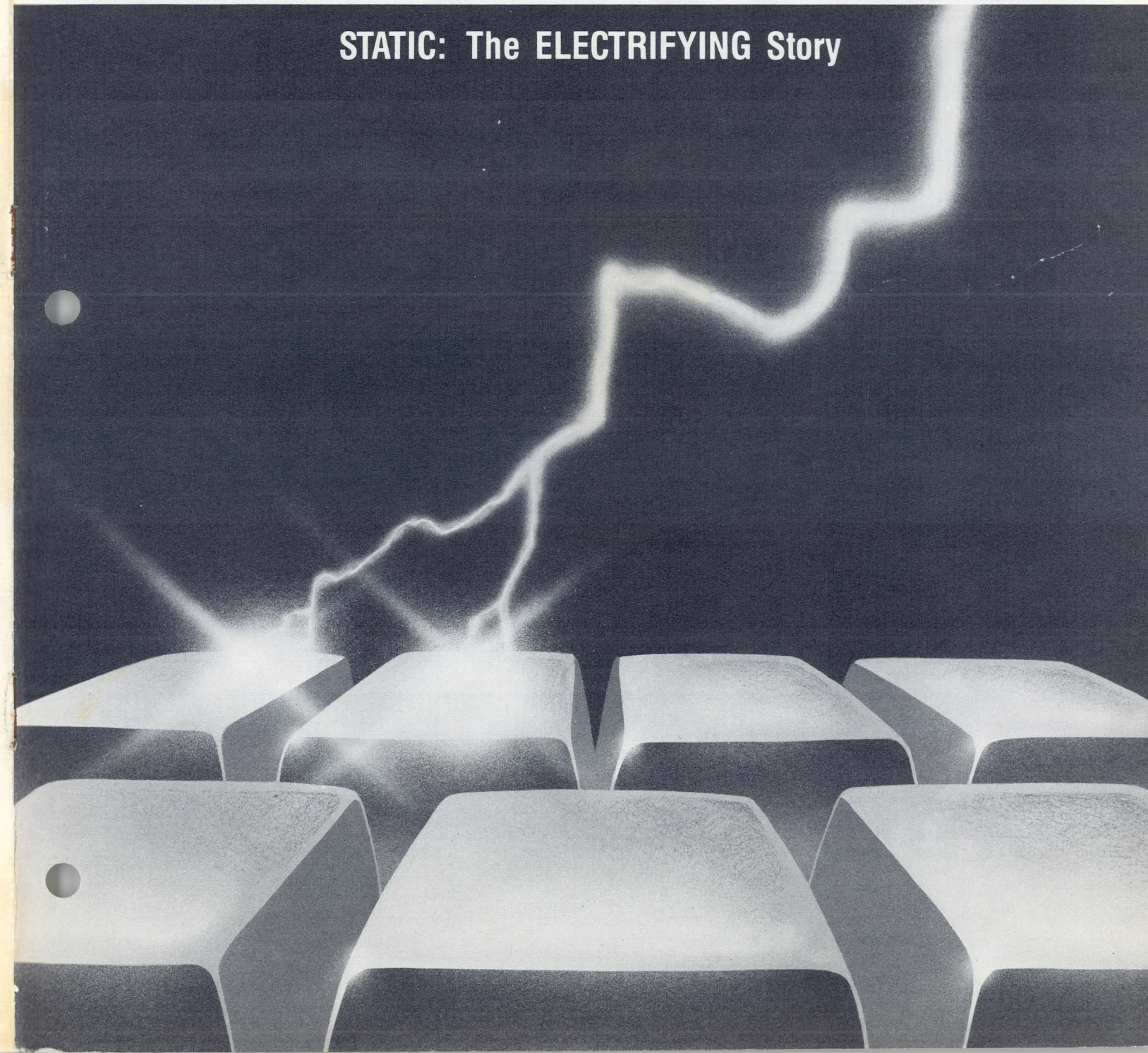
SOURCE DATA

THE DATAPOINT CUSTOMER MAGAZINE

Volume 5, Issue 4

July/August 1983

STATIC: The ELECTRIFYING Story



IN THIS ISSUE

Volume 5, Issue 4 July/August 1983

Editor: Karrie Oertli

SOURCE DATA is published bimonthly as the DATAPOINT Customer Magazine. There is no subscription charge for the magazine. Eight weeks are required for change of address. Please give both new and old address as printed on the last label. Address all correspondence to: Karrie Oertli, Editor, SOURCE DATA, 9725 DATAPOINT Drive #T41, San Antonio, TX 78284. Telephone: (512) 699-5244. Manuscripts, drawings, and other material may be sent in care of the editor; these items must be accompanied by a stamped, self-addressed envelope if return is required. All material in each issue is accurate as of the publication date. Rely on up-to-date permanent documentation, such as user's guides, for details of software or equipment features. Software release and hardware availability dates are estimates and are subject to change without notice. The "D" logo, DATAPOINT, Attached Resource Computer, ARC, LightLink, DATABUS, DATASHARE, and DATAPOLL are trademarks of DATAPOINT Corporation registered in the U.S. Patent and Trademark Office. AIM, Associative Index Method, ARCNET, DASP, Integrated Electronic Office, Resource Management System, RMS, EMS, MULTILINK, and SHARE are trademarks of DATAPOINT Corporation.

SOURCE DATA

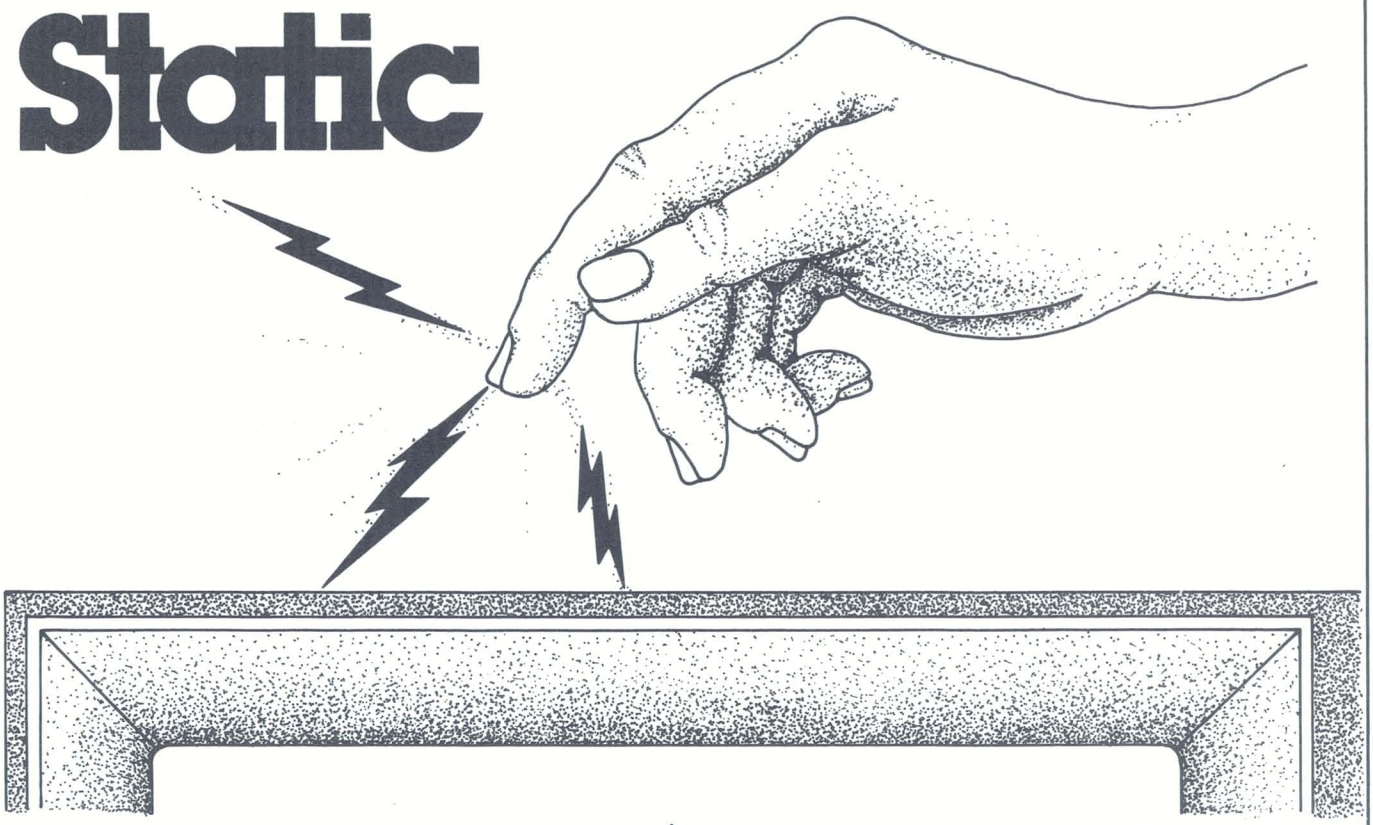
DATA PROCESSING

OFFICE APPLICATIONS

PERIPHERALS

ADVERTISING

Static



Have you ever touched your processor or bumped the desk it sits on, and seen an "F8 INSTRUCTION ERROR" appear on the screen? It's a strange error, and often, it's unexplained. Well, most of the time, that error is caused by static. That's right, it's caused by the static we store in our bodies. This article explains how static is generated, how it's stored, and how it is released.

Most of us have been "shocked" by the carpet. At least, that's what we think it is. In all reality, it is due to "a discharge of electrostatic potential," or static. We can all remember walking on new carpet, dragging our feet, just so we could surprise someone. When we got close to a conducting surface, like our younger brother or sister, we got a shock when the electricity jumped between our bodies and the "conducting surface" (or body, as the case may be).

This experience we've all had illustrates the basic concepts of static: the buildup of electric potential on one surface (our bodies) and the discharge, or redistribution of electric charge, to another surface (our little brother). Static basically needs three factors to be produced: two dissimilar materials (like wool carpet and rubber soled shoes), friction (like dragging our feet), and a lack of humidity (usually caused when air conditioning or heating dries the air). A static charge is accumulated on the better conductor when two different materials are rubbed together. So, when we drag our feet on the carpet, the static charge is stored in our bodies, since bodies are a better conductor than wool carpet. Static is stored on the better conductor until its charge is transferred to a lower potential conducting surface (like our little brother or a metal table).

If you can recall your high school or college physics, you'll remember that atoms make up all matter, and that they consist of four items: protons, electrons, neutrons, and a nucleus. Protons are positively charged, electrons are negatively charged, and neutrons are neutral, or have no charge. Electrons and protons attract each other (unlike charges attract; like charges repel), and this holds the atom together. Some atoms have an extra electron that can be lost without changing the basic structure of the atom itself. These extra electrons form the basis for static.

Static is accumulated when extra electrons are taken from one surface and given to another. For example, when a glass rod is rubbed with a piece of wool carpet, static charge is built up on the glass rod. What actually happens is that extra electrons from the atoms in the wool (an insulator) jump to atoms in the glass (a conductor). And since electrons are negative forces, the glass becomes negatively charged, since it now has more electrons than protons. When static occurs, the extra electrons on the now negative conductor jump to a more positive conductor. Thus, one conductor becomes positive (from losing electrons) while the other becomes negative (from gaining electrons). Releasing static means that electrons move freely to the conducting material and leave a net positive charge behind.

If both materials that come into contact are good conductors, the charges do not build up. Instead,

continued on page 4.

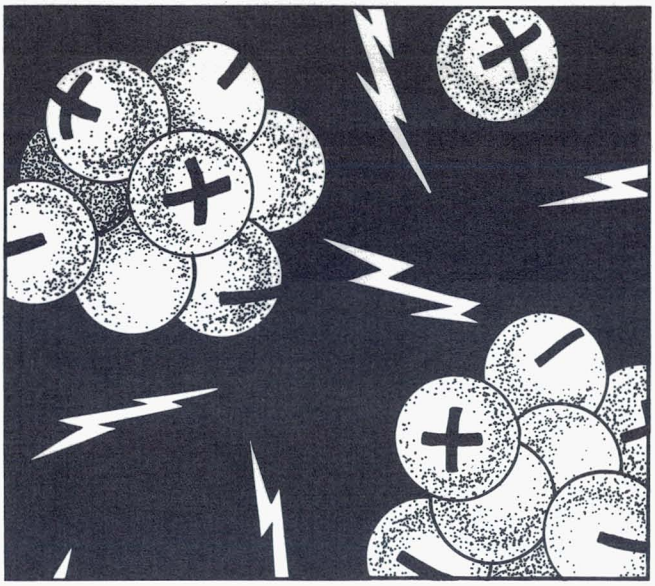
they are conducted away and redistributed, since like forces do not attract. When either or both materials are insulators, the electric charge is stored, not redistributed. Let's go back to the example of wool and glass to have a look at conductors and insulators.

If we rub a piece of wool on a glass rod, we will store up a static charge on the glass; the amount of static will depend on the amount of friction we produce (taking into account the amount of time and force we put into our rubbing). When we take that glass rod and get close enough to a metal surface, for example, static is released when the static voltage becomes sufficient to ionize the air in the discharge gap. This gap can appear between the charged body and any other body. Usually, the discharge occurs between the charged body (either negatively or positively charged) and a good conductor (charged in reverse of the other material). We'll sometimes actually see an arc, a visual display of electrons that looks something like a very small lightning bolt. (Incidentally, that's exactly what lightning is -- a redistribution of static in the atmosphere.) What is actually happening is that electrons are moving freely from one surface to another.

As the number of stored electrons increases, the electric potential difference, which is called voltage, also increases. Sometimes this static voltage can become as high as 50,000 volts, though this isn't the norm. Fortunately, the total number of electrons flowing by static is very small, so damage is usually microscopic; so microscopic, in fact, that we don't usually even feel it. When we do feel static, the voltage is at least 3,000 volts. To put it in other words, we probably won't be hurt by static because the amperage, or the measure of the number of electrons flowing through any material, is low, although the actual static shock can be painful!

As the local relative humidity increases, the tendency for static charges to build up decreases. As humidity increases, some insulators become better conductors, because they become more saturated with water. Such is the case with wood. We all know how the electric companies use wood poles to connect power source to power user. Have you ever noticed the tops of those poles? In some rural areas, you'll see that these companies use glass insulators on their wooden poles. They use an insulator because those wooden poles can become good conductors when they are saturated with water. Glass insulators don't become saturated, so they are used between the "power" and the wood.

You might wonder what this has to do with computers. Well, since computers are based entirely on the transfer of electric charges, any fluctuation in those charges can cause immense problems. Such is the case with static. Basically, static causes a fluctuation in electric charges. Computer circuits run at relatively low voltage levels (5 to 12 volts) and at low current levels. These circuits are very susceptible to static disturbances. Static charges can reach



as high as 3,000 volts without our perception; consequently, without our perception, they can cause a surge of power to parts of the computer and can actually destroy semiconductors, those small pieces of silicon, which are the basis for the integrated chips used in the computing industry today. So, you can see just how destructive static can be. When a static charge occurs within a computer system, the chance of a semiconductor being blown is high. As many as ten thousand silicon circuits can reside within one integrated chip. Whenever a surge of energy is received in this circuit, such as a surge it would receive from a static discharge, the silicon chip, which is as small as one of the characters in this article, can actually be blown up. This "explosion" can cause a physical peak to form on the metal circuitry. This peak will cause incorrect electrical current flows, which make the circuit unusable. A single character that an operating system will recognize requires many chips to control its makeup; blowing one chip in a circuit can render a character unrecognizable. And one lost character can make a processor unusable, since none of the software that includes that character as part of its makeup can be executed.

Many of the problems you'll see from day to day are probably caused by static. Even though DATAPOINT spends quite a lot of money to make its equipment immune to static, there are still times when static voltage can reach levels that will harm the equipment. If you can take the time to see how you can prevent static in your office environment, you can probably eliminate most of the problems that are static related. And the bottom line is less down-time for you and a more error-free environment for your computer system.

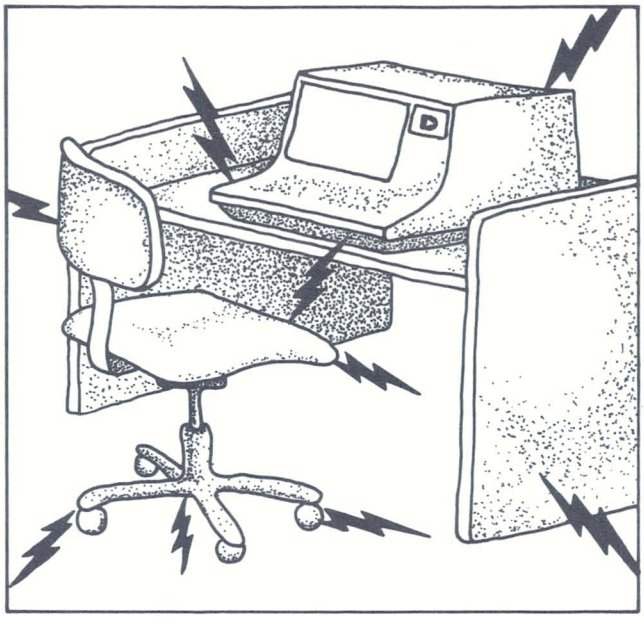
Now that you have a hold on static and can understand just how it occurs, please read the next article, "Take the Charge Out of Your Computer." It can help alleviate the problem of static with your equipment. ■

TAKE THE CHARGE OUT OF YOUR COMPUTER

If you've read the cover story on static charges, you know just how detrimental static can be to your computer. In extreme cases, static discharges can blow up sensitive silicon chips that are the basis for the integrated circuits used today. That means that static can be a real problem for you. This article will describe what you can do to get rid of static in your environment.

Breeding Static

You'll remember from the first article that static buildup is nurtured by three things: friction, lack of humidity, and dissimilar materials. Now think about the last time that you shocked yourself on a piece of your computer equipment or its surroundings. You were probably sitting in a chair with plastic casters, rolling around on the carpet that is common in most modern, comfortable offices today. There, you have all the circumstances necessary to build up static. First, there's the friction of rolling around in the chair. Next, there are the dissimilar materials between the floor and you: plastic rollers and a synthetic carpet. And finally, in that modern, comfortable office of yours, you're either being cooled by air conditioning or warmed by central heating, both of which drain almost all the humidity from the air. No wonder you have a problem with static. Let's take a look at some solutions to your problems.



Flooring

Of course, floor tile, rather than carpet, is recommended to prevent static problems in the areas where the disk drives will be kept. But, in your offices, you'll probably use carpeting, because it's more comfortable and it's a lot warmer, too! So, let's conclude that in the normal office evironment, most of your computer equipment will be situated in a carpeted area. You can have carpeting and control static by using it! On the market today, you'll find many good grades of antistatic carpet. You can identify antistatic carpet by the metal fibers or clear plastic strands that are embedded in the carpet pile.

When you're making your decision on what to buy, keep some things in mind:

* The carpet should allow less than 2 kilovolts (KV) of static buildup.
* You should buy the best that you can afford, since a better grade can make a difference in this situation.
* Look at the carpet closely. Can you see the metal fibers or clear strands? When you examine it up close, you should see them clearly.

Like we've said earlier, for the best antistatic results, you should install floor tile. This would probably be feasible and acceptable in the computer room, where the most equipment is set up and in use. For the best results, the floor tile should have a maximum resistance of 2×10^{10} Ohms between the floor surface and the building floor.

Furniture

We all use some kind of furniture for computer equipment. Our processors rest on desks, and we use chairs to sit at our processors. In our world today, manufacturers are constantly trying to improve our office lives, by making a better product at a lower price. We bring up this point to help you realize how much plastic is used in furniture today. Plastic rollers on chairs scooting around on synthetic carpet can cause static havoc in your office, as we saw in the first example. Your chairs should have metal rollers, and your furniture should have metal casters, regardless of whether the furniture sits on antistatic carpeting or floor tile. The resistance between any furniture and your floor should be less than 10^9 Ohms.

Humidity

As we've mentioned earlier, air conditioning and heating can dry out the air in your office environment. You might suggest lots of plants. Well, you'd have to have a jungle to make up for the amount of humidity that modern conveniences in our offices take out of the air. The relative humidity of your computer site should range between 40% and 80% relative humidity. Most modern office buildings are sealed from the outside to help reduce the amount of energy required to regulate the inside temperature. In most offices, the relative humidity will often drop to less than 10%. That's quite a drop from 40% relative humidity! And if the humidity drops below 40%, your chances for a static shock increase.

You can use a humidifier to raise the humidity level in your office, though an efficient temperature controlling system will take that humidity out of the air in time, too.

continued on page 7.

RMS DATABUS/DATASHARE: INTERNAL MANAGEMENT

This article provides an overview of the internal operation of RMS DATABUS® and DATASHARE® with concentration on the main areas of memory usage, buffer allocation, and overlay management. All byte figures are assumed to be decimal unless otherwise specified. These figures are valid for DATABUS/DATASHARE Version 1.7.1.

Using the PCR

The Program Communication Region (PCR) slot can also be used for the code window or User Data Area (UDA). The UDA is loaded into the PCR if the UDA requested is small enough to fit into the available memory in the PCR. This depends on how many environments one has, the program size, etc. If it fits, the PCR is mapped into the area between 100K and 110K octal and PCR is also used as UDA. If it is not big enough for UDA for the program, then memory sectors are allocated and mapped into up to four slots. Memory for UDA is dynamic in that it grows upward; however, UDA memory sectors are never released. If you run a program that causes 12K to be allocated and chain to a program that needs 4K to be allocated, the other 8K just sits there.

PDA (Private Data Area) takes up 4K and contains buffers for KEYINs, DISPLAYs, print lines program counter, flags, traps and general data areas.

GDA (Global Data Area) takes up 4K and contains link list headers for buffer chains, printer name, NQDQ pipe name, etc.

The Data Window is where data buffers are mapped in and viewed.

Buffer Descriptor allocation depends on the amount of memory available and the processor type.

This memory layout is duplicated for each DATABUS program run on a workstation. Thus, DATABUS would use a minimum memory of 32K and a maximum memory of 124K if there were that much memory available in a 3800 processor, 188K on a 6640 processor, and 316K on a 8800 processor. The derivation of these figures is discussed in the next section.

Buffer Allocation

DATABUS controls a large pool of buffers by using buffer descriptors. Buffer Descriptors (BDS) are located at the end of the 4K boundaries of the Resident Interpreter, PDA, and GDA. They take up 5 pages, 1 page, and 14 pages, respectively. Each page holds 15 BDS, and each descriptor is 17 bytes long. The BDS contains information such as the Physical Sector Key (PSK) identification of the buffer and page offset. Each code buffer descriptor contains the PSK and information such as the page of the buffer, link list pointers, Logical Sector Number (LSN) and file ID (code buffers) of the current file. A total of 23 pages is guaranteed to be present on the 3800 processor. This translates into a maximum of 345 buffers (=15 BDS/page x 23 pages), which takes up 87K (= 345 buffers÷16 buffers/page x 4K/page). On the 6600 processor, an additional memory sector (giving 16 pages) is provided if available and is mapped between 150K and 160K octal. This translates into a maximum of 540 buffers (=15 BDS/page x {20 + 16} pages), which takes up 136K (=540 buffers÷16 buffers/page x 4K/page). On the 8800 processor, 3 additional memory sectors (giving 48 pages) are provided if needed and available. They are mapped between 150K and 200K octal. This translates into a maximum of 1020 buffers (=15 BDS/page x {20+ 16 + 16 + 16} pages), which take up 256K (=1020 buffers÷16 buffers/page x 4K/page). The more BDS, the larger the Buffer Pool one can manage.

The minimum memory allocation is 36K. The memory is divided in the following manner:

PCR=4K
Resident Interpreter=16K
PDA=4K
GDA=4K
Overlay Area=4K
UDA=4K

The maximum memory requirements are derived as follows:

Memory Area	3800	6600	8800
PCR	4K	4K	4K
Resident Interpreter 16K	16K	16K	
PDA	4K	4K	4K
GDA	4K	4K	4K
Overlay Area	4K	4K	4K
Maximum UDA	16K	16K	16K
Max. Additional BDS	0K	4K	12K
Max. for buffers	76K	136K	256K
Total Max. Memory/ Program	124K	188K	316K

When buffer allocation is requested and no buffers are available, memory sectors are allocated until memory is exhausted or until the BDS are exhausted. It is only at this point that a buffer is retrieved from a free list. DATABUS will grow until is reaches a processor memory size limitation. Then it reuses whatever buffers it may have already. DATABUS will never get smaller. It is possible to have a small amount of memory, run out of memory, and still have BDS left. Space used for BDS will be used for buffers, so that there will never be a surplus of descriptors not used for anything.

It is possible for buffer allocation to conflict with UDA allocation. If one PSK for the UDA is allocated, and the program acquires all memory up to the memory limits for buffers, and then chains to

RMS DATABUS/DATASHARE *continued.*

another program needing UDA, it won't find it. It never releases buffer memory for anything other than buffers.

The buffer pool consists of two parts: data buffers, which are always private (being attached to a file until the file is closed); and code buffers, which are managed on a queue and are discarded in favor of data buffers if the need arises. Code buffers are maintained on hash chains (hashed by LSN) for fast lookup. Code buffers are mapped onto the PCR region between 0K and 10K. Data buffers are mapped into the Data Window between 140K and 150K.

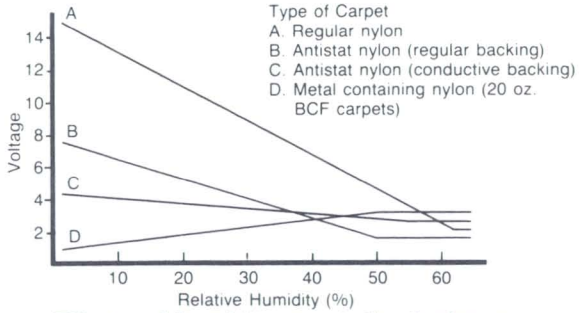
The maximum number of data buffers needed by any program is computed as follows:

1) For a direct and unmanaged file, add the figure given in the STATIC parameter of the file statement or one buffer if DYNAMIC is specified.
2) For each ISAM file, add the number given in the Sectors Per Block (SPB) parameter of the INDEX utility command.
3) For each managed file (via FMS), add 8 buffers (2K) if this file has not been previously opened by any other program.
4) For each pipe, add the figure obtained by adding the number of variables in each COMLST of the COMFILE statement, rounding up to the nearest multiple of 2, and then dividing by two.

Remember, memory requirements vary, depending on your specific needs. Please contact your local Systems Engineer for memory requirements. For information concerning other details of this article, please contact your local DATAPOINT representative. ■

TAKE THE CHARGE OUT *continued from page 5.*

The chart below will help you see how humidity can affect your environment.



Effects of humidity on static discharge.

In the Meantime. . .

Since you must touch the processor to use your system, you probably won't be able to immunize all data processing equipment from all static discharge. As a temporary measure, you can cover existing non-static carpeting with static mats in the area adjacent to the equipment. Although there are many antistatic sprays on the market today, don't use them as a permanent solution, since the sprays must be applied very regularly and may stain carpets. If you miss even one application of the spray, static will reappear, and you might find yourself with intermittent system operation, system failure, or data loss.

At this point, you might be throwing up your hands and saying "I'll never get rid of static!" And you might be right. Although these precautions will certainly help lower the chances of your system becoming a victim of static discharges, you will probably never eliminate static completely. But, if you implement most of the precautions we've listed here, you should be on the way to a less "shocking" environment. ■

A LOOK INSIDE THE LASER PRINTER

How does a laser printer work? This article details just how the DATAPOINT 9660 Laser Printer works, from the inside out.

At a simplistic level, a laser printer works in much the same way as does a copier. The original sheet used for copies made in the copier is replaced by software, electronics, and the laser beam. From that point the toner is applied, and then the image is fixed on the page.

Now you know exactly how the 9660 Laser Printer works, right? There's a lot more to it than that first paragraph implies. Laser printers use lasers, acusto-optic modulators, multi-sided polygonal scanners, sophisticated electronics, photoconductors, toners, and developers. Those are some pretty technical sounding names for some pretty sophisticated parts. To understand the concept behind laser printing, we'll examine how photoconductors and electrophotography work within the copier part of the printer called the print engine. Next we'll see how the laser beam is controlled and shaped by the components of the optics bench. Finally, we'll discuss how the optics bench and the print engine are combined through the use of sophisticated electronics to produce the printing mechanism of the 9660 Laser Printer.

Take a Picture With Your Printer

Electrophotography is a process that is very similar to chemical photography (the process used when you take a picture with your camera). Rather than relying on a chemical film (the negative), electrophotography uses a photoconductor. Photoconductors are made of substances that have varying electrical properties, depending upon the amount of light to which they have been exposed. The photoconductor is placed on a belt of aluminum and then is overcoated with clear plastic mylar. When the photoconductor is charged and not exposed to light, it will maintain a certain amount of electrical charge (potential) between the various layers in the belt. Light, if applied to this charged photoconductor, will reduce the potential that is retained in the exposed areas. Unlike film, photoconductors can be reused many times before they must be replaced.

In the electrophotographic process used with laser printers, the photoconductor moves under a number of locations in the print engine. It is first exposed to a uniform positive charge by the first corona. (Coronas are assemblies in the print engine which use fine wires to apply the charges to the photoconductor.) This positive charge, which is applied in the dark, "erases" any previous latent print image that might have remained on the drum (part of the laser printer that holds the characters before they are applied to paper), allowing the electrophotographic process to start with a "clean slate."

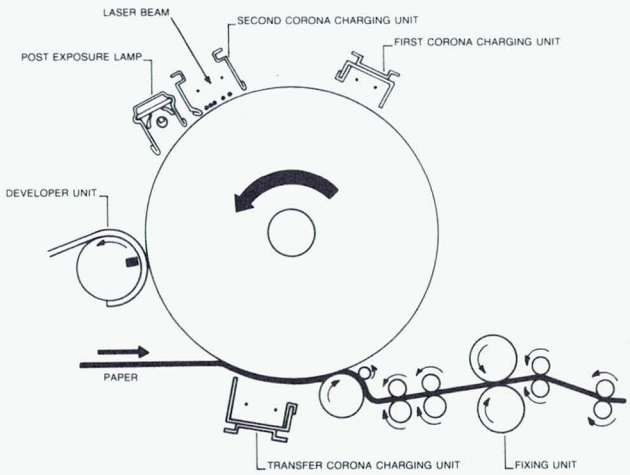
Getting A Charge Out Of It

In the next phase of the process, another (but this time negative) charge is applied to the photoconductor while the laser draws light and dark areas. The areas of the photoconductor that have been exposed to the laser accept a different amount of the charge from the second corona than do the unexposed areas.

At this point, tungsten lights are used to expose the photoconductor. This additional step changes the charges on the laser exposed and unexposed areas into an image with positively (unexposed) and negatively (exposed) charged areas. The 9660 checks the apparent voltage of the surface of the photoconductor immediately after the exposure. This check, in conjunction with special electronics, allows the 9660 to alter the laser exposure to compensate for changes in the drum charging due to temperature, humidity, or photoconductor condition.

What Makes a Character?

To turn the image on the drum into a visable image, toner (a black plastic powder) must be applied to the dark areas of the image. To be attracted to the dark areas of the photoconductor, the toner must be negatively charged. The toner becomes negatively charged in the toner module's developer section, where the toner particles rub with developer particles (iron-based particles). The ratio of the toner to developer is critical for good black coverage; it is continually sensed and maintained by special circuitry. The toner module then moves this combination of charged toner and developer particles with a magnetic brush assembly to the point where they are exposed to the charged photoconductor. The positively charged dark areas on the photoconductor



The Electro-photographic Process

attract the negatively charged toner particle. The negatively charged light areas repel the similarly charged toner particles. This pulling of the toner into the positive areas and the pushing of the toner

out of the negative areas accounts for the dense blacks, the white whites, and the crisp edges on characters with the 9660 Laser Printer. The developer particles, because they are iron based, remain on the magnetic brush.

The photoconductor on the drum continues to rotate. The leading edge of a sheet of paper, which has been moved into place by the rest of the printer, is now tangent to the drum. As the drum continues to rotate, this sandwich of paper, toner, and photoconductor passes over the positively charged transfer corona. The positive charge attempts to pull the negatively charged toner particles through the paper.

The paper is then stripped away from the drum. The sheet travels next to the fuser. The fuser in the 9660 Laser Printer is a device that uses a combination of heat and pressure to melt and press the toner particles onto the paper.

Laser As The Basis

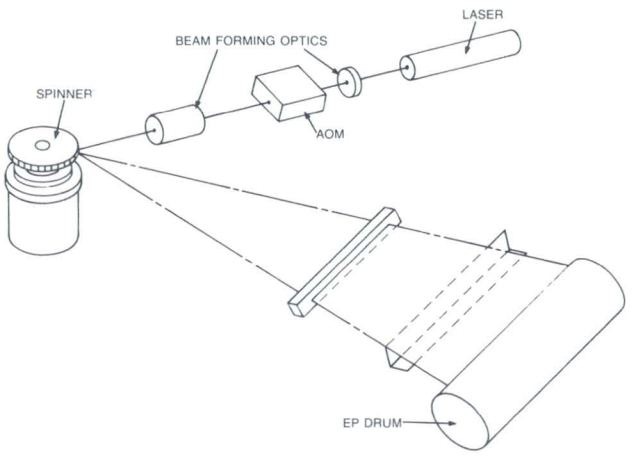
Let's look at the optics bench. It sits on top of the copier mechanism and takes digital input from the system and writes it on the photoconductor. The optics bench is also the source of the controlled laser light source that exposes the photoconductor at the second corona. The laser is used as a light source because it produces coherent light of a single color. The beam of light it produces can be accurately focused and controlled. The laser used in the 9660 is a low intensity, helium-neon device like those used in supermarket scanners; it doesn't burn holes in the paper as you might expect.

The laser is used to scan across the photoconductor, creating the charges that correspond to the black and white pixels that make up the characters. (Pixels are the smallest picture element that you can define, the smallest piece of black or white that you can have.) When a page of paper is printed in the 9660 Laser Printer, the text on that page is imaged on the drum, not as entire sheet, but as a series of extremely fine sequential lines placed parallel to each other on the photoconductor. The lines are composed of black and white pixels. The combination of the pixels and the scan lines ultimately produces the image on the page.

Controlling the Laser

The optics bench of the 9660 Laser Printer can select a transition from white to black pixels or from black to white pixels at any of 480 positions for each inch along the direction of the scan. An acusto-optic modulator (AOM) is placed in front of the laser beam and is used as a high-speed light switch. The AOM takes this information in video form and either diverts the laser beam for areas that are to remain black or allows the beams to pass for those parts of the image that are to become white.

A spinner is used to create the scanning effect. The spinner is a mirror with eighteen faces that is



Scanning the drum with the 9660 Laser Printer

mounted on a motor. The motor maintains a constant number of revolutions per minute. The leading edge of each mirror interrupts the laser beam as the spinner turns; these interruptions create the start of a scan line. As the spinner continues to turn, the angle between the beam and the mirror changes, drawing a scan line. In other words, each time a new mirror face is encountered, a new scan line is drawn. There are additional optics in the optics bench that focus the beam and guide it ultimately to the photoconductor. This scanning occurs as the photoconductor drum is moving. The combination of the moving photoconductor and the spinner causes a scan line to be drawn on the photoconductor every 240th of an inch.

The electronics within the 9660 use two types of information: font information and print information. The font information is down-line loaded from the operating system to the printer. The print information is sent from the unspooler to create the on/off signals that control the laser. Additional electronics ensure that the scanned lines are coordinated with the motion of the print engine and that the paper type requested by the user arrives at the print engine when the print image is ready to be transferred to paper. The delivery of the printed page to the expected output bin is even coordinated by electronics.

All of this sounds rather complicated. In many ways, the process is very complicated. However, the best feature of the 9660 Laser Printer is that you have complete control at your fingertips: you simply give the 9660 Laser Printer commands from your workstation to take full advantage of the laser, AOM, spinner, photoconductor, and the sophisticated electronics to produce top-quality printing in a snap.

For further information concerning the 9660 Laser Printer, please contact your local DATAPOINT representative.■

— STARTUP

In our continuing "Concepts of RMS" column, we're beginning to explore other system utilities used in the Resource Management System operating system. In this article, basically, we'll discuss environments and how they fit into RMS, making it easier to use. The subjects we'll cover are:

Environments
User Environment Table (UET)
ENV utility
STARTUP utility
STARTUP files

All About Environments

To make the explanation of an environment easier, we'll imagine that the RMS computer system, including all of its components, such as disks, printers, terminals, communications adaptors, and RMS networking, resides in a large house. The system components are spread throughout the house: in the kitchen, in the living room, in the bedrooms, and in the dining room. You, the user, want to see some information on one of the disks. The walls of the house are going to block your vision, unless you use something that allows you to see through walls: windows. To see into the house, you would simply put a window in the wall. Voila! Transparent input/output (I/O) is the result. Perhaps you want to see the printer in the next room. No problem. Just put another window into the wall between you and the printer. In other words, you simple establish environments between yourself and the resource you want to "see" or use.

So, you can see that, in our "house," to gain access to a printer, or disk, you provide a window. There are, however, a few problems. The walls of the house have two basic functions: security and support. If we continue to put windows into the walls, we'll compromise both functions. Windows are permanent structures, and there is no security feature that can keep someone from looking through them. This isn't a good idea, particularly if your resource is security sensitive. Keep in mind that the walls have capacity for only so many windows. If you put too many windows into the house, the whole structure will collapse. Too many windows may prove to make access difficult. With windows in every available space, allowing you to see any resource, you'll have so many windows that you won't be able to find the information you want.

Controlling Environments

Now replace all the windows with closed circuit TV. We can look at every resource using cameras instead of windows. On the inside, the house has several remotely controlled closed circuit TV cameras. You, the user, sit comfortably in the control room, where one TV screen corresponds to each TV camera in the house. At the master control panel, you can control every camera/screen system. You can also turn on or off any of the systems, and you can point the remotely controlled camera at any resource in our house. With this solution, we've solved several problems. The camera/screen can be turned on or off, so the resource isn't permanent. You can use a single camera/screen system to look at the resource you want to see; in other words, you don't need a window for every resource in the system. What you do need is one camera/screen combination for every resource that must be accessed at one time. Also, since all of the TV screens are controlled from one point, you won't have problems gaining access to the resources you're interested in; in other words, you wouldn't have to run around the house peering in every window, looking for the resource you want to use.

One important aspect of RMS depends on the pointers that environments provide. Pointers are parts of a computer program that tell the system what information to look at. You, as the user, determine what the environment points to. Thus, the environment you provide is a pointer to the RMS operating system. The environment is not the focus of your attention: your attention is directed, by means of the environment pointer, toward the resource that you want to see. There is one last important point you must understand: environments, unlike our TV cameras, allow for two-way data transfer.

The RMS ENV Utility

Now, with your understanding of environments, we can look at the RMS ENV utility. Your workstation keyboard allows you to control the environment pointer. The RMS ENV utility is the specific tool that allows you to manipulate environment pointers. With the ENV utility, you can add and remove environments, just as you could turn on and turn off the camera. You can tell the environment to point to any computer system resource that you desire (such as disks, printers, tape drives). You can also list all of the current environments, using the ENV utility.

So, RMS efficiently allows us to point to resources we need to see. That's fine, you say, but how in the world can the operating system keep track of all those resources and environments? That's why the User Environment Table (UET) was created. The UET is a table consisting of all your environments. Since you, as a user of RMS, have your own UET, RMS can keep track of the resources for you.

The STARTUP Utility

Now we are ready to examine the STARTUP utility. The STARTUP utility uses the UET as a main part of its function. In the computer, this UET information is stored in main memory as data that may be added, changed, or removed. The STARTUP utility simply copies all of the data from the UET into a work file on disk. This will allow you to stop at a certain point and be able to pick up from that point at some other specified time. This is one function of STARTUP. The other function of the STARTUP utility is to read the data previously saved in the work file and reload the saved UET.

Now, what can you do with STARTUP files? The first and most obvious use of STARTUP files is for user log-on. When you first log on to RMS, the system asks you for your user id. The information entered is used to retrieve a file created by STARTUP and recreate a previously saved UET, which contains all your user log-on information.

Using STARTUP

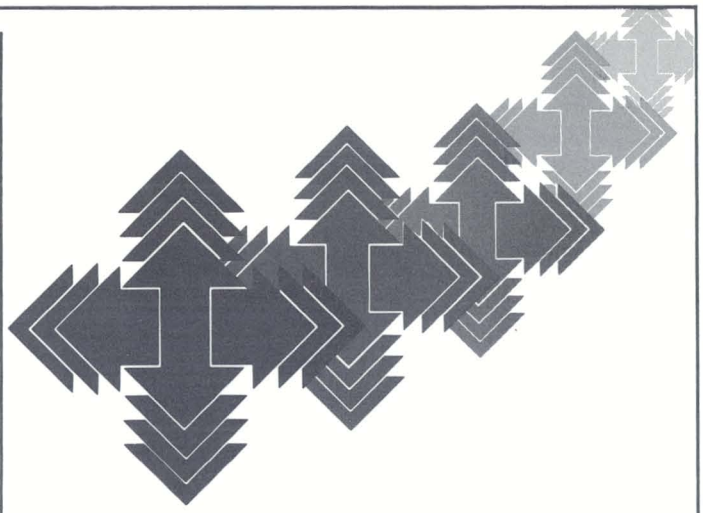
STARTUP can be used for other applications as well. With it, you can insert environments pointing to the resources you need. To perform, for example, a payroll job, the user could enter the command line, "STARTUP PAYROLL." A file called PAYROLL/ENCD would provide the data required to establish the UET for the payroll application to the STARTUP utility.

Another effective use of STARTUP files is more philosophical in nature. After you've logged on to RMS, you may execute STARTUP to insert new environment pointers that were saved from a previous work session. The disk file that STARTUP reads to obtain the UET data may be a file that is accessible to all users. It may also be a file accessible only to you or you and a few carefully chosen co-workers, providing another level of security.

Here is where the philosophy comes in: suppose that your RMS system has ten users who do a variety of similar tasks, such as word processing, Multiplan, program maintenance, and data entry. To gain access to the programs, data files, and other resources (such as printers) necessary to perform these functions, environment pointers must be established, pointing to appropriate locations on disk and to the other resources. For example, unless printers are attached to the workstation, an environment pointing to a printer must be inserted into each user's UET to allow any kind of printing.

Organization is the Key

Using disks, three methods may be used to establish the appropriate organization techniques for disk structure: 1) organize the disk so that all programs and files are accessible to all users at all times by placing all necessary files into a common disk catalog; 2) when the user logs on to RMS, establish environment pointers that point to every resource that could possibly be used; 3) use common



STARTUP files that establish the environment pointers for the next function to be performed. Each method has advantages and limitations.

If you choose the first method to establish environment pointers, you'll end up with all files in one place. Yes, you will know exactly where the information is; however, your environment will be very cluttered, and locating a specific file may be difficult and time consuming. In a way, it would be like placing all your correspondence in one file drawer, using no kind of separation. At first, everything would be simple to find. Eventually, though, the file would become one big pile. One letter would be almost a nightmare to find.

Method 2 is the most commonly used way of establishing environment pointers. The advantage to using method 2 is rapid transition from one process to another, since the pointers are already established for the information you need. This method does have a disadvantage: if additional processes must be performed, more active environments must be inserted. Also, when any resource is moved, every user's pointer to that resource will have to be recreated. If that pointer is not recreated, more information than is usable will be given, and you'll end up wasting a lot of time trying to locate the information you want. Think back to the control room. If you had to turn on 50 cameras to search through 50 screens, information might be hard to find.

The number of environment pointers that may be inserted into the UET is limited. This limit can be easily reached. Since every user must have a unique place on disk for scratch space, every user must have a unique log-on procedure, or STARTUP file. If any system resources are moved, when, perhaps, another disk or printer is added, the STARTUP file for every user accessing those resources must be recreated. (Recall those 50 cameras you'd have to use to search for a resource.) The time required to locate information is directly related to the amount of searching that must be done. Execution time of the RMS CAT utility is a good example of the degredation you can expect if you use method 2.

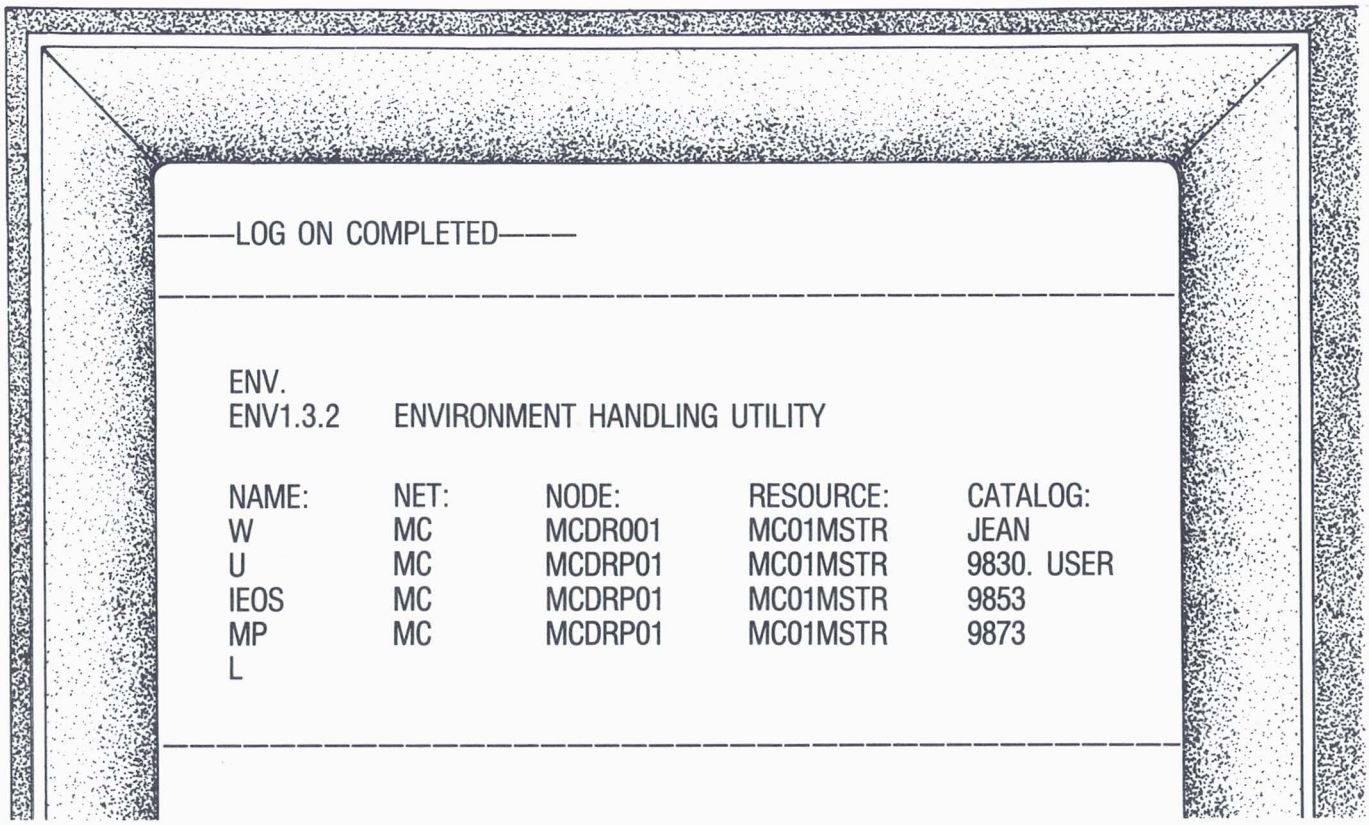
continued on page 12.

An additional disadvantage is this: all environments have user-assigned names. The name may be from one to eight characters in length, and the selection of names is restricted only by the requirement of using proper characters (letters, digits, special characters). Also, the name must be unique for that UET. This may be a problem if more than one program specifies a particular environment name. For example, an accounts receivable system may require access to an environment called ":DATA," which points to the accounts receivable data catalog. The accounts payable system may require access to an environment called ":DATA," which points to the accounts payable data catalog. Thus, you have conflicting environments. You must develop standards and naming conventions for your company, to keep anything like this from happening with your system.

The third method of establishing environment pointers resolves some of problems encountered in method 2 but also has one disadvantage. Using method 3 would mean that the number of environment pointers in the UET would vary according to the process being performed. Let's look at an example using method 3. The user logs on to RMS using a unique STARTUP file. At this point, the UET has environments pointing to unique scratch space (the :W environment), system utilities (the :U environment), and the local resources (the :L environment). The user could then execute the STARTUP utility to insert environments pointing to the resources needed. The user would remove these environments after the task has completed. The next process may be performed using the same STARTUP technique. The advantages to method 3 are that the UET will not get full, since the extra environment pointers are removed at the end of each process. Also, conflict of environment names will not occur for the same reason. STARTUP files located in a common catalog (like the utility catalog) may be used by all users of a process, simplifying the procedures required for moving resources, since only one STARTUP file must be modified. There are, however, disadvantages to method 3: the environments you use for an application must be removed at the completion of that application (using the ENV utility); in addition, more time is required to move from one process to another than is required when using method 2.

So, which method works best? That is difficult to say. Using method 1 (all of the files in one catalog) leads to disk management problems so severe that the use of that structure becomes totally unworkable. Methods 2 and 3 both have strong points, and neither method can be ignored. A combination of method 2 and method 3 will work best. That is, when a user signs on to RMS, some environment pointers other than the :W, :U, and :L may be inserted into the UET, followed by insertion of additional environment pointers when needed.

For details concerning the STARTUP utility and RMS, please see the RMS Users Guides. For additional information, please contact your local DATAPOINT representative.■



Separate environments keep IEOS files and Multiplan files distinct.

ADVANCED ASSEMBLER TIPS

This is the second in a series of articles concerning the Assembler language and some tips in using it at an advanced level. In this article, the author deals with tips on recovering from interruptions, such as an abort. (If you haven't had the chance to see the last article on Advanced Assembler Tips, you'll find it in SOURCE DATA Volume 5, Issue 3 [May/June, 1983] on page 9.)

Some advanced applications systems operating under the DATAPOINT DOS require long-term operation, without being subject to even fairly unusual interruptions. One example of such a program operating at the system level would be the DOS ARC® (Attached Resource Computer®) Network File Processor Monitor. This program should never abort, even in the presence of disk read/write failures, recoverable processor memory failures, or other unusual conditions. An example of an applications system requiring long-term operation with minimum unscheduled interruptions might be a system supporting automated bank teller machines; another example might be a system supporting numerically controlled machinery on an assembly line.

Many programs abort when a sufficiently unusual event occurs. Often, the event might be very unexpected and it must be assumed that the occurrence could make continued operation undesirable (or even catastrophic). Sometimes, the nature of the failure precludes continuing operation as usual and will require manual intervention to resolve.

There should be a mechanism, however, so programmers anticipating (or able to consider) unusual events might be able to develop and specify automated recovery procedures. These procedures could, for example, reduce the frequency of unplanned system downtime.

This article will describe some techniques that can allow a user-written procedure to obtain control in the event of unusual situations that would otherwise result in a program abort. Please note at the outset that many of the techniques to be described may be of limited usefulness or possibly impractical to those applications developed in a higher-level language. Some languages that allow the use of assembler language subroutines can still make use of some of these techniques. Advanced software written at the assembler-language level could possibly employ one or all of these techniques beneficially.

Intercepting the Operating System

Many types of failures that can cause a program or system to abort can be intercepted at the assembly language level by the use of the standard DOS routine, TRAP$. This routine permits the interception of drive offline, disk read/write, disk file write protect violation, and the like. The use of this routine is described in the DOS Users Guide. However, in this article, I'm assuming that you are familiar with (and presumably have made use of) all these normal methods.

Many of the useful routines within the DATAPOINT DOS are made available through specially set aside places in memory called "entry points" or "vectors." Usually, these routines receive control by a program executing a CALL instruction and specifying the entry point of the routine. One or more parameters (providing the operands, shall we say, for the desired operation) are usually specified in one or more registers, with the address of the next instruction to be processed in the calling program stored on the processor's stack. (Again, I'm assuming that you are familiar with the processor architecture used in the DATAPOINT family of processors.)

Often, results from the routine being called are passed back either as values in memory, specific registers, condition flags, or a combination of these. You must understand that many higher-level operating system routines include calls of their own to other, lower-level operating system routines. As a rule, these calls internal to the operating system still use the same entry point vectors as are used by user-written programs. This fact makes it possible to intercept the operating system during higher-level functions and monitor lower-level irregularities that might cause a higher-level operation to abort.

Writing the Routine

Of course, the user-provided intercept routine must be written so that it is benign, and does not lose or modify information that is important to the functions being monitored. Once such a monitoring routine has been created, it is installed by saving the normal DOS entry point in a new place and replacing it with a jump instruction to the user-provided intercept routine.

Such a routine might double-check the parameters being passed to the lower-level routine according to the rules indicated in the DOS User's Guide; the user routine could also just pass control to the regular DOS routine (by using the original, moved DOS entry point jump vector) but set the stack to get control back after the DOS routine executes. This way, an intercept routine can check that any return condition resulting from the lower-level routine is permissible, before allowing it to be seen by the higher-level operating system routine that requested the function.

It is beyond the scope of this article to describe all possible DOS entry points that might be intercepted via user-written routines, and to define exactly what type of parameters are legal and which type of returns are normal. You can find this information in the DOS User's Guide.

continued on page 14.

However, some DOS routines are more useful than others for the purposes described above. Many of these will now be mentioned, along with what sorts of key things can be monitored and with what benefit.

BOOT$ — This entry point, receiving control unexpectedly, means that some sort of fairly severe system failure has likely occurred. However, by this time, most of the damage from the abort may have already occurred or become unavoidable. Although intercepting this routine may be useful as a final effort, usually problems will be uncovered sooner at other entry points.

LOADX$ — This routine is used to load most operating system overlays, significant ones usually having a PFN of less than 7. Intercepting LOADX$ and checking for an attempt to load PFN 5 (which is the DOS ABORT overlay) can tell the intercept routine that a DOS ABORT is about to be processed, even before the DOS ABORT message has been constructed and displayed. Sometimes, this intercept routine can perform some corrective action and restart the failed process.

DR$/DW$/DRW$ — These routines are used for most disk DRW$ transfers, and are especially useful to intercept. The ARC network's magic of accessing remotely stored information uses this intercept as its key way of tricking the disk operating system into thinking the sector being transferred is being done to a local disk unit, just like on a standalone system. It can be useful to both verify the validity of disk addresses and to examine the return conditions to observe if either drive offline or parity error status conditions have occurred. On remote volume transfers under DOS ARC networks, a drive offline could represent a communications, FP, or disk drive failure. A CRC error indicates a genuine CRC error, occurring either at the local disk volume or at the file processor being used.

OPEN$ and PREP$ — These routines can be monitored on entry to verify names of the files to be opened or created, as well as whether they may be on a specified, or any, logical volume. Monitoring the OPEN$ routine, on return, will tell you whether or not a file (maybe a critical one), was found.

ERROR$, EXIT$, DOS$, NXTCMD, CMDAGN — These routines are the most usual program exit vectors. If these intercept routines get control, it means that the program has probably terminated and control is about to be returned to the operating system. ERROR$ and CMDAGN usually suggest that a failure has occurred, while EXIT$, DOS$, and NXTCMD usually suggest normal completion.

WAIT$ — This routine is somewhat of a special case, since it has neither entry parameters nor exit values. The usefulness of intercepting this entry point vector was described in the first Assembler Tips article.

DSPLY$ — This routine can be very useful to intercept. Although DSPLY$ cannot really abort or cause program failure, many programs call DSPLY$ as one of the very first things they do. Intercepting the DSPLY$ entry point can give a way to get control from an about-to-be-invoked program (for example, to install a patch) before the program gets a chance to do very much.

Some Thoughts on Intercept Routines

First, intercept routines must be careful not to use too many stack entries. This is especially true of routines that call the originally desired routine so they can monitor the status it returns. Remember that several different intercept routines could receive control "recursively," resulting in more stack levels being used than normal and therefore an increased likelihood of stack overflow.

Intercept routines (unless otherwise indicated) both receive control and return control in ALPHA mode of the processor, and with interrupts enabled.

An important consideration is where in memory these intercept routines can be placed. This is especially critical on applications written in high-level languages, many of which carefully leave very little memory space unassigned. There is no one good place for all situations. If you are in doubt, DATAPOINT Systems Engineers will either be able to suggest a place, or get suggestions from someone who can.

If an unusual situation does occur, remember that it may not be too easy to recover and continue; often, considerable planning and much attention to details will be required. Many errors are fatal for the very reason that there is no real way to recover gracefully from the fault. ■

DOS.H ARC NETWORK ENHANCES THE 1560

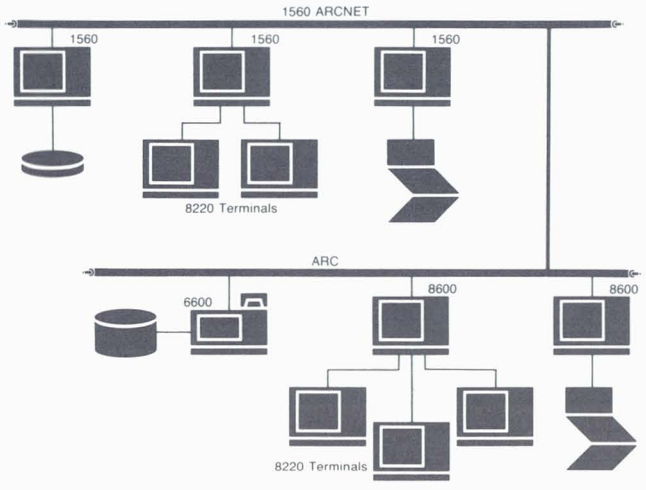
In October of 1982, DATAPOINT announced the newest member of the 1500 series, the 1560 processor. The 1560 is designed for use in data entry, data processing, word processing, electronic mail service, and data communications applications. In the March/April issue of SOURCE DATA (Volume 5, Issue 2), we discussed the differences between the 1550 processor and the 1560 processor. Now, the 1560 is availabe with the DOS.H ARC Local Area Network. The DOS.H ARC network on the 1560 combines new benefits with the best features of the traditional DOS.D ARC.

DOS.H ARC Local Networks on the 1560

The DATAPOINT ARC local network, the Attached Resource Computer Network, is a baseband, token-passing local area network composed of DATAPOINT software and hardware. The physical link of an ARC network consists of coaxial cable, hubs, and network interface chips. A variable number of ARC networks may be linked together for virtually limitless system expansion. The ARC network offers full resource sharing and an extensive selection of software. Now, through the DOS.H ARC15 network, the power and flexibility of the ARC local network is extended to the 1560 processor.

Conjoined Systems, Conjoined Capabilities

In a conjoined DOS.D and DOS.H network, 1560 application processors can now access existing data files on a DOS.D ARC network, since data file formats are completely compatible between DOS.H and DOS.D. A 1560 applications processor must boot and load programs from a DOS.H file processor or a DOS.H local disk. Once the AP is booted, it can mount DOS.D volumes and access all of the data files. This feature allows you to add large scale DOS.D media storage to an existing DOS.H ARC network to accomodate requirements for large or numerous files. The 1560 also allows you to add low-cost, 1560 applications processors to existing DOS.D ARC networks.



CONJOINT 1560 LOCAL NETWORK AND ARC

True Multifunctionality

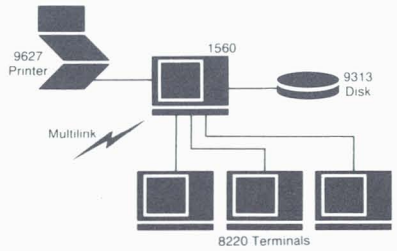
The 1560 supports a variety of languages (such as BASICPLS and FORTRAN) and software packages (such as Multiplan* and IEOS) to meet your processing needs. DATASHARE is available to make your 1560 a multi-user terminal in either a standalone or ARC local network environment.

The ARC local network, DATASHARE, and concurrent jobs under JOB15 all fit into a 1560 applications processor simultaneously, allowing unparalleled flexibility of operation. Communications capabilities include SNA/SDLC and HDLC protocol compatibility, teletype emulation, DATAPOLL® and automatic call answering, as well as multiple terminal and printer support.

The newly available DOS.H ARC network software and the 1560 processor can give you features such as local networking and multifunctional processing in a very cost-effective package. The capabilities of the 1560 provide an easy alternative for starting or adding to your present ARC local network.

More New Features

When combined with the new version of 1500 DATASHARE, many existing DATASHARE applications can be used. You can have up to four users in a DATASHARE system with up to 40 megabytes of disk storage. DATASHARE users can also perform DATABUS programming and data entry tasks at the same time.



1560 — DATASHARE

The extensive array of communications products that have been available for stand-alone 1560s are now available for application processors. In addition, Multiplan, a sophisticated financial modeling software package, is now available in a 1560 networking environment. Workstations can perform word processing, Multiplan, SNA/SDLC, and DATASHARE tasks. (See Volume 5, Issue 2 of SOURCE DATA for more on communications.)

New Features with Printers

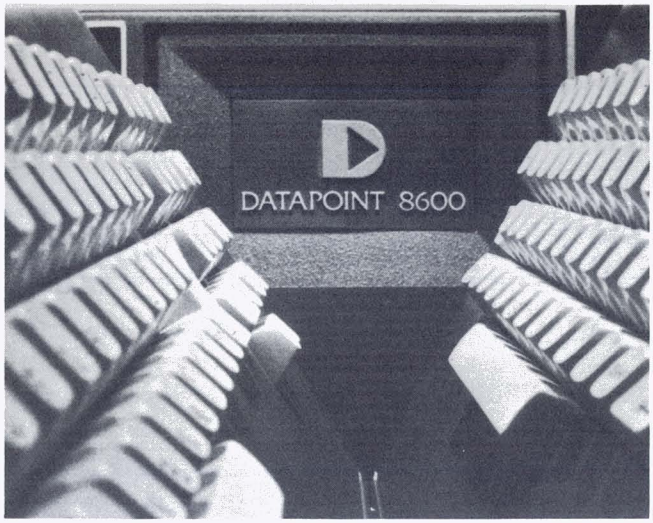
New features have also been added to the printers that can be configured for use with the 1560.

The 9611 Letter Quality Printer, with a print speed of 35 CPS at 10 characters per inch, also has

continued on page 16.

*Multiplan is a trademark of Microsoft, Inc.

THE ENHANCED 8600



DATAPOINT has announced two major enhancements for the 8600 processor: a new entry-level disk configuration, and support for the DOS operating system.

New Entry-Level System

The 8625 disk/diskette system is an entry-level system that includes a 10MB Winchester disk with a 1MB back-up diskette, the high-performance 8602 processor, 128K memory, a serial port for a printer or a terminal, and an ARC local area network interface. An additional 128K of memory can be added.

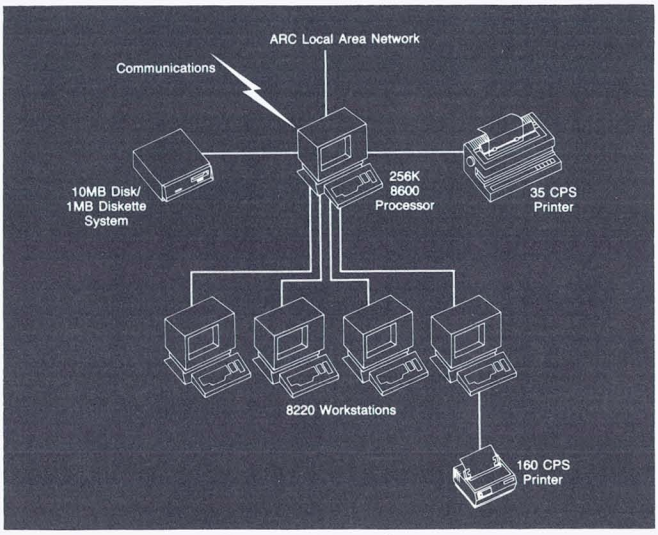
With this memory expansion, additional options are available. A four-port terminal interface, a communications adapter for batch data communication (RMS only), and a Parallel Bus Adapter (which allows attachment of certain communications devices, magnetic tapes, and parallel printers — DOS only) are available options in addition to the memory expansion.

DOS Support

The new DOS support in addition to the previously available RMS operating system will allow existing DOS customers to take advantage of 8600 processing power. Now, you can use an 8600 as a file processor on your DOS ARC network. The DATAPOINT applications software such as word processing, electronic mail, and financial modeling are supported, as are high-level languages such as DATABUS, COBOL, BASICPLS, and FORTRAN. Additional DOS support for the 8600 systems provides an easy move to more efficient systems.

The 8600 Processor

An ergonomic design with amber screen, detached keyboard, and a tilt/rotate base combine to make the 8600 easy and comfortable to use. High throughput rates make the 8600 ideal for office operations like word processing, and the modular internal architecture means memory and capabilities can be added easily and economically. The 8600 systems offer cost-effective dispersed processing workstations with virtually unlimited system growth potential and flexibility of the ARC local area network.



A Typical 8625 System

For further information concerning the 8600 enhancements, please contact your local DATAPOINT representative. ■

DOS.H ARC NETWORK *continued from page 15.*

an optional cut-sheet feeder with two bins. Other features include a variable character spacing for column widths of 132, 158, and 198 characters. In addition, printing can be done on multipart forms, single sheets, and fanfold paper.

The 9628 Matrix Printer has a print speed of 160 CPS in dot matrix mode and 40 CPS in correspondence quality mode, under program control. Program-controlled character spacing is available using 5, 6, 8, 10, 12, and 17.4 characters per inch. Line spacing can be controlled, also through programs, to allow 6 or 8 lines per inch. Page length is variable from 3 to 14 inches. The 9628 Printer can use multipart forms, single sheets, letterheads, and continuous roll paper in widths up to 8.5 inches. Printing can also be done on fanfold paper in widths up to 9.5 inches. The 9628 Printer can be fitted with an optional tractor feed for use with pinfeed forms.

The 9621 Matrix Printer is also available for use with the 1560. Its features include a print speed of 160 CPS at 10 characters per inch, printing on fanfold paper in widths of 1.5 to 15 inches, as well as on multipart and single sheets, and optional international character sets.

For more information concerning the 1560 and DOS.H ARC Local Networking, please contact your local DATPOINT representative. ■

AUTOTYPE SAVES TIME, IMPROVES RESPONSE

Whether it's a political newsletter, a direct mail fund raising appeal, or an inspirational message to your sales force, the added impact of a personalized mailing can make a big difference in your response. The AUTOTYPE function of your IEOS word processing system enables you to create personalized letters by combining standard text and a list of variables. You'll probably use the AUTOTYPE feature most often when you are merging a form letter with a list of addresses to produce a mass mailing.

To use AUTOTYPE, you only need access to three sources of recorded information: the document containing the text, the data file from which the variables (title, name, address) will be selected, and a control file that identifies and specifies the data file. Both the data file and the control file are DOS text files. The document can be any of those in your current library.

Determine the Data File Format

Take a look at the data file we'll use for the actual mailing list. (If you're starting from scratch by building your own data file, you'll want to consult the IEOS Supervisor's Users Guide for more detailed instructions.) Let's say that you are creating the DOS datafile you'll use for the mailing list. Create an IEOS file, and make sure the format uses the following parameters:

```
JUSTIFY LEFT
PAGE LENGTH          1
LINE SPACING         1
PARAGRAPH            1 0
SET TAB
LEFT HEADER
RIGHT HEADER
LEFT FOOTER
RIGHT FOOTER
PAGE NUMBER          1
TOP MARGIN           0
BOTTOM MARGIN        0
LEFT MARGIN          1
RIGHT MARGIN         80
PITCH                10
CLICK                OFF
```

You'll want to use this format, since the file will be output as a DOS file. By setting your top and bottom margin to 0 and your page length to 1, you won't have any extra space that might confuse the situation.

Before you get into the document, set up the tab stops you'll want to use. If you have a four line address (name, company name, address, and city/state/zip), you will need eight fields:

Mr.	John	Smith	ABC Company	123 Main St.	Anytown,	TX	78200
title	*name1*	*name2*	*company*	*address*	*city*	*state*	*zip*

In this case, you could set your tab stops at 4, 10, 20, 35, 50, 57, and 59. Remember, you can set the right margin farther to the right if you need to have more information in your mailing address. Once you have set up your document, simply enter the mailing address using the tab key to space between entries. After entering all the addresses, end the document, and output the document to a DOS file using the command line:

```
output document maillist
```

This command will create a document called MAILLIST/TXT on the first volume available.

What you've created is a list that will be used by the control document to actually do the autotyping for your letter. When you set up the tab stops, you were actually creating fields for your variables. The variable is the information you're going to use for the letter. For example, in the document we've just created, you have the following variable fields:

Title	1-3
Name1	4-9
Name2	10-19
Company	20-34
Address	35-49
City	50-56
State	57-58
Zip	59-63

Keep these in mind as we create the control file we'll use.

Create A Control File

With variable names and column numbers in hand, you can now use IEOS to create the control file. After opening a library, enter:

```
create document control like maillist
```

and press the return key. You'll want to use the same format that you did for the mailing list. Once you're in the document we have some special rules: each line of a control file must begin with one of three keywords: DATAFILE, VARIABLE or COMMENT. These may be abbreviated D, V, and C.

A control file using our earlier information would look something like this:

```
DATAFILE MAILLIST/TXT
VAR      title       1-3
VAR      name1       4-9
VAR      name2       10-19
VAR      company     20-34
VAR      address     35-49
VAR      city        50-56
VAR      st          57-58
COMMENT U.S. Postal Code State Abbreviation
VAR      zip         59-63
COMMENT Use to AUTOTYPE LETTER2
```

DATAFILE specifies the DOS data file containing the information to be used in personalizing the document. In our case, this will be the mailing list, MAILLIST/TXT, that we created in IEOS. VARIABLE lines link each variable name to the col-

umns they occupy in the data file. The variables we determined earlier will fit in this category. The COMMENT keyword allows you to put in explanations and notes.

Remember, each of the variable names must be unique. Each may contain no more than 12 alphanumeric characters: no blanks, punctuation or special characters are allowed. You need to make sure that the column numbers are absolutely correct, since AUTOTYPE will print whatever information is in the columns you specify in the control document.

When you are satisfied that the information is correct, enter the command

output document control

This command creates a DOS file named CONTROL/TXT.

Modify Your Document

The next step in using AUTOPTYPE is to insert the variables in the appropriate spaces of the letter you plan to use with the mailing list you've created. Let's say you have a document named LETTER that you want to use. In IEOS, using the MODIFY command, enter the variable names as you want them to appear in the completed letter, as shown in the following example:

August 1, 1983

title name1 name2
company
address
city st zip

Dear title name2:

As we prepare to enter a new fiscal year, it is...

Now, with the variable names in place, press the command key and use the VARIABLE command to embed the instructions AUTOTYPE will use. Place the cursor at the beginning of the first variable name (title), press the command key, type "variable" or "v", and press the return key. As you press the right arrow key, each of the characters in the word will be highlighted. Be careful not to highlight the spaces between variables. Each variable name must be selected individually so the system can recognize and insert each one.

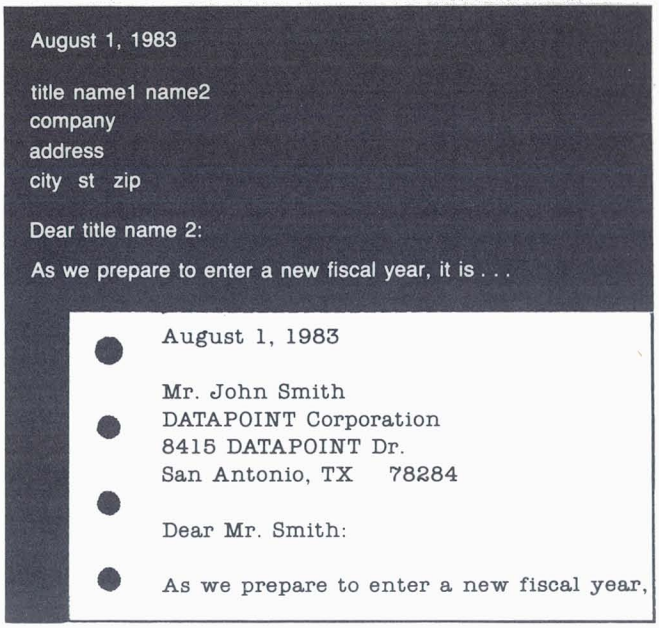
The Finished Product

After you've entered the variable names and selected each one using the VARIABLE command, you are ready to begin AUTOTYPE. To do so, simply enter:

autotype document letter using control/txt

This command line will result in a complete merging of all the records in the data file with the text of the document.

Several options make AUTOTYPE an extremely flexible command. You can autotype a document



other than the current document by specifying the DOCUMENT option using the command

autotype document letter5 using control/txt

The RECORDS option lets you select individual records or blocks of records for autotyping. The record numbers must be separated by commas but they can be in any order. An example would be:

autotype document letter2 using control/txt records 2,7,5,10 to 21,50 to end

If you do not specify a record number, the system assumes that you want to begin at the first record and autotype to the end of your data file.

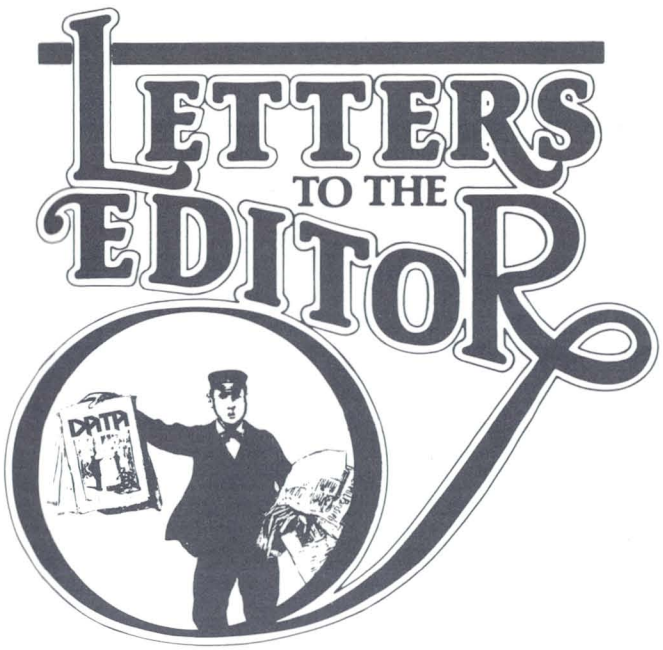
Two final options allow you to control the output of the autotype command. With the ON option, you can choose an output device other than the printer for your letter. For example, you may want to have your document autotyped to the screen so you can see what it looks like before you print on paper. The command line would be:

autotype using control/txt on screen

If you are printing documents with a 35 CPS printer equipped with an automatic cut-sheet feeder, you may specify the paper source in the AUTOTYPE command using the PAPER option. You have three choices: UPPER for the upper paper tray, LOWER for the lower paper tray, or MANUAL which allows you to insert paper in the feeder manually. To have the paper used from the lower paper tray, for example, the command line would be:

autotype using control/txt paper lower

For more information on how to use AUTOTYPE, please refer to the Integrated Electronic Office Station Simplified User's Guide. With a little practice, you'll find that AUTOTYPE can save you time and improve the response on your mass mailings. ■



Madam:

The new DOS 2.7 CHAIN allows considerable programming and decision making in its compile phase. One capability is to stop compilation until a certain time. This could be extremely helpful if, for instance, you wish to start something before going home for the evening and you want the process to commence at a given time, no matter when you start it.

The CHAIN USERPROG, PAUSE, allows the CHAIN compilation to be delayed any specified number of milliseconds. It is written:

//PAUSE nnnnnn where nnnnn is the number of milliseconds to delay. One second delay would be 1000, one minute would be 60000, and one hour would be 36000000.

To enable PAUSE to be compiled in the CHAIN, the compiler must first encounter a USERPROG statement:

//USERPROG PAUSE ("UTILITY/REL","PAUSE")

To set the PAUSE to the right length, the CHAIN must compute the number of hours and minutes left until the desired time. It gets the current time from the predefined data item ARCTIME. NOTE: The value of ARCTIME remains fixed once compilation starts. This means you may not test it repeatedly to wait until a desired time. You must use it only as the start time of your CHAIN.

ARCTIME has a string value of "HH:MM." The substring operators may be used to get the hours and minutes separately:

```
//ASSIGN X=#X#ARCTIME^(1:2)     gets the hours
//ASSIGN Y=ARCTIME^(4:2)        gets the minutes
```

These expressions produce a STRING value. Before you can do arithmetic, you must convert to numeric values. The trick is:

```
//ASSIGN X=-X--
```

These are all you need to write your CHAIN.

```
//USERPROG("UTILITY/REL","PAUSE")
//ASSIGN HOUR=<hour you want compilation to continue; if hour is after midnight, add 24 to get positive delay>>
//ASSIGN MIN=<minute part of start time>
//ASSIGN X=#X#ARCTIME^(1:2)
//ASSIGN Y=#Y#ARCTIME^(4:2)
//ASSIGN X=-X--
//ASSIGN Y=-Y--
//ASSIGN DELAY=HOUR-X
&
& assure a positive delay
&
//IF MIN<Y
//ASSIGN DELAY=DELAY-1
//ASSIGN MIN=MIN+60
//XIF
//IF DELAY<0
//KEYIN "ABORTING: PAST SET TIME"
//ABORT
//XIF
&
&
&
//WHILE DELAY  =0
//KEYIN "OVER #DELAY# HOURS LEFT"
//PAUSE 3600000
//ASSIGN DELAY=DELAY-1
//END
&
& compute remaining minutes
&
//ASSIGN DELAY=MIN-Y
//KEYIN "ONLY #DELAY# MINUTES LEFT"
&
& to within 30 seconds
&
//ASSIGN DELAY=DELAY*60000-30000
//PAUSE DELAY
& . . . . rest of CHAIN follows
```

Louis J. Bookbinder
Software Engineer
Stanford University

Remote ARC Networks

I would like to see an article or literature devoted to capabilities of tying together about 2 or 3 remote sites, spanning the 50 to 75 mile range, running ARC [networks] under DOS. Non-technical write-up (as non-technical as possible), possibly featuring some organization(s) currently having this type of configuration. Considerations for installation of such, Dos and Don'ts of such. Viable alternatives to this type of configuration, excluding for the most part converting to RMS! Communication facilities and considerations to implement,

continued on page 26.

USERS GROUP NEWS

USERS GROUP FORMED IN SAN FRANCISCO

A DATAPOINT users group has been formed in the San Francisco Bay Area. The first meeting was held April 27 at Chevron Research Company in Richmond.

If you are a DATAPOINT user and are interested in being added tothe mailing list for future meetings, please write or call:

Mary Wawrzonek
Chevron Research Company
576 Standard Ave.
Richmond, CA 94802
(415) 620-4685

DATAPOINT COMMUNICATIONS USERS' ASSOCIATION

At the close of the Spring Meeting in Denver on April 18-21, 1983, we became more aware of the adverse effect that the divestiture of AT&T may play on us as Bell System users. Group leaders of LDCS, ACD, Resellers, and Software Products conducted group discussions regarding problems solving and future enhancements.

We are very enthusiastic about our Association and are looking forward to an informative fall seminar in San Antonio in October. These seminars have been excellent learning experiences and provide us means for sharing product knowledge with other users.

We encourage all DATAPOINT users to contact our Membership Director for more information.

Ginny Ferrias
Sundstrand Corporation
Box 7002
Rockford, IL 61125
(815) 226-5187

TRI-STATE USERS GROUP MEETS

On June 28, 1983, the Tri-State Users Group met in its first luncheon meeting at La Veranda in New York. The group listened to a presentation by Rick Sparacino of DATAPOINT Corporation on DATAPOINT hardware. He informed the group about the strengths of each DATAPOINT processor, allowing the users to decide which processor is best for their applications. Karrie Oertli, Corporate Liaison for Users Groups at DATAPOINT, presented information on the new World-Wide DATAPOINT Users Group that is currently being formed.

If you are interested in being part of the Tri-State DATAPOINT Users Association, please contact Ruby Kirk, the group's founder, at 212/754-0473.

HOUSTON USERS GROUP TO MEET

The members of the Houston Users Group will hold a different kind of meeting in July: they're planning a picnic on July 31, 1983. They will meet at the park at 11 a.m. and will eat at 1 p.m. Bring a covered dish, and the users group will provide the drinks. For further information about the place the picnic will be held, please call Virginia Schwartz at (713) 759-0059.

WORLD-WIDE USERS GROUP TO FORM

DATAPOINT and its customers, in a joint effort, are in the midst of forming the World-Wide DATAPOINT Users Group to foster a strong association. At the invitation of Karrie Oertli, Corporate Liaison, Users Groups, representatives of some local groups came to San Antonio June 27-29, 1983 as part of the Formation Committee. The committee elected a chairperson and extended invitations to DATAPOINT users to make up the first Board of Directors. The new Board will decide on a name for the group, incorporate the group into a non-profit corporation, and initiate the first call for members. A conference is planned November 20-22, 1983, in San Antonio. The September/October issue of SOURCE DATA will contain details concerning the conference and membership requirements. Please watch SOURCE DATA and your mailbox for further information concerning your part in the World-Wide Users Group.

CUSTOMER EDUCATION SCHEDULE FOR AUGUST, SEPTEMBER, AND OCTOBER

Below is the schedule for Customer Education for August, September, and October. The Customer Education Course Catalog, Document No. 60269, contains course descriptions, prerequisites, education center information, and cost. Copies are available at your local DATAPOINT office. Enrollment for courses is handled by the Customer Education Center; general questions may be directed to the Center by calling 512/341-3268.

COURSE	LOCATION							
	San Antonio	New York	Boston	Washington, D.C.	Chicago	San Mateo	Seattle	Houston
Introduction to Datapoint Programming	9/19	8/22 10/24		8/8	10/17	9/12		8/8 10/24
DOS Basic Concepts and Operations	8/29 9/26 10/31	8/22 10/3 10/24		9/19	9/12	8/15	10/10	8/1 9/19
DOS Advanced Operations	8/22 9/19 10/17	9/12 10/31			10/24	10/3		9/12
RMS Basic Concepts	9/12	8/1 10/10			10/17	8/1	9/19	8/15
DOS Basic Word Processing	8/1 8/22 9/12 10/3 10/24	8/1 9/19 10/17	8/15	9/12	8/15 9/26 10/7	9/12 10/17	10/31	8/22 10/3
DOS Advanced Word Processing	8/1				9/19			10/17
RMS Basic Word Processing	8/15	9/26		8/29	10/10	8/22 10/31		
RMS Advanced Word Processing	9/26							
DOS DATABUS	8/8 9/12 10/10	8/29 9/26 10/31	8/22 10/10	10/17	8/15 10/3	8/8 10/17		8/29 10/10
RMS DATABUS	8/29 10/10	8/8 10/17		10/3		9/19		
DOS Advanced DATABUS	8/29	10/3						
RMS for New Datapoint Customers	8/22 10/17	8/13	10/24		8/29		9/26	9/26 10/31
RMS for DOS Customers	8/8 9/19 10/31	8/29	10/3	8/15 10/24	10/31	9/26	10/17	
DOS Attached Resource Computer (ARC)	8/22 10/3	8/15						
DOS Electronic Message System	10/17							
DOS Multiplan	9/19 10/31							9/7
DOS SNAP	9/26							
Basic LDCS	8/1 10/31							
Advanced LDCS	9/19							
Basic ACD	8/22 9/26 10/24							
Advanced ACD	10/10							

LETTERS TO THE EDITOR *continued from page 19.*

batch vs. online, software available to support this configuration (DATAPOINT Software) anticipated reponse times, baud rates, etc. Any info of this nature would be appreciated. Thank you.

Daniel L. Thomas
Carnegie Natural Gas
Pittsburgh, PA

Thank YOU! These are excellent ideas for an article. You might check Issue 17, February/March, 1982, page 5. The article, "ARC Communications and Emulation Software," gives you information about the different communication software available from DATAPOINT, including information about baud rates. Volume 5, Issue 1 contains an article (page 1) about Custom Computing and how they use ARC with both DOS and RMS capabilities. Also, in Volume 5, Issue 2, you'll find an article entitled "Installing an ARC System." It gives you the considerations for installing an ARC network. And, in the last issue of SOURCE DATA, Volume 5, Issue 3, two articles, "MLX25B and Switching Networks" and "ARC Network Gateways," provide more information about communications and ARC networks. Keep watching SOURCE DATA for more information about ARC networking.

Customer Education Schedules

I would like to see the Customer Education Schedule advanced at least one month. My SOURCE DATA does not arrive in time for me to be able to schedule anyone for classes. For example, the March/April issue arrived on March 28.

Richard Ries
Okanogan County
Okanogan, WA

In this issue, you'll find an extended Customer Education Schedule. The schedules you'll find from now on in SOURCE DATA will be much more usable. Remember, you can always call the Customer Education Center at 512/341-3268 for more information concerning classes.--Ed.

SOURCE DATA

ADVERTISING INFORMATION

Advertising rates are $22/block. The block below contains the correct number of spaces to fill a one-block ad space (1/30 pg.). Use one space for each character, space, or punctuation mark you want to use. Headlines will occupy the same amount of space as regular text but will be printed in bold type. Camera-ready art must conform to DATAPOINT graphic standards. Failure to conform to these standards will result in a significant delay of publication and may cause the ad to be rejected, solely at the editor's discretion.

Payment must accompany all advertisements. In the case that it does not, the advertisement will be held for publication until payment is received. For publication in the September/October issue of SOURCE DATA, advertisements and payments must be received no later than close of business, July 29, 1983.

Classified as above: $22☐ Display:☐ No. blocks_____Payment: $22 /block $_____

I agree to idemnify DATAPOINT against liability from this ad and that DATAPOINT'S liability for any matter connected with this ad shall not exceed $22.

Your signature_____

Name_____ Company_____

Address_____ City_____

State_____ Zip_____ Phone_____

DATAPOINT

DATAPOINT CORPORATION
9725 Datapoint Drive #T41
San Antonio, TX 78284