrument voices at one time. Each voice consists of a pair (for stereo) of sound generators controlled by an envelope generator. MIDI Synth can allocate instruments dynamically so synthLAB does allow defining up to 16 instruments for use within a song, but no more than seven of these can play at the same time.

synthLAB's functions are divided into three working screens. When you first enter the program (after dispatching the "splash" screen that plays the introduction) you see the sequencer screen; we'll spend more time on it next month. The other two screens are what we're after this time; they allow playing with the sounds synthLAB uses.

The structure of the synthLAB Envelope Edit and Wavelist Edit screens are based on the underlying structure of the MIDI Synth synthesizer. (At this point you may want to grab your synthLAB disk and follow along through the on-disk manual or within synthLAB itself.)

There's an array of three large buttons at the upper left of the synthLAB screen just under the menu bar that selects the function (sequencer, wavelist edit, envelope edit) you want to work with. The second button (the button that looks like a sine wave with an arrow underneath it) takes you to the Wavelist Edit screen.

Just so we're all playing from the same sheet of music, so to speak, pull down the File Menu, select "Load instrument..." and load the instrument file called COMBO.BNK. Next, pull down the Instrument menu, and you'll see the 16 instruments in this file. Select the first one, "Drum Kit 1." Now click on the keys on the small keyboard on the screen. If you click on the right, near the center, and on the left, you'll hear three different drum sounds.

Now punch the button at the lower right of the screen that's called "Top Key." This brings up a long keyboard and eight sliders. The sliders control which of eight "wavelists" the keyboard uses for each key. For example, if you pull the top slider all the way to the right, return to the previous screen, and click on the keyboard in the same places as before, you'll hear just one drum sound in a variety of pitches. By moving the top slider, you told MIDI Synth to use "wavelist 1" for all the keys. The "Top Key" screen allows you to use up to eight wavelists with one instrument; the one that actually plays is controlled by its position on the keyboard. This gets very complicated very quickly, which is probably why most of the instruments supplied with MIDI Synth have that top slider moved all the way to the right and use only Wavelist 1.

So pull down the "Wavelist" menu and make sure Wavelist 1 is selected. Below the small keyboard is a switch labeled "Gen 1" and "Gen 2". Each Wavelist has two sound generators associated with it. Let's start with Gen 1.

Below the generator switch is a diagram of the oscillator configuration for the generator. There are six possible configurations; clicking on the current configuration steps to the next one in sequence. The manual that comes on the disk with MIDI Synth calls these Type 0 through Type 5. With Type 0, each oscillator (A and B) plays over and over ("loops") at the same time. For Type 1, oscillator A plays through once (a "one-shot") while B loops. For Type 2, both play straight through once. With Type 3 the two oscillators play sequentially; first A plays once, then B plays and loops to itself. Type 4 has each oscillator playing once in sequence, A then B. Type 5 has A play, then B, then (going by the sound) looping back to the start of A to loop both.

The significance of these configurations is more apparent when you think about how a sound might be constructed, which we'll do in a moment. First, though, notice that at the upper right there are two rectangular boxes labeled "Wave A" and "Wave B." If you press down on one of these boxes, you'll get a popup list of the waves in the current waveset. Under the File menu there's a selection called "Load Waves..." that lets you change these ("Load Instruments..." also changes them.) Wave A is used by Oscillator A and you can control its pitch and volume with the sliders on the left. Likewise, Wave B plays through Oscillator B and is controlled with the sliders on the right.

By convention, waves designed for looping are designated by a name beginning with a tilde ("~"). Many sounds have a quickly varying "attack" wave followed by a different sustained wave. Imagine the word "So". The initial "ess" consonant sound would be followed by a steadier (and different) sustained "oh" sound for the vowel. Let's assign the "ess" sample of the attack to Wave/Oscillator A and then put the sustained "oh" sound in Wave/Oscillator B. Playing these in the Type 4 configuration makes the two waves play sequentially to form the complete word "So". We could have put the entire sample in one wave, but if we do that there's no way to vary the length of time the "oh" is sustained to say "Soooooo" instead. Using two waves together allows us to use the same generator to say the word "So" recognizably over different stretches of time.

Consider a similar example using a piano. The sound of a piano note usually starts percussively as a result of striking a key, then dies down slowly at a more steady tone. The initial keypress would be the attack assigned to the Wave/Oscillator A and the sustained sound would be assigned to Wave/Oscillator B. This doesn't complete a realistic piano sound (we still need to shape the periods of attack and dying down with an envelope generator) but it does give us the two most important waveforms in the life of a piano note to work with.

The other oscillator configuration types are needed to mimic other sounds. For example, a sound that cycles periodically as if "throbbing" can be created by sampling the "throb" and looping that portion of the sound over and over while it is sustained (the Type 3 configuration).

The sliders include three tuning controls of different sensitivity; those for Wave/Oscillator A on the left and those for Wave/Oscillator B on the right. The "Octave" control will cause the greatest change; "Semi" is moderate; and "Fine" allows the finest adjustment. Tuning controls are needed to adjust the basic pitch of the tone to correlate to the proper pitch when played by the synthesizer. Frequently, a sampled wave is not recorded at the exact proper pitch. You could also use these sliders to "detune" the oscillators individually for musical effects (the familiar "honky tonk" piano is one that is effectively out of tune giving a more "imprecise" feel). For very subtle detuning there is a special "Detune" control at the lower center of the display.

Below the wave names are volume controls for the playback volume of each wave. These are used to adjust the relative loudness of the two waves. (Adjusting the volume of complete instruments is better accomplished by the "Volume..." item of the "Setup" menu.)

Below the volume controls is the Top Key button. Some instruments do not maintain a constant timbre throughout their playable range. The MIDI Synth tool allows the construction of a multiple-sampled wavelist that records several different waveforms at various pitches to get a better overall synthesis of the instrument characteristics; the "Top Key" parameters are used to determine what the cutoff point is for the use of each waveform.

There's also a control named "Stereo" that varies the stereo "channel" for the sound. If you have a stereo card, this determines where the sound is placed in the stereo image. "0" is all the way toward one side, "7" all the way toward the other. Whether one or the other is left or right depends on your your stereo card maps the sounds and where you have your speakers positioned. (Not all programs seem to agree on which is left or right, either.)

Before we go on, let me remind you that MIDI Synth allows you to assign four Wave/Oscillators to a single instrument sound — two for each sound generator. Go back to the instrument list and pick "Slap Bass". Now click on the Gen 1 and Gen 2 buttons and you'll see that Wave/Oscillator A is using a different wave in each sound generator.

The envelope edit screen. Pressing the third button in the "function" panel takes you to the envelope editor. Again we have a switch to select which generator we're looking at, but most of the screen (the whole right-hand side) is occupied by a plethora of adjustment controls.

The sophistication of the MIDI Synth envelope generator model puts the simple ADSR model to shame; instead of four adjustable parameters, we have 13. Envelopes are easier to visualize graphically than in words, so lets start off with a picture (Figure 3) of a proposed envelope. We'll also stick in our basic ADSR points of reference.

In most cases we're defining the level (amplitude) that the envelope will reach at a certain time and the rate it will assume to get there. We refer to "rate" (change per unit time) as opposed to an absolute time because the interval may not be constant. Two events are monitored for triggering two specific cycles in the envelope's life: the "note on" (key down, if we believe a key depression is triggering the envelope) event starts the envelope program running from the attack through the three decay periods and holding at the sustain point. The sustain is "untimed"; it ends when a "note off" (key released, if we stick to the keyboard model) event is received. At this point, we move into the portion of the envelope defined by the three release periods.

synthLAB's screen divides the adjustments into two horizontal rows; the lower row defines the time (or rate) quantities for various points in the envelope, the upper row defines the target volumes. "Atk" in the lower row determines the attack rate for the initial attack; above it "Vatk" defines the target volume. Continuing in a zig-zag pattern between the lower and upper rows of sliders we follow the stages of the envelope through the three decay rates and their two intermediate target volumes ("Dk1", "VD1", "Dk2", "VD2", "Dk3"), the sustain level ("Sus"), and then the three release stages and two intermediate volumes ("Rel1", "VR1", "Rel2", "VR2", "Rel3").

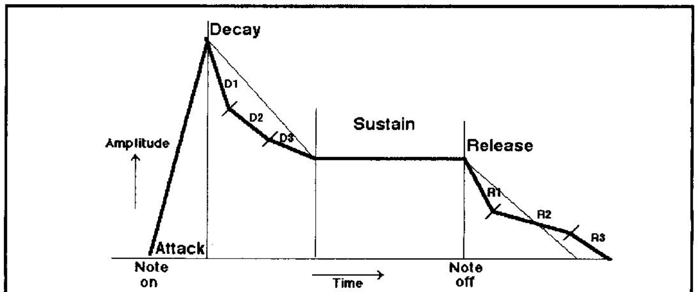


Figure 3. MIDI Synth's intricate envelope.

Envelope generators on some synthesizers can be used to control sound characteristics other than the variation of volume over time but in most cases they are used as a MIDI Synth envelope is: to shape the percussive nature of the sound over time. We've drawn an envelope that roughly resembles what we would expect for a hard-hit piano note: a sudden increase in the volume of the note followed by a sharp decrease (the "plink" you hear when a piano key is struck sharply) then a slower decrease (as the piano string loses some energy and volume) into a moderate sustain as the string reaches a comfortable level where it can resonate relatively constantly for a while. When we release the key, the sound is muted rapidly down (through the three release periods) to zero. Adjusting the envelope affects the way the sound is expressed over time; if you wanted a "legato" (stretched-out) sound resembling a gentler, slower, striking of a piano key you could decrease the attack and release rates and lower the "peak" attack volume.

In fact, MIDI Synth can react to these "touch responsive" characteristics of a piano keyboard and simulate the effect by adjusting the envelope dynamic response to reflect it. Two adjustments on the envelope edit screen control this; "Vel Gain" determines how responsive MIDI Synth is to the intensity of the "attack" reported by the keyboard, and "Decay Gain" determines how responsive MIDI Synth is to the speed of release ("aftertouch").

Some keyboards include a device called a "pitch wheel" that changes the characteristic of a sound either by changing the pitch of a note or the overall amplification ("gain") of the envelope as it shapes the sound. Again, MIDI Synth supports this feature for the envelope by a "Pitch Pend" (we're not sure where "Pend" came from) control; the higher the setting, the more pronounced a "pitch wheel" variation will be expressed by the envelope.

There is a lot here to digest here, but we hope this gives you a little more insight into synthLAB's control of the IIgs as a synthesizer. Now what we need is a way to let the IIgs communicate with other musical instruments to control them or to play notes in response to their control. For that, we have to get into the musical communication standard of MIDI, and we're saving that for next month.

If you're interested in some hard theory on the physics of sound John R. Pierce's recently revised *The Science of Musical Sound* (W.H. Freeman and Company, 41 Madison Avenue, New York, N.Y. 10010, ISBN 0-7167-6005-3, $19.95) is an in-depth resource. For digital music synthesis, I recommend Hal Chamberlin's *Musical Applications of Microprocessors, Second Edition* (Hayden Books, 4300 West 62nd Street, Indianapolis, Ind. 46268, ISBN 0-672-45768-7, $39.95).

The good apple

by Jack Powell

A brain teaser (remember them?):

It is probably the largest Apple II user group in the world. The users of this group are not generally enthusiastic although their needs are real. There's hardly an expert among them. They use their computers every day although many do not own one. Meetings are over two hundred times a year in all fifty states. If a member doesn't show up or leaves early, a note is required from the parents.

We do not usually think of schoolchildren as members of an Apple II user group. But if we do, it's easy to see that the opportunity to excite a whole generation is knocking very loudly. In fact, in some places, opportunity is practically breaking down the door. How many times has Apple, Inc. been told how much it could gain from supporting Apple II users? Let's step down a level or two and see how we ourselves, forget Apple, can support this very large Apple user group—our schools.

Getting started is easy. Pick one school to keep yourself from getting spread too thinly. A nearby school is easier to keep in touch with. All the better if the school is afflicted with your own children, for they can make great messengers.

The only formality comes in introducing yourself. Since teachers perpetually need volunteers, there is always a signup list at the class orientation meeting at the start the school year. Leave your name and phone number and note that you would like to help with the computers. Shake the principal's hand and find out which faculty members are interested in computers. Notice what kind of Apple IIs are being used in the school and scan the software library to see which programs are available.

The fun begins when you reintroduce yourself by practicing the fine art of donating software. Anything you send in is a demonstration of what an Apple II can do. It also shows that you can do things for the teacher with the Apple that were previously unimagined, and an unexpected enthusiasm for the machine can become volcanic.

Donating software (legally) has lots of purposes and doesn't involve more than the price of a blank disk. I usually send a disk in at least every two weeks, with a note offering to help in any way. Freeware and public domain offerings are abundant. Tried-and-true favorites include *MatheMusic, Modulae* (from the FTA), and *HyperStudio* stacks. *Print Shop* graphics (with a printout) can be used for artwork, announcements, and in some word processing programs for the classroom. Games can be used as rewards or motivators. There are some awesome graphics out there. The shareware program *PixMix* makes jigsaw puzzles out of SHR graphics. It is available from the major online services, your local Apple BBS's or Big Red Computer Club, 423 Norfolk Avenue, Norfolk, NE 68701, for $3.50. Even utilities are useful to those who have no idea what they do, as they are invaluable in customizing disks. One teacher I know has students clamoring for him to fire up an educational program on his Apple IIgs, because his disks are customized with programs such as *StartPic*. The first disk he got opened with an SHR graphic of a stunning model; a balloon caption had her saying that this particular teacher was **her** kind of guy! The howls could be heard all the way to Cupertino.

The demo disks ignite in the classroom and it won't be long before you learn what the school **needs**. Just ask and listen, especially to gripes or frustrations. Maybe some students need drill-and-practice on some part of the grammar curriculum and you know of a program that fills the bill. The parents who volunteer to work in the computer lab or in the classroom often do so because the school needs warm bodies in those slots. They want to help, but their knowledge of the Apple II frequently ranges from nonexistent to abysmal. Give some pointers to one of them, or to a student; that person can teach others. One teacher sent software programs home with one of my daughters so she could learn to use them, then teach a few others, and so on.

More sophisticated hands-on training demands even less time and effort on your part. What is required is software that sparkles. Suppose some students need to learn how to use a mouse or a menubar. You can give a half-hour talk and a demonstration on your own time. Or you can be much more time-efficient and send in a copy of *MatheMusic*, in which nearly every selection within the menubar has an instant and dramatic effect. The same ideas hold true for such features as New Desk Accessories. A truly crafty person would send in one system disk with *Quadomino* on it, and a short note on how to access this *Tetris* clone. After about a week, every member of the student body will be familiar with NDA operations; a substantial number of bleary-eyed faculty will be proficient as well.

Not feeling very proficient yourself? Remember three things. First, those who help others solve a problem often gain some knowledge themselves. Second, if you can turn on an Apple II, you are way ahead of a large percentage of these Apple II users. Third, a school user group will learn its most important lessons when their questions have finally stumped you. Consider a question or problem for which you have no answer. Let's say the answer is known only by some higher being in the Apple world. You have a number of good options.

To begin, if there's someone else supporting the school, check that person out. Maybe it will turn out to be Roger Wagner or Steve Wozniak. Then again, maybe not. Telephone the dealer or another local user group. In my town we have neither, so our next step is to write ***Resource Central*** (answers are surprisingly fast even for snailmail), or post the question to one of the online services via modem.

No modem, you say? Pishposh. A mere and inconsequential detail. Someone in another user group may have one. A parent in the school with a modem and a subscription to an online service can reach an Apple II forum with any kind of computer, Apple II or not. Many schools have modems that they use to communicate with one another or the district offices. You can reach the local bulletin boards in this manner and some benevolent soul can relay the message to a major online service if need be. Whatever route is taken, your school's query can eventually get to the guru and then the process reverses itself till you strut in with the answers to all the woes.

There is recognition that you are doing something important. There is a realization that the students and faculty members have status in the Apple II world because they are Apple II users. It is a wonderful morale builder. There is no question too trivial and no problem too small. People are glad to help. They want to hear from you.

Forums are already in place and waiting for school user groups to step in and benefit from their area of expertise. These include ***A2-Central***, Quality Computer's *Enhance* and sections within the online services. These are all places in which there can be sharing of questions, answers, successes, and ideas. Ideas of all sorts, from how to get a community to donate educational software and old hardware, to methods of encouraging technophobic teachers to use it.

Companies need to know what hardware and software works, and why. Or what problems are popping up. After all, a software product may be immensely more popular if just one feature is changed. Maybe your software donations can include a demo of a company product, which are often available through online services or by contacting the company itself.

Programmers might enjoy hearing about novel uses for what they have written, as well as ideas for new software and ways of recycling old products. A person whose laurels came from one popular game may be very interested in a group of sixth graders who need some drill and practice on multiplying decimals. The programmer may conceive of a series of simple modular programs in which problems are presented on a graphics screen showing an arcade game and a coin slot. Each correct answer drops a nickel into the slot. When the total is fifty cents, one round of gameplay is allowed. A hundred correct answers in a row allows unlimited play. A little work, and new shareware programs are hatching out all over. (Teachers are very conscientious about paying shareware fees, in my experience. Parent-teacher organizations can help too.)

That creaking sound in the distance may be a school market opening up. It will get louder if there is publicity of any type. Parents may hear about what great things are happening in the school being supported by another Apple user. If their children don't go to that school, they will know that their children are missing out on something good. And you know, parents hate **that** with a purple passion. They will demand something similar in their own school. That will not be your problem, but it means that there may be more benefits spread around.

As Apple II users support the user groups in the schools, the results are something like a party. Folks have fun. One way or another you will meet all sorts of people. And as all the commotion gives an ulcer to those PC owners down the street, you can tell them this:

The party is just getting started.

Miscellanea

Another one bytes the dust. Although they tried every trick in the book to stay above ground, the shrinking Apple II market proved to be too much for the programming-oriented magazine, *Nibble*. After a dozen years of meeting the needs of Apple II users and innumerable *false* accounts of their demise, reality hit. *Nibble* subscribers learned a few weeks ago that the July issue would be their last and that we here at ***A2-Central*** would fulfill their remaining issues. While we're more than sorry to see *Nibble* go; we're glad that we are able to continue to provide support to Apple II users.

We want to hear from you. We realize that we're at the midway point of a narrowing funnel. At one time, Apple II users had many choices when it came to sources of information. Now we have a few. Our subscribers are a diverse bunch with diverse needs. While it would probably be impossible to fill **all** the needs of **all** of you, we do want to know what you want. And the only way we can know is if you tell us. Tell us.

Those of you planning to attend the Apple Central Expo on July 25-26 but who didn't get the official brochure from Event Specialists, might be interested in knowing that the official Expo "Show Hotel" is the Overland Park Marriott. The hotel is offering a rate of $69 for a single/double room which includes break fast and free bus service to and from the show. Not a bad deal for a place with both a McDonalds *and* a KC Masterpiece barbecue restaurant across the street. Call them at 913-451-8000. Make sure you mention that you want the special rate in conjunction with the Apple Central Expo.

In keeping with the summer events theme, our conference this month looks like it will be bigger and better than ever. The dorm rooms at Avila college are filling up sooner than they did in the past, which just goes to show you something that we already knew. The Apple II is far from dead. No corpses at this conference. Aside from all the dynamite sessions planned, there are a lot of fun activities and surprises in store for all of you who are attending.

The Byte Works, Inc. has just released a new self-paced course for toolbox programming on the Apple IIgs. Geared toward the intermediate — advanced programmer, *Toolbox Programming in Pascal* is designed to guide you through the Apple IIgs Toolbox using the ORCA/Pascal development environment. It uses a hands-on approach to learning (as an ex-teacher I can attest to the fact that this is definitely the best way to learn) and by the time you're through with the course, you'll have written a dozen desktop programs.

Toolbox Programming in Pascal comes with the largest library of Pascal toolbox source code ever assembled, (four disks-full) plus an abridged toolbox reference manual. The materials in this package are so comprehensive, you won't need any other reference materials. For more information contact The Byte Works, Inc., 4700 Irving Blvd. NW, Suite 207, Albuquerque, N.M. 87114, 505-898-8183.

***Express* LaserWriter (but no delivery).** Since our *Express* review we've found a glitch with the use of AppleTalk printers. *Express* disables printing to the LaserWriter when loaded into memory even if *Express* is turned "off." Printing to direct-connect printers works fine. What's confusing about the failure is that you don't get a warning that something's wrong until you actually get to the stage of sending data to the printer. At that point, the normally lethargic LaserWriter progress dialogs (the ones you see after selecting "Print..." from the "File" menu and clicking the "Print" button in the resulting dialog) flash past and nothing is actually printed.

If you have a printer that operates with a port driver and a network printer, you'll need to reboot with the *Express* CDev inactive to use the network printer during that session.

Questioned SANE-ity. One reader recently questioned the utility of the Innovative System's *Floating Point Engine* when their AppleWorks spreadsheet was not recalculating noticably faster with the math coprocessor and software installed. We checked and found the correlation that we expected. However, others may not anticipate this: a math coprocessor shines on complex calculations.

The spreadsheet in question used mostly simple four-function math (addition, subtraction, multiplication, division) with most of the calculations being addition. Most microprocessors can add (or subtract) very fast. Multiplication is a bit harder, division harder yet, but still not major drains. If you really want to test the SANE package used by AppleWorks or the IIgs ToolBox you need to work with mathematical functions that require more complicated calculations, such as trigonometric or logarithmic functions, which are usually evalated by a series of calculations until the answer seems to "stabilize".

We replicated addition, division, and sin() functions over a few hundred cells (A block covering "A1" through "R100") and timed recalculation for comparison. The speed-up for the first two were not earth-shattering (AppleWorks was pretty quick by itself) but the difference with the sin() calculations was striking (all times in seconds):

	Addition	Division	Sin()
AppleWorks 3.0	9.75	16.24	117.99
AW 3.0 with FPE	8.71	12.16	12.18

The moral here is that if you're doing basic math the floating point processor may not be what you need to speed things up; an accelerator (*TransWarp* or *Zip*) may be more advantageous. If you're doing more complex math, though, the *FPE's* strength kicks in.

On the other hand, users have reported problems in using the *TransWarpGS* (even the newer model) in conjunction with the *FPE*. We have yet to hear of a resolution for this. We have calls in to Innovative Systems, but response is sometimes erractic.

The recommended configuration for System 6.0 is two disk drives. This requirement is contrary to our original information but we have recently been told that a single-disk configuration has not been fully tested. If you're having trouble with a single-drive configuration, we'd like to hear the details.

Pain in the FST Department. A couple of months ago we published a dogmatic statement about the "transparency" of GS/OS FST's as used to read data from foreign disks. Again, karma seems to have run over the dogma. One of the applications mentioned was reading High-Sierra disks from HyperCard IIgs. It turns out this doesn't work because when you attempt to open the file the combination kicks back a spurious error indicating that HyperCard can't create the new file of the given name (wake up FST; we're trying to open an existing file). For some reason, this only appears to happen with the HS.FST; we've passed the glitch back to Apple (if we'd been awake, we'd have found this earlier when testing System 6.0 and HyperCard IIgs 1.1).

The terminator revisited. The new SCSI-2 standard seems to be causing some consternation among users of the Apple II High-Speed SCSI Card. It's not time to panic yet; here's why.

The SCSI-2 standard defines that a host (e.g., the SCSI interface) *must* supply terminator power and that the devices on the bus *may* supply terminator power. This has caused some owners of the Apple Card, which doesn't supply terminator power, to panic about the use of drives that use SCSI-2 mechanisms. So far we haven't seen a problem as long as the drives supply power to their own terminators; apparently the power that "seeps onto" the bus is sufficient to appease the terminators on the Apple II High-Speed Card. At least, it's worked for us.

Although not strictly adhering to the new specification, we suspect this will continue to work into the forseeable future if you select a drive that works on the Mac Plus (which also does not supply terminator power to the bus) or the new Mac PowerBooks (which use minimal internal termination). Such drives must supply their own terminator power since the host computer doesn't and the High Speed SCSI card seems to be able to live on the amount that remains on the bus. However, there is one trick; if you have more than one device attached to the SCSI chain besides your computer, you need to terminate the first and last device on the chain. We picked up this trick from the PowerBook manual and it seems to work with the Apple SCSI interface as well. (The easiest way to terminate the first device is with a pass-through terminator that attaches to the SCSI port on the first drive and accepts the SCSI cable from the computer. These can usually be located in the 50-pin SCSI form for about $20.)

Matt Gulick, designer of the High-Speed SCSI interface, dropped by on GEnie recently to relate that newer builds of the card do supply power to their own termination resistors (but they still expect other devices on the bus to supply their own terminator power). But we've found the above techniques to work even with the older cards.

We repeat an old caution: don't plug or unplug SCSI cables with the system or the drive powered up. On some drives this will cause a microfuse (not a user-replaceable item; it may be integrated into the drive circuitry) to fail and you'll lose terminator power from the device. **This may cause it to stop working.** If you blow the fuse, you may find you've blown your warranty on the drive (don't ask us how to *fix* it; just don't *do* it).

On the other side, we've had a few reports of older Seagate mech anisms suddenly losing terminator power. We haven't had luck in getting these repaired (the last replacement mechanism we received also didn't supply term power). To get the drive to work we had to use a CV Tech RAMfast interface, configured to supply terminator power.

Foreign voltage on overdrive. One other drive problem we've heard about is power supply failures on units used overseas. Usually these are "multivoltage" units that can tolerate wide variations in input voltage (maybe 90-270 volts at 50 or 60 cycles per second).

The problem comes in when you're in an area where the voltage is very close to the top end of that "permissible" range. We've had higher rates of power supply failure in areas where the voltage is high (up over 240V), apparently taxing the supply. If you're in one of these areas, you may want to take two courses of action. First, plug the drive into a surge protector; this may help filter out dangerous spikes. Second, if the drive includes power outlets on the back don't use them. They may cause "spikes" as the attached devices are powered on or off.

In most cases the replacement of a power supply is not extremely expensive. But if the supply doesn't fail in time to protect the hard drive circuitry then that mechanism may have to be repaired or replaced, and that can be painful.

Ask (or tell) Uncle DOS

Corrections

Okay, I did not do it this time. In fact, I've not been doing it for two months but nobody's paying attention. Attention. Yeah, that's why I did it—or didn't do it. Okay, people, take out your pencils and sharpen them. Open up the three-ringed binder that you keep your back issues in and turn to June 1992. Renumber the pages so that 8.26 becomes 8.34 and so on. The last page of the June issue should be 8.40. Wait. We're not done. Renumber the July issue so that its' page 8.26 becomes 8.42 and so on.

Where in the world

Where did Mr. Coughlin (*Science Fiction Theater*, **A2-Central** pp 8.20-8.22) find a 10 mhz TranswarpGS? His slot 1 "contains an Input/Output board with eight input and eight output switches," is this a Slotbuster II? It's the only card I know of that fits that description. Where did he find the 4-way switch connected to serial port 1 and the X-10 Powerhouse Computer Interface?

Frank Gizinski
Racine, Wisc.

Here's the answer, straight from the horse's mouth:

I didn't actually find a 10 Mhz TransWarpGS. I upgraded a stock 7 Mhz TWGS (v 1.5) to 10 Mhz using a faster cpu, a 40 Mhz oscillator and Applied Engineering's 32k cache upgrade. There is a file online in the A2 library on GEnie (#11958) that explains various speed modifications for the TWGS (some of the information regarding replacing the cache has been superceded by the 32k upgrade).

The board that I have in slot 1 is the I/O 8 made by Applied Engineering. The current version (as far as I know) is the I/O 32 (32 on/off "switches" instead of 8).

Switch boxes for the Apple IIgs serial ports can be obtained from most computer supply stores. You will probably want one with the same "mini-8"connectors that are used for the Apple IIgs ports. The least expensive ones I've bought (2-way {A/B})mini-8 connectors part #2278—$14.95, 4-way {A/B/C/D} mini-8 connectors, part #2279—$18.95 came from Lyben Computer Systems, 1150 Maplelawn, Troy, MI 58084, 313-649-4500. They've been working fine for 6 months now.

These are the standard rotary knob Data Switch switches; lifetime warranty, gold-plated contacts and shielded metal case.

The X-10 Computer Interface is also available from several sources. There are three models: Apple II, IBM and Mac. Each model consists of the same basic X-10 unit, a compatible cable and software specific to the machine. I bought my unit from Heath, 800-253-0570, part #BC-290-A (Apple II).

There are a few "gotchas" with the Apple II model. It was designed for the Apple IIe and IIc, which means there is a cable to connect to the IIc to the X10 unit and a cable to connect a serial board in a Iie to the unit, but no cable to connect to a IIgs serial port. Likewise, the supplied software will run on a IIe or IIc but will choke on the IIgs serial port. As far as I know, the only way to run the software on the Apple IIgs is to plug a Super Serial Card (or equivalent) into a slot and connect to the unit with a IIe cable. Since I built my own cable and wrote my own software, this was of little concern to me.

If you intend to use the Apple IIgs serial port and therefore don't care about the supplied software, I believe the Mac model has the correct cable for the IIgs.

Speaking of X-10 software, I was recently "encouraged" by two GEnie members to upload an NDA I wrote years ago to control the X-10 through either IIgs serial port (file #18473). The NDA will provide access to all 256 devices, however there are some restrictions on the use of timer events, since the NDA was written specifically for my needs. By the way, the NDA documentation contains enough information on the IIgs /X-10 pin outs to construct your own cable (or have Redmond Cable, 17371 A1 NE 67th Court, Redmond, WA 98052, 206-882-2009, build one for you).

I am currently re-writing the NDA for more general use and to take full advantage of the X-10 features. —Art Coughlin

Barrell of mail

Ross Barrell's ("Just a (bad) memory," July 1992, page 8.44-45) problem isn't memory, or any of the solutions you listed, but the AE Vulcan 20 MB Controller Card!

In the fall of 1990, I purchased an Apple IIgs, ROM 03, with various add-ons. After the first week of use, the file names began to change, along with the file extension. I tried renaming, reformatting, and again renaming files, all to no avail.

I therefore proceeded with the infamous SPOE/R&R (Standard Process of Elimination/ Removing & Replacing)-including the power supply. I found the culprit was the AE Vulcan 20 MB hard drive! I called Applied Engineering's technical support and informed them of my findings. The technician told me that they had a problem with a batch of 20 MB controller cards. (the noise generated by the IIgs motherboard was interfering with the WRITE ROM on the Vulcan Controller Card and altering characters in filenames during the disk write.)

I returned the card for a free replacement. I received my new card, reformatted the hard drive, and haven't had a problem since!

Michael Demyan
Bethlehem, Pa.

The letter from Ross Barrell made my day. He is the first to relate the problems I have been having for two years. I think he will find his 20 MB Vulcan is the problem. I have a 40 meg Vulcan but did not use AppleWorksGS much because it crashed so often. The first crash happened as he described, with the nonsensical volume/file names and illegal ProDOS symbols; mine did not corrupt the name of my partition. The Finder screen was even worse with most of the folders not even recognized. To keep it short, I will just say that I could only recover a few files and they appeared damaged.

I began to get "bad block on disk" or "AE1 may be damaged" errors when copying or moving files. The verify and validate tools on Deliverance always reported that everything was fine. I struggled for almost a year before calling Applied Engineering. After suggesting a low level format (after a backup), they finally suggested that I check the version of my hard drive controller card. (Hold it to the light, look on the left end of the card.) Versions A and B allow RF interference to corrupt information but if there is a resistor soldered on the back of the board it has already been fixed. I replaced the card for about $50 and the problems are less severe but still persist. The most irritating problem is the apparent internal corruption of system files—some of the GS/OS programs crash and have to be recopied to the drive.

In the interim, I have been using Prosel-16 and it initially showed all types of minor file corruption. There were many bad block calls that could only be removed by going to the "sort" screen, sorting the file to the end of the list, then deleting it. There was no other way to get rid of it. Voila! Although I still have occasional problems, this seems to be the final solution.

Jim Sharp
Waco, Texas

With reference to Ross Barrell's letter in the July 1992 issue of ***A2-Central***, describing problems he's had with AppleWorksGS corrupting his disks, I've had somewhat similar problems myself. Assuming that he's using version 1.1, my guess is that he's running into memory utilization problems.

I have a ROM 03 machine, currently with 2.25 megabytes of memory. I had all my problems with the package when I had only 1.25 megabytes of memory. Even a moderately complicated page layout would lock the system up completely. All problems were with the spreadsheet module; saved spreadsheets would frequently be unreadable. Even with the memory I have now, I still frequently run low. But I haven't had a corrupted file since upgrading the machine.

Here are several suggestions that have helped me when I was having problems:

1. Save often and always under a different name. You should be able to back up to a previous file that hasn't been corrupted.
2. Monitor memory usage closely. If you hold down the Option key when you select "About AppleWorksGS.." from the Apple menu, the package will display some very useful memory statistics. If the "largest contiguous free block" drops under 100K, you're probably about to get into trouble.
3. Never edit and print a complicated document in the same session. Always save, exit AppleWorks, re-launch AppleWorks, then print the document. This will make the maximum memory available to the print tool set. (I think that this was the source of my problems.)
4. If you are running really low on memory, you can preconfigure AppleWorksGS to preload the page layout module; this is usually more memory efficient than demand loading when you open the file. I would only do this when you are having problems with a particular file, however; you should generally not have anything preconfigured.

Face it—AppleWorksGS is a pig! And it doesn't get any better under System 6.0. I think there's another megabyte in my future.

David A. Shaw
Manassas, Va.

Celebrating

There has been a little bit of game coverage in ***A2-Central*** lately ("Imagination Rules the World," December 1991, page 7.81), so I thought I'd test the water and ask why Doug Smith's *Lode Runner* did not rate a mention.

My kids have almost every game in the Universe but they always get back to *Lode Runner.* Since I bought my first Apple II, they have constructed some 200 levels of varying difficulty. With all the power and gloss of the Apple IIgs, it's a game that is in fact better played on the early II's. A copyable ProDOS version for the IIgs would be nice, but of course we'll never see it. I started to write a ProDOS 8 version some time ago but ran out of time and knowledge.

I would also like to give Glen Bredon a plug. *Prosel 16* is the most remarkable piece of software I have used on any computer. I've had my share of crashes and problems, but Prosel has worked without the slightest glitch. Almost all software lacks some aspect or feature if you look hard enough, but I can't think of a single thing for a *Prosel* wish list. After backing up and optimizing my 100 meg hard drive, I almost expect it to hand me a cup of coffee.

I must look at the System 6.0 Finder one day but after *Prosel,* it would need to be something special indeed to get my attention. I work on MS-DOS and have an Apple IIc, IIgs and a Mac at home. The IIc gets flogged by the kids and in five years has not required any service of any sort. The IBM is about as interesting as an adding machine and the Mac has a screen the size of a postage stamp. That leaves me the Apple IIgs and what a machine it is! I don't care if Apple pulls it off the market tomorrow. It will suit my needs for years yet.

Richard Stephens
Toowoomba, Queensland

Fiddlin' with FST's 'n flatbeds

Something of interest for GS/OS users. The HFS.FST which comes as part of System 6.0, works very well with GS/OS 5.04. You install it just as you would with System 6, i.e. copy the FST into the FST folder and restart the computer. You have complete access to Macintosh volumes even though you aren't running System 6.0.

I have tested the combination extensively and found no surprises. *Platinum Paint* with its' *MacPaint* import capability is very fast, *SuperConvert* and *Graphic Exchange* also work smoothly. Programs such as *MY.WORD* access Macintosh Read.Me files and text files without a hitch. I even went so far as to attach a Macintosh hard drive to my Apple IIgs for testing. The System 5.04 with the HFS.FST worked flawlessly, allowing me to pull graphics, data and text files directly. Other file activities such as rename, delete, copy, etc. all worked fine. Just thought you would want to know.

Another subject. The Apple Programmer and Developer Association (APDA) has a CD-ROM called *Twenty Thousand Leagues Under the CD* which contains a partially completed Apple IIgs application called *AppleScan GS.* This actually works with the Apple Flatbed Scanner hooked into the SCSI chain of the IIgs. I say "partially completed" because the application does not have a functional Save command. Aside from that, the application works with the scanner to produce absolutely superb gray scale 320 mode pictures. The speed easily compares with the bigger Macs. Since the Save function doesn't work, I have found several ways of working around the limitation. One is to use an NDA called *Clip-it* to transfer the scanned image to the clipboard and then into *Platinum Paint* or *816 Paint.* The other approach is to use the *FingerPrint GSi* screen capture board to transfer the screen to a disk file and subsequently into *Platinum Paint* or *816 Paint.*

The CD-ROM mentioned is available for a very reasonable fee, but is in Macintosh format only. This requires that you convert all of the Apple II material to Apple II format before you can use it but the effort is worth it. Credit for the research and conversion of the Applescan software I have been testing goes to a young man by the name of Scott Jennings. It was also his suggestion that I try the HFS.FST with 5.04 software.

Vern Mastel
Mandan, ND

A2-Central™

© Copyright 1992 by
Resource Central, Inc.

Most rights reserved. All *programs* published in ***A2-Central*** are public domain and may be copied and distributed without charge. Apple user groups and significant others may obtain permission to reprint *articles* from time to time by specific written request.

Publisher: **Tom Weishaar**
Editor: **Ellen Rosenberg**

with help from:
Denise Cameron, **Dennis Doms**, **Sally Dwyer**, **Dean Esmay**, **Richard Ginter**, **Jeff Neuer**, **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): $34 for 1 year; $60 for 2 years; $84 for 3 years. All back issues are currently available for $2 each; bound, indexed editions of our first six 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 $90 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.

A2-Central is sold in an unprotected format for your convenience. You are encouraged to make back-up archival copies or easy-to-read enlarged copies for your own use without charge. You may also copy ***A2-Central*** for distribution to others. The distribution fee is 20 cents per page per copy distributed.

WARRANTY AND LIMITATION OF LIABILITY. We warrant that most of the information in ***A2-Central*** is useful and correct, although drivel and mistakes are included from time to time, usually unintentionally. Unsatisfied subscribers may cancel their subscription at any time and receive a full refund of their last subscription payment. The unfilled portion of any paid subscription will be refunded even to satisfied subscribers upon request. OUR LIABILITY FOR ERRORS AND OMISSIONS IS LIMITED TO THIS PUBLICATION'S PURCHASE PRICE. In no case shall our company or our contributors be liable for any incidental or consequential damages, nor for ANY damages in excess of the fees paid by a subscriber.

ISSN 0885-4017 **GEnie mail: A2-CENTRAL**