SYNTAX 1.6

JAN - FEB 1986

T A B L E O F C O N T E N T S

FIRST CANADIAN ADAM USERS' GROUP
P.O. Box 547 Victoria Station
Westmount, P. Q.
H3Z 2Y6

POSTAGE PAID

Mtl. PQ Permit Pending

For those who have received all previous issues of SYNTAX, (starting with issue 1.1) your subscription expires with this issue. We urge you to renew now because we have many new things planned for year 2. Besides our soon to be released E-Z EDIT program, we are now selling SmartBASIC 2.0. The price is $30 for DP or disk. When ordering this new software, please make separate cheques or money orders. Please allow 4 - 5 weeks for delivery. Look for our in-depth report on it in the next issue. We have heard from many users who have had problems obtaining their fully paid orders from Hi-Tek. After our personal disappointing experience with Hi-Tek, we are pleased to announce that AEC (see ad p.20) has succeeded in acquiring substantial quantities of most of the original Coleco software. We have frequently dealt with AEC in the past and highly recommend them as a reputable supplier to all Adam users. They guaranty delivery within two weeks or less.

Voici ce que j'ai organisé pour que mon ordinateur ADAM, en mode de traitement de textes, puisse me donner un texte français avec mon imprimante. Comme le suggérait le SYNTAX 1.1, je me suis procuré une marguerite avec caractères français. SYNTAX suggérait la Prestige ou la Diablo. Voici les caractéristiques de celle que j'ai choisie: QUME 82078, PICA 10. Je l'ai achetée chez Lauzier et Little, 150 Clément, LaSalle, Qué. H8R 3W1, Tél: (514) 367-3333 ou 1-800-361-7370. J'ai payé environ 7.75 pour la marguerite. L'avantage de cette marguerite, c'est qu'elle comporte des lettres déjà accentuées, ex: é, è, ù, et le ç est déjà formé. C'est une marguerite de 96 caractères mais notre imprimante n'en imprime que 94. Malheureusement, le "à" est l'un des caractères qui n'existe pas. Un autre problème, présent sur toutes les marguerites, c'est qu'elle ne comporte pas de lettres avec l'accent circonflexe.

Pour résoudre ce problème, j'ai changé trois pétales à ma marguerite. J'ai mis l'accent circonflexe à la place de la barre de soulignement au-dessus du 6 sur le clavier. C'est le seul endroit où notre ordinateur tolère que deux signes occupent la "même case" sans qu'il soit nécessaire de changer la programmation. J'ai mis le "à" à la place de "^" sur la marguerite et, enfin, j'ai placé la barre de soulignement à la place du "à" qui n'apparaissait pas. Ce sont les trois seules modifications que j'ai apportées à la marguerite. Il y a quelques autres caractères qui sont différents de ce que nous avions sur notre clavier original. J'ai marqué les touches correspondantes avec de petites étiquettes adhésives.

Partie technique: J'ai coupé les trois pétales spécifiés plus haut en chauffant une aiguille, puis en brûlant le plastique de la marguerite à la limite du cercle de plastique plein au centre de celle-ci. Pour tenir le pétale bien en place durant le collage, j'ai utilisé un papier adhésif (masking tape). J'ai aussi utilisé de la colle (à tuyaux de plomberie) ABS 55Y (ciment à base de dissolvant). J'ai payé 1.89 pour 1/4 de chopine; il y a peut-être moyen de l'obtenir en plus petite quantité. Cette colle se vend dans n'importe quelle quincaillerie. Conseil: faites bien attention qu'aucune colle n'adhère à la partie lisse à l'arrière de la marguerite, sinon il faudra l'enlever avec une lame de rasoir. Bon bricolage!

Jean-Marie Vadeboncoeur, Pierrefonds, Qué.

The ampersand is not intended as a REM shortcut - notice how "?" is expanded to PRINT while "&" remains "&" in a listing. The ampersand is to be used to call a machine language routine whose address bytes are stored at 16132 (while the USR address is right next door at 16130.) Its REM function is a default function. For example, try POKE 16132, 150: POKE 16133, 66: &.

SET SPEED = 0, type GR ?() for some interesting stuff (if you want to, put paper in your printer and leave in the TEXT mode). The part of the <RETURN> routine (chr$(13)) which everything to the right of the cursor upon a <RETURN> is located at 18047. I found this out by using my SEARCH program to look for 58,58,66 (LOAD A with (16954)).

Q: I'd love to modify and fix my BASIC so that my improvements are there already instead of having to be implemented with each BASIC load. But it isn't even in the directory, so how can I find and modify it?
A: Try putting your changes in a HELLO file. That way your changes will be poked in as soon as your BASIC finishes loading. Better yet, get your hands on a tape editing program (we briefly mentioned one in SYNTAX 1.4). We will try to make one available shortly. What this type of program does is read a section of the tape to an area (buffer) in memory. You can then edit this area to your liking and then rewrite it back to tape (or wherever else yoy want to put it). This is how BASIC was modified to run the HELLO file from disk.

BREAK and NOBREAK, SHLOAD, STORE and RECALL: These are statements which are not mentioned in the manual, but can be seen with a memory scan program in the statement table. BREAK and NOBREAK set or clear the CONTROL C abort action during a running graphics program (^C always works during a PRINT). If a graphics program is running when the abort is disabled with NOBREAK, ^C will not stop the program. The benefit is that your program will run 50% faster. Be sure to put in your own short routine or you'll be stuck in that program! SHLOAD was intended to load a shape table, but it isn't even an OS command. It has no known function. Use of STORE and RECALL always produce OUT OF DATA error messages. The format for both is statement <variable>.

Tired of outputting to two sets of screen memory when programming in machine language? Here's a way to turn off the secondary video memory. Type GR or HGR, then CALL 17046. You'll have regular full text, but the characters will be 128 - 255 with color determined at 17126. FLASH, INVERSE, and NORMAL will not work. An added benefit is the faster scrolling of the screen. List or print anything long and you'll see the difference. You now only need to output to primary video memory (address 6144, VRAM). If you output 0 - 127 to the screen, your character color will be determined at 17115 as usual. One warning - TEXT deactivates the graphics statements for the graphics mode used to deactivate secondary memory. CALL 17046 does not. If you PLOT or HPLOT onto the screen locations whose VRAM addresses coincide with TEXT's tables while in 17406 mode, you'll get strange results even though you won't see any graphics appear directly. This demonstrates that the routine starting at 17046 is not all of TEXT.

Addresses 25573, 25574, 25576, 25577 determine the dimensions of the HGR text window just as the four locations given in SYNTAX 1.2 define the TEXT screen dimensions. Loading (registers) HL and DE with the appropriate numbers and then CALLing 17334 is how to set the screen widths. The BASIC routine for the command "TEXT" has been isolated. It starts at address 17046 and ends with 17220. One needs only to find the number 201 (return from subroutine) which is not part of an argument, and directly after it is the beginning of something else (CALL to the beginning address verifies if it is a routine or a table). Typing CALL 17406 is identical to TEXT (in most respects).

Throughout the TEXT routine are LOAD A commands (opcode 62). The POKE 17115: TEXT color routine alters what A is loaded with. Note that address 17114 contains a 62. Among other locations in the TEXT routine that could change the situation are 17068, 17076, 17084, 17099, 17104 (effects unknown as of yet). The effects of the following pokes are known. 17115 changes screen color in TEXT mode (as we have already seen) but has no effect in any of the graphics mode or the border. 17126 changes INVERSE and FLASH colors. 17164 changes screen fill (32 = space; TEXT fills screen with spaces) See screen memory below. 17175 produces the same effect as above but for secondary video memory only. 17207 and 17210 affect the cursor in an uncertain way.
Many jumps to subroutines are made to the jump table located in pages 251-253 of memory which lead to various I/O routines in pages 224-uncertain address (in the OS, where the old pokes don't go.)

SCREEN MEMORY: Screen memory is not one set of bytes like Commodore for example. There are two sets, primary and secondary. To see this POKE 17164,99 ("c") and POKE 17175 ("W"), then type TEXT. You will see W's and c's flashing alternately. Pass the cursor over them and the W's will disappear while the c's remain. This is because byte 17164 holds the character inputted into primary memory, 17175 the secondary character. Passing the cursor over these alternating locations will set secondary memory to the corresponding primary location's bit pattern. This is how FLASH is implemented (notice how the flashing stops when your cursor passes over the blinking locations). The text windows in graphics do not have this feature.

Audio Cassette to Data Pack Conversion

Those of you who ordered E-Z COPY or the first program library on tape have noticed that we used mostly modified Sony audio cassettes as data packs. Due to the high demand for these two items, and because digital data packs are so hard to come by, we had to format our own data packs. Although this type of conversion is possible with most popular brands of tape, it is both time-consuming and a real experience in frustration and patience. Try it for yourself and you will know what I mean.

The data packs designed to be used on Adam are essentially modified audio cassettes. If you compare the two, you will notice the major physical differences in the shell. A data pack has two locating holes on the "B" side that seem to serve two purposes: 1) to prevent you from inserting it incorrectly and 2) to hold it firmly in place in the drive. In addition, a data pack has two less holes in

its shell (at the bottom where the exposed tape travels) to prevent it from being inserted in a tape recorder on either side. By far the most significant differences between a data pack and an audio tape are its internal design as well as the actual format stored on the tape itself. A data pack was engineered to run at high speed with minimum resistant and friction. Audio cassettes on the other hand are designed for much slower speeds. Therefore, even though we are able to convert an audio cassette for use on Adam, it cannot perform as well as a true data pack.

To turn an audio cassette into a data pack is quite a simple task. The only drawback is that you need access to a couple of tape decks or preferably a "dubbing" cassette deck. The ideal procedure is if you can use a high speed tape duplicator. The reason why you need a cassette recorder is because you will have to copy the contents or format of an original data pack onto the audio tape. This procedure will apply to any protected program on data pack you may want to copy as well. But for this type of job, it is much easier to use a back up utility program such as E-Z COPY with a pre-formatted tape or data pack. A back up program will not allow you to format standard audio tapes.

To start the conversion process first you will have to open the data pack and transfer the tape into the shell of the audio cassette. This is necessary in order that it fits into your tape deck. You can drill holes in the data pack so that it fits in the tape deck without being disassembled but it is usually preferable to do it this way. This method does not require you to modify the source tape thus less risk of damage is involved. Once you have safely transferred the contents of the data pack into the shell of the audio cassette, it will serve as your master for future use. Next you will need to duplicate this tape onto a blank C60 audio cassette. This can be a slow and laborious task unless you happen to be lucky enough to have at your disposal a dubbing deck with high speed recording capabilities or ideally a high speed duplicator. A blank data pack is slightly shorter than a C60 standard audio cassette. This minor deviation might cause problems as the entire tape cannot be formatted. To compensate for this rewind the tape past the leader part so that the actual recording begins slightly after the beginning of the tape. When you have finished duplicating the tape, drill the two holes in the shell on the unused side of the cassette and try it out in your data drive.

As you can see from this brief outline, it is not the fast and easy solution to the problem of generating data packs. In fact, if you don't already have a cassette deck or two, it will require an investment of a couple of hundred dollars in such equipment. In addition, you will soon find out that the cheap data packs you intended to accumulate are nowhere as reliable as the original thing. I for one would not trust them to hold my most valuable files. Besides, I'd rather spend my time programming instead of trying to go through this. If you plan to accumulate an extensive library of software you should upgrade to a disk drive especially if you intend to get heavily involved with CP/M. With CP/M a disk drive is more of a necessity than a luxury. Hopefully, an alternative disk drive to the Coleco one will become available so that all users may have the opportunity to own one.

The CP/M Corner

You can modify your CP/M so that it autoloads any command file that you wish. This will give an effect similar to that of the HELLO file in SmartBasic. The only thing that you will need is a tape editing program and the information presented here.

We mentioned that there were tape editing programs available in SYNTAX 1.4 when we explained how to fix up SmartBasic when copied to disk. There are also public domain CP/M editing programs available. In fact, this article is based on the information provided in the DOC file of just such a PD program called DUTIL. By next issue, we hope to make available our tape and disk editor program, E-Z EDIT (the sister program to E-Z COPY). Regardless of the type of editing program you use, these programs will let you edit your tapes and disks whether they are EOS or CP/M formatted.

Once your editing program is loaded, load block 7 into memory. (Note - All the following numbers are HEX numbers.) Scanning through this block, will get you to the Digital Research copyright identification. This I.D. starts at byte 218. Before this notice is a string of 20's starting at address 208 and before this string at byte 207 is 00. If you have gotten this far you are in business. The first thing to do is change the contents of byte 207 from 00 to the number of characters in the command file name that you will be entering. Next you will put the name of the command file. This name must be entered in upper case letters (lower case will not work). I usually enter the hex numbers of the ASCII code for the characters that I am using. Your editing program should allow you to enter the hex, decimal, or ASCII equivalent. I enter it in hex because I find it is faster to do it that way. Finally, you must put a 00 right after the last byte of the command file name. The illustration below shows the area in question. It is a before and after example for autoloading the MODEM7 program which is called MDM7.COM on my disk.

Although this technique works for any command file, it is not always advisable to use it. Microsoft Basic is a case in point. After exiting this program, a warm boot is performed automatically. This makes Microsoft Basic load again. So it can be seen that any program that performs a warm boot on itself upon exiting will get caught in a similar loop. You either take the disk out before exiting or find a way to prevent it from making a warm boot.

```
00 C35CC7C3 58C77F00 20202020 20202020 |C\GCXG¬.         |
10 20202020 20202020 434F5059 52494748 |        COPYRIGH|
20 54202843 29203139 37392C20 44494749 |T (C) 1979, DIGI|
30 54414C20 52455345 41524348 20200000 |TAL RESEARCH  ..|
40 00000000 00000000 00000000 00000000 |................|
50 00000000 00000000 00000000 00000000 |................|
60 00000000 00000000 00000000 00000000 |................|
70 00000000 00000000 00000000 00000000 |................|
```

```
00 C35CC7C3 58C77F04 4D444D37 00202020 |C\GCXG¬.MDM7.    |
10 20202020 20202020 434F5059 52494748 |        COPYRIGH|
20 54202843 29203139 37392C20 44494749 |T (C) 1979, DIGI|
30 54414C20 52455345 41524348 20200000 |TAL RESEARCH  ..|
40 00000000 00000000 00000000 00000000 |................|
50 00000000 00000000 00000000 00000000 |................|
60 00000000 00000000 00000000 00000000 |................|
70 00000000 00000000 00000000 00000000 |................|
```

Adam Graphics: BASIC Game

I hope that you have been experimenting with the BASIC shell given in the previous issue. At any rate, you can type in the game "filling" provided in this issue (see p.19 for program listing).

The game goes as follows; there are 4 bases on the moon. You must land a spaceship at each of these bases. The screen shows the moon landscape with different colors representing the different elevations. You can judge your spaceship's height by viewing the altimeter on the right of the screen. As your ship goes higher the altimeter indicator moves up the screen, behind it are the elevation colors, pass the indicator into one of the colors and your ship will be able to fly over that color on the moon map. Each change in elevation color represents a 2500 meter change. At the bottom of the screen is the instrument panel. It gives your current altitude (in meters), fuel level (in Kilograms), vertical and horizontal velocities (in meters per second), and power (in Mega newtons). The white bordered number to the far right of the screen is the base that your ship should land at next.

You use the game controller to fly the ship which has four side and one bottom rocket. Moving the joystick in any of the 4 directions will thrust the ship in that direction. Watch the 'hvel' indicator when you do this, it gives your overall traveling speed. To stop the ship you must supply thrust in the opposite direction. The left fire button will decrease the power your bottom rocket puts out while the right one will increase it. Changing the power your ship puts out can be done in 1, 5, or 10 unit increments. This depends on the power change settings. If the '1' key is pressed on the keypad then the 1 unit setting is selected. The '2' key selects the 5 unit setting and the '3 ' key selects the 10. As long as your ship is thrusting it is burning fuel. You want to get to the next base with as much fuel as possible. When landing at a base you must land slower than 30 m/sec or else you'll crash. When you land you are given more fuel for the next trip. The fuel you get depends on how soft your landing was. If you can land at less than 6 m/sec then you will get the maximum amount of fuel. A power of about 40 is needed for take-off. The power will not go above 300. The altitude indicator will only display up to 99999 meters.

Try changing the safe landing speeds at lines 340 and 350 to make the game more or less challenging. You might also want to limit the amount of fuel given. Line 130 gives the initial amount while line 370 gives the amount to add after a landing. The moon terrain can also be changed. It is defined in the DATA statements of lines 3800 to 3900. The '@' represents the green elevation, the 'H' the yellow, the 'P' the orange, and the 'X' the red. Changing these DATA statements leave the last two characters of each string as is. This defines the altimeter on the right of the screen. Line 110 gets rid of the cursor while line 650 brings it back. Line 650 also makes a CALL which turns off all sound channels.

In future articles, I intend to get right into the VDP. This will allow us to explore the various graphic modes it has. One possibility is for a 40 column by 24 row text screen.

ADAM HI-RES VRAM STRUCTURE

This brief written by Ted Bik, describes the ADAM's Vram structure in the HGR mode of operation. A diagram of the Vram memory map when the ADAM's HGR mode of operation is invoked is shown in figure 1.

Fig. 1

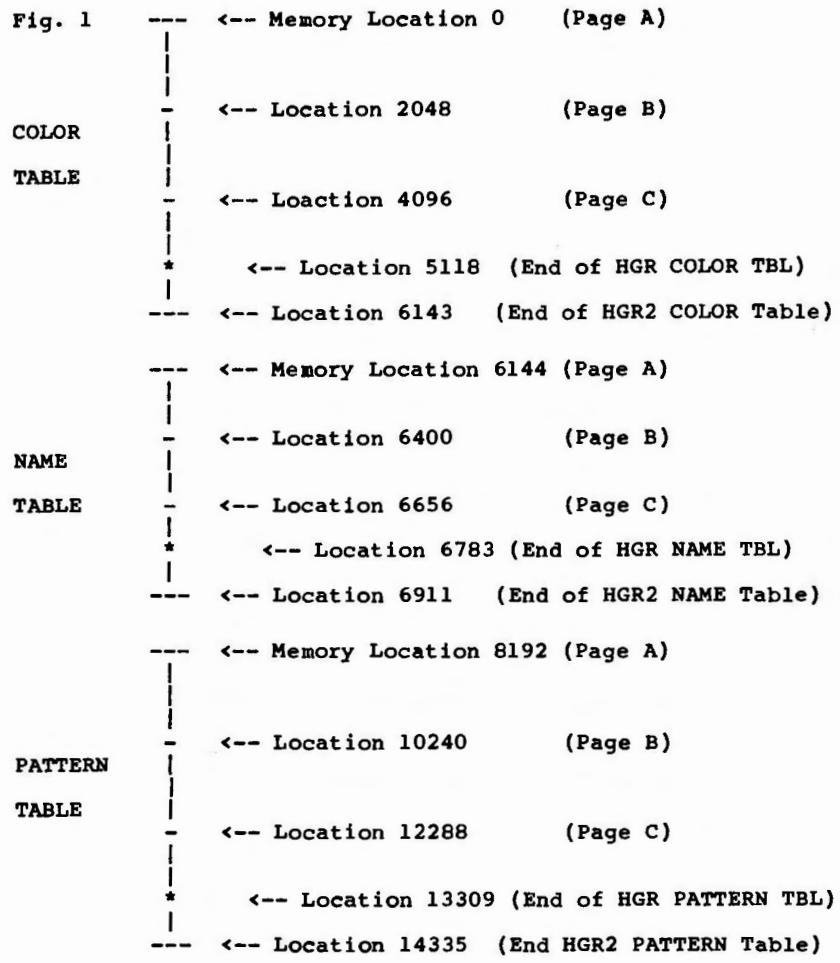
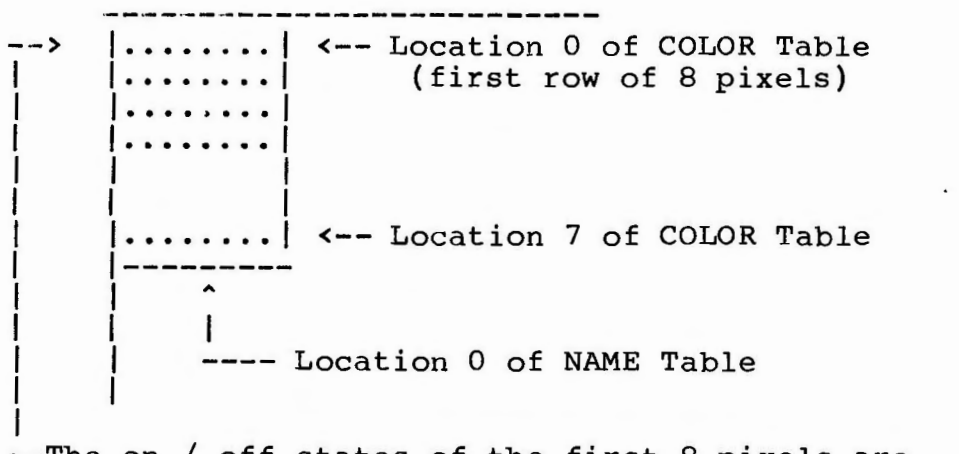


Fig. 2 Video Screen



--The on / off states of the first 8 pixels are stored at location 8192 of the PATTERN Table as a number between 0 and 255.

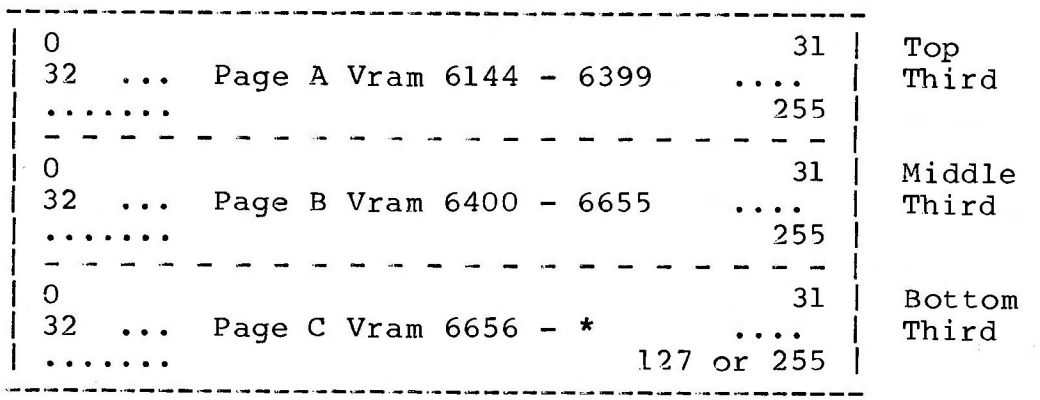
Figure 2 is an illustration of the top left corner of a video screen. The 8 x 8 matrix of dots shown is mapped into the ADAM Vram as described in the following text.

There are three tables in the ADAM Vram. The 8 x 8 pixel matrix as shown by fig. 2 is mapped into each of the three tables. The first table in the Vram is the COLOR table located starting at address 0 through to address 6143. The second table in the Vram is the NAME table. Its location starts at address 6144 through to address 6911. Finally, the third table required to finish describing the 8 x 8 pixel matrix is the PATTERN table. This table exists starting at address 8192 through to address 14335.

Each of the above three tables are divided into thirds. The first third of each table maps into the top third of the video screen while the second third of each table maps into the middle third of the screen. Finally the last third of each table maps into the bottom third of the video screen.

The NAME table is used to identify each of the 768 screen locations when the ADAM's HGR2 mode is invoked. However, in the ADAM's HGR mode the NAME table is truncated at Vram address 6783 to accommodate 4 text lines at the bottom of the video screen. Each location of the NAME table identifies an 8 x 8 matrix (64 pixels) location on the screen as shown in fig. 2 for NAME 0. The value stored in each location of the NAME table specifies the starting location of the pattern and color bytes in both the PATTERN and COLOR tables. Figure 3 illustrates the mapping of the NAME table to the video screen.

Fig. 3 Video Screen



* 6783 When in HGR mode [Final location (NAME) = 127]
6911 When in HGR2 mode [Final location (NAME) = 255]

For example if Vram location 6656 (page C) is loaded with an 8 bit value of 0, the Hi Res shape displayed in the first location of the bottom third of the screen will have the color as identified in the COLOR table addresses 4096 through to 4103 and the pattern as identified in the PATTERN table addresses 12288

through to 12295. Thus it can be seen that the NAME table is used to map 8 bytes of the COLOR table and 8 bytes of the PATTERN table to a specific screen location.

Note also that the NAME table is pre-initialized upon entering the HGR or HGR2 modes. This initialization is as indicated below:

Memory Location	Initialized Value	
6144	0	-|
6145	1	| Page A
		|
6399	255	-|
6400	0	-|
6401	1	| Page B
		|
6655	255	-|
6656	0	-|
6657	1	| Page C
		|
* 6783	127	|
6911	255	-|

* End of NAME table in ADAM's HGR mode.

The COLOR table starts at memory location 0 and continues through to address 5118 in the HGR mode or address 6143 in the HGR2 mode. Again, this table is divided into 3 parts with the first third mapping into the first third of the NAME and PATTERN tables. There is also a corresponding mapping between the second and last thirds of the COLOR table with the respective portions of the NAME and PATTERN tables.

Each memory location of the COLOR table can hold any 8 bit value (0 - 255) which sets the color of a row of 8 pixels of a pattern referenced in the NAME table. The following illustrates this:

Name Table	Color Table Address	Value Stored	Pixel Colors
0 (Page A)	0	240	White on Black
	1	240	White on Black
	2	0	Black on Black
	3	160	Yellow on Black
	4	.	.
	5	.	.
	6	.	.
	7	.	.
-------- End of COLOR table dedicated to NAME 0 ---------			

The above mapping is repeated for page B and C with corresponding shifts in addresses. Upon invoking the ADAM's HGR mode the COLOR table is initialized to all zeros.

The PATTERN table is the third table in Vram and starts at memory

address 8192 through to 13309 in the HGR mode and 14336 in the HGR2 mode. Like the previous tables, the PATTERN table is divided into thirds with each third mapping to the respective NAME and COLOR tables as previously described. Each memory location in the PATTERN table can hold an 8 bit value (0 - 255) which defines the on / off state of a row of 8 pixels referenced in the NAME table. This can be seen below:

NAME Table	PATTERN Table Address	Value Stored	Pixel States (0 = OFF , 1 = ON)
0 (Page A)	8192	32	0 0 1 0 0 0 0 0
	8193	80	0 1 0 1 0 0 0 0
	8194	136	1 0 0 0 1 0 0 0
	8195	136	1 0 0 0 1 0 0 0
	8196	248	1 1 1 1 1 0 0 0
	8197	136	1 0 0 0 1 0 0 0
	8198	136	1 0 0 0 1 0 0 0
	8199	0	0 0 0 0 0 0 0 0

It can be seen that the above pattern stored at location 8192 through to 8199 of the PATTERN table will define the character 'A'. Of course, the ON pixels will only be seen if the corresponding location in the COLOR table for that row of pixels holds a valid color value. (e.g. if the corresponding color value is 0 the pattern displayed will have black pixels on a black background which can be hard to see.)

The following mapping table will clarify the relation between the NAME, COLOR and PATTERN tables:

MAPPING TABLE

NAME TBL	Value Stored	COLOR TBL	Value Stored	PATTERN TBL	Value Stored	PIXEL STATES (. = black) (W = white)
Pg A		Pg A		Pg A		
6144	0	0	240	8192	32	. . W . . . . .
	-	1	240	8193	80	. W . W . . . .
	-	2	240	8194	136	W . . . W . . .
	-	3	240	8195	136	W . . . W . . .
	-	4	240	8196	248	W W W W W . . .
	-	5	240	8197	136	W . . . W . . .
	-	6	240	8198	136	W . . . W . . .
	-	7	240	8199	0	. . . . .
6145	1	8	240	8200	248	W W W W W . . .
	-	9	240	8201	32	. . W . . . . .
	-	10	240	8202	32	. . W . . . . .
	-	11	240	8203	32	. . W . . . . .
	-	12	240	8204	32	. . W . . . . .
	-	13	240	8205	32	. . W . . . . .
	-	14	240	8206	248	W W W W W . . .
	-	15	240	8207	0	. . . . .
6146	2	16	240	8208	248	W W W W W . . .
	-	17	240	8209	32	. . W . . . . .
	-	18	240	8210	32	. . W . . . . .
	-	19	240	8211	32	. . W . . . . .
	-	20	240	8212	32	. . W . . . . .
	-	21	240	8213	32	. . W . . . . .
	-	22	240	8214	32	. . W . . . . .
	-	23	240	8215	0	. . . . .

CP/M: COPY Command Limitations

To copy files from one tape to another, you can use PIP.COM or COPY.COM. The latter file is not a standard CP/M file and is quite superfluous because PIP.COM does what it does and more. It was probably included for the newcomer to CP/M. It's command syntax is a little more English-like and therefore easier to follow. However, COPY.COM has problems. Transferring large CP/M files is an impossible task for it. It can make multiple file transfers, so things like "copy *.* d:" will transfer the file contents of an entire tape to another one without any problem. The CP/M package can be backed up to a formatted tape or disk using this command syntax. However, if you have very large files or just one very large file COPY.COM will not work. What you'll get is a truncated version of your original. So until, or if ever, this problem is fixed, it's probably a good idea to use PIP.COM exclusively if you want to insure proper file transfers.

G A M E R E V I E W S

Electronic Game Pack

Developer: APE Software, 4756 Lalonde, Pierrefonds, Que., H8Y 1V2

We're happy to review yet another good software package from a small Canadian company (See the Martin Consulting reviews in Syntax 1.1 and 1.2 if you haven't already). This time it's a Montreal based company called APE Software. Their first effort is a software package called Electronic Game Pack. It contains computer versions of the following popular games: Backgammon, Battleship, Master Mind, and Tic-Tac-Toe. In addition to these games, a new one called Electronic Miners has been added. EGP is well documented and even contains its own backup program.

When you first load EGP, its menu screen appears showing the 5 game options in addition to the on-line instructions. On pressing the latter option, you are provided with another 6 options. You can either scroll through the game instructions on screen or by choosing the last option, you can get a print out of them. The instructions are all less than one page long and contain the game rules and operating procedures. Since they are not accessible during game play, it's best to get a hard copy beforehand. There are some options (like erasing a move) that are not apparent on the first try. The on-line instructions will probably only be used to refresh your memory on a specific point of the game. Both the rules and operating procedures are quite simple to learn and remember.

All of the games include a musical introduction that is played until any key is pressed, signifying to Adam that you are ready. These are all one player (you vs. Adam) games. You use the hand controllers and/or the keyboard. Only the 3 dimensional tic-tac-toe game gives you the choice of different skill levels. This does not seem to be too much of a liability as some of the games

(backgammon especially) seem to randomly select whether to play a defensive type game or an offensive one. This introduces a degree of unpredicability not often seen in these types of games.

Being computer versions allowed certain enhancements to be made to these games. For example, until playing electronic backgammon, I never thought of being able to keep a score. However this is very easy for the computer to do. Adam just counts the moves away from the endpoint for each piece and adds them up. In Master Mind you not only play against Adam in trying to solve the color pattern, but against the clock as well. Again, Adam makes something that could be done without a computer (but would otherwise be troublesome) easy.

There are a few other things that I like about EGP. One is the excellent graphics used especially on Battleship. Each boat is drawn in quite remarkable detail. When you fire at the enemy's screen, you hear a sound like a bomb dropping and then you see a splash (miss) or a puff of smoke (hit). The Twilight Zone music for Master Mind is great. The new game Electronic Miners deserves some mention. You play on an 8 X 8 grid. The robots alternate moves but can only move like a knight can in chess. You must pick up as many points as you can (each square is worth from 1 to 5 points) on each turn. However, once a screen is occupied you cannot return to it. Thus a large part of the game strategy involves trying to trap Adam so that he can no longer move. The robots are always placed on opposite rows but the columns are randomly chosen. The square values are also different each time the game is played. It's quite an engaging little game.

EGP is interesting in that it appears to be using SmartBasic as its core. The games are sped up Basic programs (much like we have done for EZ COPY) but are not accessible from Basic or word processing. However, being Basic programs inevitably slows down the response time between keyboard and program. Since these are strategy and not action games however, this is not too serious. Keep this in mind though if you have to hit a key twice to get Adam's attention.

All in all Electronic Game Pack is quite a good package. I have not seen any major problems with it so far. Loading time may be a problem on tape but since I reviewed the disk version, I don't have any complaints about this at all. The modular structure of the programs allows for shorter loading time even from tape, but "one minute...loading program" messages would probably be useful. Do not be concerned if the tape stops while the program is loading. The drive will stop only to let Adam "digest" the last batch of information that it entered. I mention this because it is one of the few things that the author of the program mentioned as a possible trouble spot for users. It's not so serious that I was able to notice it. I feel that APE Software should be commended for the fine job that they have done with their initial effort, Electronic Game Pack. I think that most users will really enjoy this package. The games are fun, challenging and absorbing. You certainly can't beat such a combination.

The Best Of Broderbund

Coleco Industries Inc. (1984)

This package containing two Broderbund favorites offers Adam owners two excellent games on one data pack. This package was not widely distributed in Canada so you might have a difficult time locating it. One of these games, Choplifter, was also released on cartridge by Coleco. Both versions are identical.

CHOPLIFTER

If you have ever dreamed of flying a rescue helicopter on a daring mission, Choplifter provides an excellent simulation of a chopper in action.

A terrorist situation has developed and set a serious threat to world peace. A group of 64 U.N. delegates have been captured and are being held hostage in separate barracks by a bunch of fanatic Bungelings. Your misson is to rescue as many of them as possible. You take-off at the U.S. Command Post and head west past the border into the militarized zone. There, waving hands of desperate hostages fleeing in your direction seize your attention. You must touch down carefully to board them in. Enemy tanks approach and try to destroy you. Use your unlimited firepower to fend off their hostile attack as you secure your landing position. Watch out for the enemy jet fighters as they streak through the sky attempting to shoot you down with your rescued hostages. As the hostages are boarding your chopper, the intensity of the attack increases and you must decide at what point to airlift them to safety. Stay too long on the ground and you risk being blown away hostages included. Return to the launch pad and bring the rescued hostages to safety. On your return mission, fire at the barracks to free the remaining hostages. There are 16 hostages in each barrack. Your mission is complete once all the hostages have been answered for. You are given three attempts (choppers) to carry out this dangerous mission.

The attention to detail as demonstrated by twinkling stars, cratered moon, sounds of battle and flying chopper contribute to the realism of the game. Choplifter provides lots of enjoyment in the beginning but it seems to lack the element of challenge and variety that would have set it apart from being just an OK game to a really addictive game. Game play could have been greatly improved had good maneuvers (for example, flying under bridges or through tunnels) been incorporated.

Choplifter is certainly not your typical shoot'em up or maze-type game. In fact, it offers a pleasant departure for a video game where the score is the number of hostages saved and not the number of hits you inflict on the enemy tanks or jets. This is all part of the daring adventure and strategy nature of the game. If you work up the nerve to undertake this rescue mission, we hope that you will learn to command that chopper fast for the sake of those hostages and world security!

A.E. (Anti-Environment)

With the A.E., it was hoped that a breath of fresh clean air could once again be taken thanks to these new pollution-fighting robots. But instead these anti-environment machines have developed a defect making them the major polluters. They're scattered throughout the immense universe and pose a deadly threat to mankind. They must be destroyed before all life forms are infected.

You use a missile launcher to fire at them and drive them back into deep space. The first screen takes place on a weird-looking desert-like landscape (with pyramids and spheres) of a binary solar system. The A.E. come at you in waves and swirl in circular patterns resembling a snake. When firing your missiles, keep the side button pressed longer if the A.E. are further away. Shoot at them and dodge their counter attack. The A.E. move in groups of eight. All eight must be eliminated, before they escape your reach, for a perfect attack. After three perfect attacks, you move on to the next encounter zone. Here the A.E. patterns are somewhat diverse and eliminating them becomes increasingly difficult with time. Each of the four screens (battlefields) present crisp and well-defined graphics. The game ends when all your three missile launchers have been destroyed by A.E. artillery or them coming in contact with you.

The sounds and absurd opening melodies, although low key when compared to other space games, are in fact appropriate for this type of encounter. Excellent graphics and color give the game a visually pleasant dimension unlike some plain and backgroundless space games. You'll find that challenge and interest is thus maintained with each successive level. The only criticism of A.E. is with the restricted mobility (left or right only) of your missile launcher. This game tends to remind me of SLITHER as the A.E. travel in an accelerated slithering motion on the screen. The fly-like movement of the enemy formation in the game GYRUSS also resembles that of the A.E. In that game all the action takes place in the center of the screen and you are able to move your fighter in a complete circular motion.

Both Chopllifter and A.E. are entertaining and somewhat of a departure from the usual categories video games inevitably fall into. Considering that they were introduced around 1982 before the video game explosion intensified, not only have they held their own popularity, but they have helped inspire a number of imitators and offsprings.

*** N O T I C E ***

If you have yet to renew your subscription to SYNTAX, please do it right away. We need to plan our printing requirements for upcoming issues. We will only undertake to send issue 2.1 to those with fully paid 1986 subscriptions. The 1986 rate is $22.00. Due to high printing costs, back issues for 1986 will be $5 each.

Product Review: BASIC MANAGER

Developer: Practical Programs, P.O. Box 244, Kalamazoo, MI 49005

BASIC MANAGER consists of 4 utility programs: BSM, INITitALL, FASTRUN and HELLO. BSM or BASIC MANAGER will display up to 34 locked program names on your screen. It also helps you perform a host of standard as well as slightly advanced tasks at a very user friendly level. In addition to cataloging the current drive, changing screen colors, reselecting the drive to view, running and loading locked files, it permits renaming, recovering backup files, undeleting previously deleted files which haven't been written over, locking and unlocking files, and deleting locked files. All these tasks are performed with minimum effort using the SmartKeys.

An interesting function included is the possibility of recovering deleted files. Since files you delete are only actually deleted when written over, with this feature you can recover those that haven't been affected. The rename, lock and unlocking features offer nothing new just simplied use.

INITitALL is a program designed to optimize the INIT function. Besides correctly initilizing a disk, this program allows you to INITilize a tape or disk which also contains the BASIC program. Plus you have the option of either saving just the BASIC program or erasing the complete contents of your media. This might come in handy if you are into the habit of putting BASIC on all your tapes or disks and in the future want to erase your files only. There is also an option to simply rename the volume ID. Like all other Practical software, this program is also SmartKey driven containing the usual provisions provided for returning to BASIC MANAGER, cataloging or exiting the program to go to Basic.

This original version which we are reviewing also contains FASTRUN V1.0. We have noticed, however, that Practical Programs has decided to sell a revised version of this program separately. FASTRUN is exactly the kind of program anybody who is working with long programs is looking for. If you are working exclusively with data packs you will appreciate its value even more. With FASTRUN you can convert any BASIC program into binary form using the BSAVE command. To run the program you then use the BRUN command. What this means is that your program will now execute 11 times faster on disk and 9 times faster on tape. The problem with growing DATA and REM statements is also corrected when a program is converted to the FASTRUN format. It also helps to reduce the number of blocks used up to store some larger programs.

The HELLO program loads the parallel printer drivers for the SP-1 interface and then BASIC MANAGER. No program reconfiguration for lomem is required because the drivers are loaded in a special area of memory.

BASIC MANAGER along with other Practical software is clearly aimed at making basic utility programs "practical" even for the novice who may have very little programming experience. All the programs contained in this package are useful and easy to use tools which we highly recommend. The decision to make them available separately or as a combined package is also a good idea. Users now have the option of selecting individual programs with more down-to-earth prices as well as more appealing to their individual needs.

Product Review: ADAMCalc - Coleco Industries Inc. (1984)

One of the finest program packages for Adam is the spreadsheet program, ADAMCalc. Granted there are a few mistakes in the manual (a miscalculation of totals on p.12) but all in all, everything from the manual to the sample programs to the program itself is easy to learn and use. And AdamCalc is very, very powerful.

Looking at ADAMCalc, I find that it's not all that different from any other spreadsheet that I've seen. The standard that all spreadsheets are inevitably compared to is the program called VisiCalc, invented in 1978 by Robert Franckston and Dan Bricklin. It was the effort of having to analyze several financial reports each night that led to the development of VisiCalc. What they came up with was a program that looked like an accountant's "spreadsheet" and did essentially the same thing. Numbers are entered column by column, row by row. This raw data is then manipulated by formulas in other areas of the spreadsheet to give the user the required information (averages, totals, etc).

Several things make this electronic spreadsheet idea better than the old pencil and paper kind. First the formulas are placed right on the sheet so that all the user has to do is enter the raw data specific to the case. The answer is then calculated so that the user only sees the result of the formula and not the formula itself. Thus assuming the same type of financial analysis is required for several sets of data, the same framework can be applied many times over simply by entering new numbers over the old ones. Similar to this idea is what is called "what-if" analysis. Let's say you have analyzed a case but now want to see the result of the change of one of the elements of your raw data on some final calculation. You simply write this change over your old number and all of the following changes are calculated automatically. Therefore, the big advantage with ADAMCalc et al is the fact that it can handle large amounts of data quickly and efficiently and that what-if analysis can be performed instantly.

When ADAMCalc first boots up, it does a role call of all your peripherals. If you have any additional peripherals a picture of them will show up on the screen. After this initial screen, what you see is basically an empty grid marked off by row and column indicators. Imagine a blank sheet of graph paper, or better yet, the rows of post office boxes in a postal station. Each box can

have row and column coordinates assigned to it in order to refer to the position of that box. The coordinates are derived from the intersection of a given row and column (e.g. row 2, column 3 would be written as 2,3). Your information is entered individually (raw data, formula, label) into one of these boxes called cells.

The ADAMCalc spreadsheet is a little larger than the original VisiCalc. The work area is 255 rows by 255 columns. However, I have never been able to get anywhere near this range. Taking the on-line help feature off frees up space, but most of the space is taken up by the formulas and screen formatting that you enter in the normal course of things. Keep this in mind when designing a spreadsheet.

Most functions are controlled by the SmartKeys. With these keys you can change screen colors, change column widths, change formatting (how numbers and letters are displayed within a cell, e.g. how many places are shown after a decimal point, whether the label is right or left justified), search the cells for information, sort on a key column. In other words, you can manipulate and display the information in your spreadsheet in just about any way you want.

Other convenient features include several built-in functions that automatically calculate the answer for your specified range of cells. The ability to automatically move the "cell pointer" (the cell pointer is always in the "active cell", ie the cell that you are about to enter information into) either to the right or down after hitting <RETURN>. The most unique feature is that of the built-in calculator. Pressing <WILD CARD> puts a calculator on the screen, so that you can check your figures within the program. If you want to use an answer from the calculator in the spreadsheet, you can use "save value" to do it.

ADAMCalc comes with many built-in functions that make formulas even more powerful. Two built-in functions are especially useful. The "if" function allows you to put one or another result in the active cell dependent on some other condition being true or false. The other useful function is the "lkup" function. This lets you look through a table of values and depending on which number is closest to your input number put that or another number in the active cell.

Other ADAMCalc features include; merging 2 files, creating word processing or spreadsheet files from your spreadsheet, creating other "windows" on the worksheet, freezing certain sections on the screen (hold/release), naming cells, copying cells, and much more. The manual is well written and the explanations for these and other functions are covered in considerable detail.

The 64K memory expander can be used as a printer memory buffer with ADAMCalc. This allows you to print out one spreadsheet while working on another. The only problem is that with the printing being so loud, it's hard to concentrate on the next task at hand.

```
  50 &  Basic Game (Continuation of program found in SYNTAX 1.5)
  60 &
 110 POKE 17529, 0: HOME
 120 br(1) = 2: bc(1) = 3: br(2) = 3: bc(2) = 24
 125 br(3) = 16: bc(3) = 8: br(4) = 20: bc(4) = 22
 130 fuel = 10000: jmpthrst = 10000: base = 177
 200 GOSUB 3000: &  load machine code routines
 210 &  print out map & altimeter onto screen
 220 GOSUB 3600: GOSUB 3700: FOR row = 1 TO 22: READ s$
 225 HTAB 2: VTAB row: PRINT s$: NEXT
 230 HTAB 1: VTAB 23: PRINT "alt.  fuel  vvel  hvel  pwr"
 235 &  randomize and choose base order
 240 INVERSE: r = RND(-PEEK(17011)): FOR i = 1 TO 4
 250 r = INT(4*RND(1))+1: IF temp(r) = 1 THEN 250
 260 IF r = 4 THEN  start = i
 270 HTAB bc(i): VTAB br(i): PRINT CHR$(r+48); : temp(r) = 1: NEXT: NORMAL
 275 &   place sprites on screen
 280 zn = 0: zr = br(start)*8-9: zc = bc(start)*8
 285 xloc = zc: yloc = zr: zs = 0: zh = 1: GOSUB 3300
 290 zn = 1: zr = 167: zc = 228: zs = 1: zh = 1: GOSUB 3300
 300 zn = 2: zr = 192: zc = 228: zs = 2: zh = 8: GOSUB 3300
 305 &     main loop
 310 zc = INT((xloc+4)/8): zr = INT((yloc+5)/8)+1: GOSUB 3200
 320 IF h <= 0 AND zb = 64 OR h < 2500 AND zb = 72 OR h < 5000 AND zb = 80 OR h
< 7500 AND zb = 88 THEN 600
 330 IF h > 0 THEN 390
 340 IF vv < -30 THEN 610: &  crashed into base?
 350 IF vv < -6 THEN  fuel = fuel+vv*167: &   rough landing?
 360 IF vv < 0 AND zb = base THEN  vv = 0: thrst = 0: &  you've landed
 370 IF zb = base THEN  fuel = fuel+10000: base = base+1
 375 IF base = 53 THEN 620
 380 IF h < 0 AND vv < 0 THEN  h = 0: vv = 0: &  can't go below surface
 390 zp = (h < 99999)*h+(h > 99999)*99999: zc = 1: GOSUB 2900: zp = fuel: zc =
7: GOSUB 2900
 395 zp = vv: zc = 13: GOSUB 2900: zp = SQR(xoff^2+yoff^2)
 397 zc = 19: GOSUB 2900: zp = thrst/1000: zc = 25: GOSUB 2900
 400 VTAB 23: HTAB 30: PRINT CHR$(base); : HTAB 31
 405 vol = 255-INT(thrst/20000): GOSUB 3500
 410 k = PDL(13): IF k > 3 THEN 440: &  check keypad numbers
 420 jmpthrst = (k = 1)*1000+(k = 2)*5000+(k = 3)*10000
 440 IF PDL(7) = 1 THEN  thrst = thrst-jmpthrst
 450 IF PDL(9) = 1 AND fuel > 0 THEN  thrst = thrst+jmpthrst
 455 IF thrst > 300000 THEN  thrst = 300000
 460 k = PDL(5): IF fuel > 0 THEN  xoff = xoff+((k = 2)-(k = 8))/4
 465 yoff = yoff+((k = 4)-(k = 1))/4: IF k <> 0 THEN  fuel = fuel-50
 470 xloc = xloc+xoff: IF xloc < 16 THEN  xloc = 16
 480 IF xloc > 208 THEN  xloc = 208
 490 yloc = yloc+yoff: IF yloc <= 0 THEN  yloc = 0
 500 IF yloc > 167 THEN  yloc = 167
 510 zn = 0: zr = yloc: zc = xloc: GOSUB 3400
 520 IF fuel < 0 THEN  fuel = 0: thrst = 0
 530 IF thrst < 0 THEN  thrst = 0
 540 dv = thrst/(5000+fuel)-2: h = h+(vv+dv/2)
 550 vv = vv+dv: fuel = fuel-thrst/2000
 560 zn = 1: zr = 167-(h-(h > 10430)*(h-10430))/62.5: zc = 228
 565 GOSUB 3400: zn = 2: zr = zr+7: zs = 2: zh = (thrst <> 0)*8: GOSUB 3300
 570 GOTO 310: &  end of main loop
 595 &  Final messages
 600 s$ = " crashed into the planet": GOTO 630
 610 s$ = " crashed onto a launch pad": GOTO 630
 620 s$ = "'ll make a model pilot"
 630 TEXT: VTAB 12: PRINT "You"; s$; "!"
 635 PRINT: PRINT "You completed "; CHR$(base-129); " landings."
 640 PRINT "You're final status was:": PRINT: PRINT " Fuel: "; fuel: PRINT " V.
  vel.:  "; vvel
 650 PRINT " H. vel.: "; SQR(xoff^2+yoff^2): POKE 17529, 66: CALL 58321: END
2900 v$ = STR$(zp)+"     ": HTAB zc: VTAB 24: PRINT LEFT$(v$, 5); : RETURN

3800 DATA  "@@@@@@HHHHHHXPPHH@@@@@@@@ XX","@@@@@@@@@@@HHHHHPH@@@@@@@ XX"
3810 DATA  "@@@@@@@@@@@HH@HH@@@@@@@@ XX","@@@@@H@@@@@@@@@HH@@@@@@@ XX"
3820 DATA  "@@@HHHHH@@@@@@@@HH@@HHHHH XX","@@HHHHH@@@@@H@@@@@@@PPHHH XX"
3830 DATA  "@@HHHHHHHHHHHHH@@@@@HPPPPP XX","@@HHPPPHHHHHHHHH@@PPPXXPP PP"
3840 DATA  "@@HHPPPHPHHHHHHHHHPPXPPPP PP","@@HPPPPPPHPPPPPHHHPPXXPPH PP"
3850 DATA  "@@HHHPPPPPPPPPPPHHPXPPPPH PP","@@HHHPPXXXXXPPPPHHPPPPPPH PP"
3860 DATA  "@HPPHPPXXXXXPPPPHHHPXXPH HH","@HPXPPPHPPXXPPPPHHPHPPHHH HH"
3870 DATA  "@HPPHH@HHHPXXPPPHHHPHPPH@ HH","@HPPH@@@@HPXXPPPHHHHHPHH@ HH"
3880 DATA  "@HHH@@@@@PPXXXXPPHHHHHHHH HH","@@@@@@@@HHPPXXXPPHH@@@@@@H @@"
3890 DATA  "@@@@@@@@HHPXXPPPPPHH@@@@@@ @@","@@@@@@@@HPPXXXPPXPH@@@@@@@ @@"
3900 DATA  "HHHH@HHPPPXXXPPPHH@@@@@@H @@","PPPPPPPPXXXXXXPPHHHH@@@HH @@"
```

A P E SOFTWARE PRESENTS...

ELECTRONIC GAME PACK

NOW PLAY EXCITING COMPUTER VERSIONS OF THE FOLLOWING POPULAR GAMES. PLUS ONE NEW ONE ESPECIALLY CREATED FOR THIS PACKAGE. GAMES INCLUDED ARE BATTLESHIP, MASTERMIND, BACKGAMMON, THREE DIMENSIONAL TIC-TAC-TOE, AND ROBOT MINER. THESE ARE ONE PLAYER GAMES THAT LET YOU MATCH WITS WITH ADAM. THE GAMES MAKE EXTENSIVE USE OF ADAM'S SPRITE, COLOR GRAPHIC, & SOUND CAPABILITIES. ALL ARE FULLY DOCUMENTED. WE EVEN INCLUDE A BACKUP UTILITY. ORDER TAPE OR DISK FOR ONLY $29.95 PLUS $2.00 FOR POSTAGE (QUE. RESID. ADD 9% SALES TAX)

A P E SOFTWARE
4756 LALONDE
PIERREFONDS, QUEBEC
H8Y 1V2

PRACTICAL PROGRAMS

BASIC MANAGER V1.1 - Make Basic more user friendly. Special binary type file loads very fast from Basic. Nice looking display shows current drives' free space, volume name, and up to 34 file names on one screen (read from directory). Uses arrow keys to select program and Smartkeys to rename files or volume name, delete or UNDELETE, lock or unlock, catalog (all on 1 screen), recover files, run regular or FASTRUN type files, load regular files, goto Basic or SmartWRITER, reselect drive, and change display colors. Comes with INITitALL, which will correctly init any DP/disk, even Basic! Disk or DP $15.95 US

FASTRUN V2.0 - A memory resident, machine lang. program which converts your Basic program in memory into a form which may be saved up to 3 times faster and reloaded up to 11 times faster (ex. 24 blocks loads in 32 sec on DP). Disk or DP $15.95 US

BASIC MANAGER with INITitALL + FASTRUN Disk or DP $25.95 US

MULTICART BACKUP - Backup most Colecovision compatible game cartridges. Up to 8 on DP or 5 on disk. Special binary type file loads very fast from Basic and makes 100% machine lang., self loading, backups. Loads game title directory loads in a flash. Printout game directory. Disk or DP $10.95 US

ADAMNET CABLES - 25 or 12 foot straight cable from keyboard or disk drive to memory console. 12' = $6.95 US, 25' = $10.95 US

Minimum $10. Orders payable in CAN $ add $2 US and use current conversion rates. Mail CK/MO (No COD's/ Credit Cards), or for free info to:

PRACTICAL PROGRAMS
P.O. BOX 244
KALAMAZOO, MI 49005

*** ADAM & Colecovision are trademarks of Coleco Industries.**

PROTECT YOUR VALUABLE

PROGRAMS WITH

E - Z C O P Y

Backup Utility

Make backups of Basic, SmartFiler, including databases, etc... Copy between single or dual data or disk drives. Backup almost any program on tape or disk. NOTE: This program is intended for making personal backups ONLY.

E-Z COPY is available on data pack or disk for $15. To order send cheque or money order payable to:
FIRST CANADIAN A.U.G.
Box 547, Victoria Stn.
Westmount, PQ H3Z 2Y6

ADAM™ EVALUATION CLUB

ORIGINAL
COLECO SOFTWARE & HARDWARE

CP/M 2.2	$75.00
ADAMCALC	$55.00
SMARTLOGO	$60.00
SMARTFILER	$45.00
SMARTLETTERS & FORMS	$45.00
COLECO HOME LIBRARY	$45.00
RECIPEFILER	$35.00
FAMILY FEUD	$35.00
JEOPARDY	$35.00
DRAGON'S LAIR	$35.00
ZAXXON	$35.00
2nd TAPE DRIVE	$55.00
POWER SUPPLY(no case)	$55.00
ADAM KEYBOARDS	$55.00
ADAM PRINTER RIBBON	$15.00
WHITE HAND CONTROLLER	$15.00

-CP/M 2.2 SOFTWARE AVAILABLE (WORDSTAR, DBASE II, COBOL... Write for Price Quotes

-MEMBERS DEDUCT 10%.
-AND DON'T FORGET AT THE ADAM EVALUATION CLUB WE HAVE MOST GAMES & PROGRAMS ON DP OR DISK FOR YOU TO TRY AT $15.00 EACH.
-MEMBERSHIP IS $10.00.
-POSTAGE & HANDLING INCLUDED
-QUEBEC RESIDENTS ADD 9% TAX
-FOR OUR CATALOG & FURTHER INFO WRITE TO:

ADAM EVALUATION CLUB
3811 PRUD'HOMME #14
MONTREAL, QUE.
H4A 3H8
(514) 487-0110
9.00 - 4.00 EST ONLY