SYNTAX 1.2

MAY-JUNE 1985

TABLE OF CONTENTS

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

Editorial

What's the worst thing that has happened to Adam in the past few months? Most of you will say "Adam was dropped". Well you're right. Adam was dropped. So what?

Of course, we'd all sleep a lot better if Adam was still in production. But it's not all over for Adam users. Far from it. Let's examine the situation more closely. First, we're in pretty good company with the likes of IBM, Texas Instruments, Timex. These well respected electronics companies have all abandoned major lines. And you know, their computers despite being discontinued are all doing well.

The answer for them and now for us, is twofold: users' clubs and third party producers of software and hardware. Get involved locally with other Adam users. There are many knowledgeable fellow users from whom everyone can benefit. Keep a newsletter like this one going. It keeps us all informed and can easily replace a national glossy magazine. Furthermore, support those independent third party producers. They really are our lifeline. This is even more true for us than it is for other discontinued computer lines. Look at it this way. Coleco never did give us, the consumer, very good service. So their departure is not so calamitous. In fact, it opens up the market to others who can very easily do a much better job.

All the things that Coleco promised: RS-232 adapter, 80 column card, programs, will find their way into the market. They are at this time being developped by third parties. And you can bet that SYNTAX will keep you posted on these and other developments. There's still a bright future for Adam - the important thing is to do like TI, IBM, and Timex users, and keep in touch with one another.

On a more positive note, you'll discover in this issue that the revised AdamLink program is now available and finding its way into the happy hands of more and more modem owners. And best of all, you will be glad to know that Coleco Canada Ltd. has finally decided to sell software directly to individuals. For complete details as to availability and price call or write the Montreal office.

The current issue of SYNTAX was a collective effort by: J.D. Moore, Pete Kopystecki, Ron Saunders, and Andrew Wiles.

- - - - -

SYNTAX is published bimonthly by First Canadian Adam Users' Group (FCAUG), P.O. Box 547, Victoria Station, Westmount, Que., H3Z 2Y6. Subscriptions: 6 issues for $20.00. Second class mail registration pending. POSTMASTER: Send address changes and notice of undelivered copies to above address. Return Postage guaranteed.

CONGRATULATIONS!

If future editions of the club newsletter are as helpful as the first one you will only gain in popularity.
You already saved me some grief with your information on the "bugged" early edition of SMARTFILER (WHICH I WAS ALMOST GOING TO BUY) Now at least I can make sure I get the "CLEAN" version!
The knowledge that the current version of the ADAM Modem cannot upload or download will also help me avoid a frustrating experience. Two of our more knowledgeable charter members of the VANCOUVER ADAM CLUB are presently involved with trying to get around that problem.
By far the most valuable information in your newsletter was the availability of back up copies of BASIC. I am paranoid about only having one copy. I don't use BASIC very much but already on two occassions I have had to 'reset' the tape several times before basic would load (causing me many anxious moments).
Again, I appreciate this service....KEEP UP THE GOOD WORK!
P.S >>>>THIRD PARTY PRODUCERS OF SOFTWARE (like Martin Consulting of Wpg.) DESERVE ALL THE ENCOURAGMENT WE CAN GIVE THEM TOO!!

Jack Graham
Maple Ridge,B.C.

Dear FCAUG,
I read with interest your first issue of Syntax and have a suggestion for a future topic to present. It seems that none of the books available in this area cover sound on the ADAM. A short tutorial is all that I would need so I could find the correct area to peek and poke. I did note the ad for Martin Consulting and will probably order Bonanza to get the sound program. Keep up the good work and good luck in this new venture. Are you issuing membership cards?
When using the word processor I find it annoying that the word wrap will not allow the indentation of the line following. To overcome this, I do one of two things:

1. Go back to the end of the line where the word wrap occurred, backspace a few and <return> this puts the end of line marker in, and the next line can be indented.
2. <Insert> <return> at the beginning of the second line, this sometimes puts the end of line marker up into the line above, allowing the indenting of the following line.

The examples 1 & 2 above show when it would be helpful to indent after a word wrap (ie numbered paragraphs).

A.E. Backshall
Crystal Beach, Ont.

ANS: You are correct about the lack of information concerning sound but I hope you will gain some insight from the sound article in this issue. And no, we are not planning to issue membership cards at the present time. Lastly, we don't understand why so many people have trouble with the word processing problem that you outlined. If you look on p.13 of the word processing

manual, section 3, you'll see a description of what the wrap-around feature does and consequently when to use the <return> key. You hit a <return> key to end a line prematurely and to create a blank line. That part is easy enough. Perhaps the hard part is the last reason, "before an indented line". Why? Because if you don't our old friend Mr. Wrap-around will pull your indented text back to the first column trying to do the job it was instructed to do. Thus you tell wrap-around not to pull text over to the left by hitting <return> on the line previous to your tabbed line.

This leads us to the problem of trying to get a <return> into a line that you've already written on. The author of the above letter gives 2 solutions I can offer a third. Simply <insert> a <return> at the end of the previous line. <Return>s are very easy to put in on "virgin" territory (ie. space that you have not typed before). If you go back to a line and try to do this, you have to <insert> or <backspace> your <return> in. This is true even if you erase the letters on a line. This erased line is not virgin territory even if it looks it. To tell for sure: if you can traverse an area with your arrow keys, it is not virgin territory. Its all related to the wrap-around feature. Adam insists on wrapping-around any non-virgin territory. Just keep this idea in mind and you should never have problems of this nature again.

Dear Syntax,
I had bought a Victory cartridge, and have had it for about six months and have never seen a scout or a quark out of all the times I have played it. I keep playing it thinking you have to get a high score to see them or have to play skill levels 2-4 in order to see them but never have. And Coleco wants $15.00 to fix it, but will only replace it anyway. I don't think that I should have to pay for it. What should I do?

Dale Bickerdike
Saskatoon, Sask.

ANS: Don't do anything Dale. We phoned Coleco and found out that scouts and quarks don't exist; that is, the manual was written prematurely and that they were never put into the program. And while we're on the subject, you might be interested to know that the same is true for the Buck Rogers game. The pause button does not work as stated in the manual. In fact it doesn't work at all. So you can stop looking for that one too. (We're sure that you would have found it sooner or later!)

Dear FCAUG,
Your letter comes as a welcome surprise and I am pleased to enclose my membership fee and application. I have recently been experiencing some of the frustrations which you described and was on the verge of writing to Coleco to try and get some information about software, etc. myself.
I have one question for now which you might consider for future

issues of your newsletter. I was trying to write a program which generates random lottery numbers (6/49) and discovered that the numbers which my Adam generates are really random numbers in a set sequence. Each time the program is run, the same numbers appear in the same order. Perhaps there are other ways of doing this.

G.C. Taylor
Vancouver B.C.

ANS: Random numbers on Adam are a bit of problem. They do, in fact, generate the same sequence of numbers every time a positive argument is used. According to Coleco, "this arrangement is consistent with ANSI standards, but it differs from Applesoft Basic which produces a new random number each time the program runs". There is a way around this problem but first we have to take a look at how the rnd function works.

Imagine a column of numbers from top to bottom. These numbers are between zero and one. Each time you ask for a random number and the argument in the brackets is positive, a number from the top of the column gets picked. The list is sequential and then incremented after every implementation of the statement. Thus when the random statement is called upon again the next number in the next "row" is picked. You stay in the same column as long as your argument is positive.

The sign of the argument is important not the value. If you use a negative argument, a new "column" of numbers is picked. You start from the top and as long as your arguments are positive you sequentially go through this new list just as you did for the old one. You don't notice it but as soon as you boot SmartBasic, a default "column" is chosen for you. To test this boot Basic, now get a random number. Repeat the process. The number will be the same all the time because SmartBasic always starts from the top of the same default column. My number is ".732004777". So is yours. Now when you switch columns (with a negative argument), you also follow a set sequence of numbers. Every time you start from the beginning of this new column you will start with the same "random" number.

Obviously the only way to get a truly random number is to
1) randomly switch to another column, and/or
2) randomly stop along a column (ie at a random row). The second method is a little clumsy. So the first method is generally used. If you've been following the row - column analagy: we find it easier to randomly pick the first row of a column than to stay in one column and randomly pick a row.

Look at the accompanying program. In answer to your question it is a LOTO 6/49 program. Lines 100-130, pick the random column. The main body of the program picks 6 numbers from 1 to 49 (lines 140-200) and redoes any repeated numbers so that each number is different. Since the timing loop randomly picks a new column, you will be getting a different sequence each time you run the program. I hope this answers your question.

```
 10 HOME
 20 HTAB 10: VTAB 10: PRINT " LOTO 6/49"
 30 HTAB 10: VTAB 12: PRINT "ADAM will pick"
 35 HTAB 10: VTAB 13: PRINT "6 random numbers"
 40 VTAB 17: HTAB 3: PRINT "press right fire button to"
 50 HTAB 14: VTAB 19: PRINT "begin"
100 FOR x = 1 TO -5000 STEP -1
110 IF PDL(9) = 1 THEN  GOTO 130
120 NEXT x
130 x = RND(x)
140 DEF FN f(n) = INT(1+n*RND(1))
150 FOR n = 1 TO 6
160 s(n) = FN f(49): NEXT n
170 FOR a = 1 TO 5
180 FOR b = (a+1) TO 6
190 IF s(a) = s(b) THEN  s(a) = FN f(49): GOTO 170
200 NEXT b: NEXT a
205 TEXT: VTAB 5: HTAB 10: PRINT "LOTO 6/49"
210 FOR a = 1 TO 6
220 HTAB 15: VTAB 10+a: PRINT a, " "; s(a); " "
230 NEXT a
235 VTAB 20: PRINT "press any key for new set"
240 VTAB 21: PRINT "press control-C to stop"
250 GET c$
260 IF c$ <> CHR$(3) THEN  GOTO 150
```

MORE POKES

We're glad to say that last issue's "Color Pokes" article met with a good response. In this article we will tie up a few loose ends with color pokes and then go on to look at other pokes and see what they do. Specifically, we'll look at another color routine, a poke to change the cursor shape, pokes to change the size of the text window, and finally how to program in sound on the Adam.

Color: You may have noticed that there was no figure one in the Color Pokes article and therefore no program to examine and run. An obmission, to be sure, but not a serious one because the logic of the program was not too difficult to imagine. It was nothing more than the color poke statement within a loop. I used this program to examine all the color combinations and to match them up with the decimal number used to get them. So for the sake of completeness, I present it on p.9 (yes, figure one!), as well as two similar programs sent in by 2 fellow FCAUG'ers. The other programs are by R. Gagne of St Laurent, Quebec and Jean-Marc Roy of Touraine, Quebec respectively.

Here's another poke to try. Poke 17059, x:text. This poke will not change the foreground color but will change the background and border color. When used in conjunction with poke 17115, x: text, this poke will be over-ridden in the background area, however, the border will remain the same.

Cursor: For those of you who didn't get a close look at D. Lelievre's "Adam: Screen Work-Out" in the last issue, you had better read this. By poking in the ASCII value of any character that you want into address 16953, you are able to change the cursor to that requested shape. The author poked in a 32 (blank) in order to stop the cursor from hopping all over the screen

while in the graphics mode. (Game designers should take note of this technique. It cleans up your screen considerably.) Poke 16953, 95 replaced the familiar flashing bar cursor. But if you want a flashing cross (+), poke 43. How about a flashing "9"? Try poking in 57. Just look at the ASCII character code table on pp. C-12 to C-15 to help you decide on which shape you want to use. Note that poke 17529,00 will make the cursor disappear and poke 17529, 66 will bring it back again.

Text Window: We already mentioned the text window when we discussed the color pokes. Now we'll learn how to change not the color but the shape of the text window. This technique can be used for at least two things that immediately come to mind. Decreasing the size of the text window adds emphasis to programs. The "overscan" problem is immediately corrected.

The overscan problem is simply this. On many TV's you can not see the extreme left or right columns. The picture is too wide for the TV. Sometimes a TV repairman can correct the problem by re-adjusting the screen width. But if you don't want to go that route you can correct the problem yourself. You do this by decreasing the width of the picture that the computer sends out.

Four addresses delineate the limits of the text screen. The right margin (number of columns) is set at 30 at address 17199. The left margin value is at 17202, default value 1. Changing these numbers will change the number of columns that are presented to you on the screen. Note that you can only get smaller; strange things will happen if you don't. The top margin (default = 0) is set at 17201 and the bottom margin (# of lines) is set at 17198, default value = 23. Now you know how it is that Adam displays 30 columns and 24 lines. Note that the form is <poke address, value:text>. The "text" is absolutely necessary.

Sound: Besides the lack of information on how to use color in Basic, stands an even more glaring deficiency. That is, how to program in sound on the Adam. "Adam's Companion" is the only public source of this kind of information I am aware of so far. It is also rather difficult to find at this time. So we at FCAUG feel a short article on sound would be beneficial to those of you unable to get this information.

Follow along with the attatched program. It is a stripped down program with all the basics on how to get a simple sound out of Adam. First you increase the protected area in Basic with lomem: 29000. This prevents the compiler from using this space as a work area. Next you must put in the frequency and duration of the note requested. The loudness is specified within the program. This value could be inputted but you can do that on your own if you wish.

The next section "sound assembler" is a small machine language routine used to access the sound chip once all the sound parameters have been calculated. It does this by loading the Z-80's register A (58) with the memory contents of address 28006

(102, 109). This address contains the parameters that you specify in the input statements. These parameters are calculated to be understandable by the sound chip in the next section. Next you tell the chip to send this number to the sound chip or more accurately output to port FFh (211, 255). You then return to Basic with the return symbol (201). So this section effectively lays down the path that the information will flow in order to activate the sound. The next section calculates what value will be sent by this section of the program.

The last section "main sound program" takes your inputted parameters and manipulates them in order that they can be understood by the sound chip. This section translates your input to sound chip input in anticipation of that transfer. The calculations for variables pitch, second and first are all made in order to get the frequency to the sound chip. First and second are then poked into address 28006. Next the loudness is poked into the same address. The number 144 is full loudness and the last poke (after the delay) 159 is for no loudness. Numbers between these two extremes give various volumes. The delay is used to control the duration that the note is produced. Keeping with standard music notation the duration is varaiable, ie. dependent on tempo. You can change the tempo by changing "20" to any number you want. The last statement goes back to input more notes. To exit from this program you must press ^C.

```
  0 LOMEM :29000
  9 &    **********************
 10 &    input sound parameters
 11 &    **********************
 20 INPUT "pitch = "; pitch
 30 INPUT "duration = "; duration
 39 &    ***************
 40 &    music assembler
 41 &    ***************
 50 FOR address = 28000 TO 28005
 60 READ value
 70 POKE address, value
 80 NEXT address
 82 &  Assemby code for sound port access routine
 84 &  LD A,(6D66h)
 86 &  OUT (FFh),A
 88 &  RET
 90 DATA   58, 102, 109, 211, 255, 201
 99 &    ******************
100 &    main sound program
101 &    ******************
110 pitch = 3597000/(32*pitch)
120 second = pitch/16
130 first = 128+(pitch-second*16)
140 POKE 28006, first: CALL 28000
150 POKE 28006, second: CALL 28000
160 POKE 28006, 144: CALL 28000
170 FOR delay = 1 TO duration*20: NEXT delay
180 POKE 28006, 159: CALL 28000
190 GOTO 20
```

```
10 TEXT
11 t = -1
20 t = t+1
22 IF t > 255 THEN  GOTO 120
25 POKE 17115, t: TEXT
30 HTAB 9: VTAB 9: PRINT "POKE for this"
40 HTAB 11: VTAB 11: PRINT "color is:"
45 INVERSE
50 HTAB 9: VTAB 15: PRINT "POKE 17115,"; t
55 NORMAL
60 HTAB 4: VTAB 22: PRINT " Press any key for next"
70 HTAB 3: PRINT "color or<q> to quit"
80 GET a$
90 IF a$ = "Q" OR a$ = "q" THEN  GOTO 120
100 GOTO 20
120 POKE 17115, 240: TEXT: END
```

```
 1 &  Figure 1
 5 &      color - number conversion
10 FOR x = 0 TO 255
20 POKE 17115, x: TEXT
30 FOR n = 1 TO 1000
40 NEXT n
50 FOR i = 1 TO 50
60 PRINT x
70 NEXT i: NEXT x
```

```
10 PRINT TAB(5); "PROGRAM POUR CHANGER"
11 PRINT TAB(5); "LA COULEUR DE L 'ECRAN"
12 PRINT TAB(5); "(tapez 0 pour sortir)"
13 PRINT
15 PRINT TAB(2); "Noir sur"; TAB(19); "Blanc sur"
16 PRINT TAB(2); "fond... "; TAB(19); "fond..."
17 PRINT
25 b = 0: a = 0
30 b = 17: a = 241
31 DATA      "noir","vert","vert-p", "bleu",
32 DATA      "bleu-p","rouge-f","aqua","rouge",
33 DATA      "rouge-p","jaune","jaune-p",
34 DATA      "vert", "pourpre", "gris",
35 READ c$
40 PRINT TAB(3); c$; TAB(11); b; TAB(19); c$; TAB(27); a
50 b = b+1: a = a+1
55 IF b = 31 THEN  GOTO 65
60 GOTO 31
65 PRINT
70 PRINT TAB(2); "CHOISSEZ LE NUMERO DE VOTRE"
71 PRINT TAB(2); "COMBINATION DE COULEURS";
75 INPUT c
76 IF c = 0 THEN  END
80 POKE 17115, c: TEXT
```

```
100 LOMEM :30000
110 TEXT
120 readvram = 28000: writevram = 28014: buffer = 29000
130 &     poke in machine routines
140 FOR i = 0 TO 27: READ code%: POKE readvram+i, code%: NEXT
142 &   Below is the assembly code of the 2 machine routines
143 &   read 8 byte of VRAM
144 &   LD HL,7148h `HL points to buffer in RAM
145 &    LD DE,(6D92h) `DE points to VRAM address
146 &   LD BC,8 `8 bytes to send
147 &    CALL E01Ah `read_vram
148 &   ret ` return to basic
150 DATA     33,72,113,237,91,146,109,1,8,0,205,26,224,201
152 &   write 8 bytes to VRAM
153 &   LD HL,7148h
154 &    LD DE,(6D92h)
155 &   LD BC,8
156 &   CALL E000h `write_vram
157 &   ret
160 DATA       33,72,113,237,91,146,109,1,8,0,205,0,224,201
170 &     get a character to work on
180 INPUT "input a character : "; c$
190 c = ASC(LEFT$(c$, 1)): &     get ascii code of character
200 &     compute the video address of the character's pattern bytes
210 vadd = 0+8*c
220 PRINT "ascii "; c; " starting pattern address : "; vadd
230 &       place video address where machine routines can get them
240 vhi% = vadd/256
250 vlo% = vadd-vhi%*256
260 POKE 28051, vhi%: &       high part of address
270 POKE 28050, vlo%: &        low part of address
280 CALL readvram: &      read 8 bytes into buffer
290 &     flip the pattern bytes in the 8 byte buffer
300 FOR i = 0 TO 3: top% = PEEK(buffer+i): bot% = PEEK(buffer+7-i)
310 POKE buffer+i, bot%: POKE buffer+7-i, top%: NEXT
320 &     ready to send inverted byte list back to video ram
330 CALL writevram
340 PRINT c$; c$; c$; c$
350 GOTO 180
```

As you have read there are 80K bytes of RAM in the Adam. But did you know that only 64K bytes can be used for your programs? The other 16K bytes are used by the Video Display Processor (VDP). The VDP is, like the Adam's Z80 CPU, a processor but one that deals with placing bytes on the screen instead of to various memory locations. The CPU and the VDP can not use each others' memory directly, so they pass information to each other through a special port (ie. like a door). Figure 1A shows how the Adam's memory is configured in BASIC.

Inside the VDP's memory (VRAM) are a bunch of tables. By tables I mean a list of bytes which pertain to a common graphics feature (ie. like color). The VDP knows where these tables start in video memory because these locations are determined by the VDP's hardware itself. Copies of these start locations are also kept in the 64K section of the RAM. The location of these tables will depend on the graphics mode of the language that you are using. In SmartBasic you can learn where the starting addresses of these tables in VRAM can be found by PEEKing at the following locations.

Table name	Peek location
Sprite Attribute	64868
Sprite Pattern	64870
Screen (Name)	64872
Character Pattern	64874
Character Color	64876

Using the above table and the following formula any VRAM address table may be found. First decide what table you want to find. Then insert the address from the above table into this formula: 256* peek(address+1)+ peek(address). For example, the character pattern table's address in VRAM can be found by using the equation 256*peek(64875)+peek(64874). Again note that these will be video addresses that you are computing.

Screen Table: In SmartBASIC's text mode each byte in the screen table corresponds to the ASCII value of a character on the screen. The first byte corresponds to the character in the home position, the second byte to the second character on the first line, and so on.

Character Color Table: The color table has a byte for each ASCII code (actually for a set of codes) which contain that character's foreground and background colors.

Character Pattern Table: Each character must have a shape to define it. These shapes are stored in the character pattern table. Each character is 8 dots high by 8 dots across, requiring 8 bytes to define a character. The pattern is stored by making the 1 bits of a byte correspond to the on or foreground dots, while the zeros correspond to the background dots. For example,

the letter 'A' has the following bit pattern :

```
00100000  =  32
01010000  =  80
10001000  =  136
10001000  =  136
11111000  =  240
10001000  =  136
10001000  =  136
00000000  =  0
```

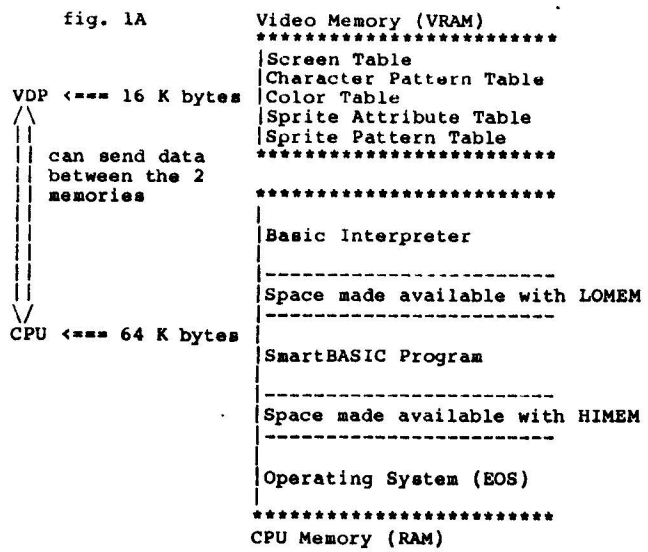
Connect all the ones together like a connect the dots game and you'll see how Adam defines characters. The numbers to the right of the equal signs are the decimal equivalents to the binary numbers on the left. From top to bottom these would be the numbers you would see if you peeked into A's part of the pattern table. To find where this might be, we use the following formula:

pattern address = Pattern table's address + 8 * ASCII code

The ASCII code for 'A' is 65 and the Pattern table starts at zero. Therefore the video address for 'A' is 0 + 8 * 65 = 520.

A program is included on page 9 which will allow you to play around with the Pattern table. It will invert the character you type in. Control-C will break the program. Type TEXT to make the characters normal again. We invert characters by reading their 8 pattern bytes into an 8 byte buffer that is located in (CPU) RAM, switch the bytes around in the buffer, and write them back to the Pattern table. The program uses 2 machine code routines, one to read 8 bytes from VRAM to the buffer in RAM and another to write 8 bytes from the buffer to VRAM. The machine code routines and the 8 byte buffer are stored above the BASIC program, see fig. 1A. This area is made available by using the LOMEM statement in the first line of the program. The machine routines are read from data statements and poked into memory. We use the CALL statement to run the routines.

I'll have to wait until next time to explain the 2 sprite tables, but I do suggest you try to modify the inverter program to allow for your own customized characters.



AdamLink Update

As mentioned in our review in the last issue, the Adamlink program that is included with the modem is incomplete, in the sense that, the user is unable to perform the two most essential telecommunication functions: receiving (downloading) and transmitting (uploading) stored information.

Users who already own the Adamlink modem and all potential buyers need no longer despair as the promised revised program has been released. To obtain Adamlink II, you need to write to the parent company at the following address: Coleco Advanced Communications Program, P.O. Box 10649, West Hartford, Ct. 06110. You will need to provide proof of purchase as well as $9.00 U.S. We suggest you contact the Montreal office for more details on availability and price in Canada. What you get is a completely rewritten owner's manual in addition to the new program.

Those of you who have had the opportunity to use Adamlink II have no doubt discovered a renewed interest in using your modem. At last the user is able to derive the full benefits from his unit. The first thing you will notice after loading the program besides the new welcome title on the screen is the new function for Smart key VI. FILE now replaces the previously useless SHOW PARMS. The old function was repetitious because Smart key IV, SELECT OPTIONS also displays the parameters. Speaking of parameters, two new ones have been added (Auto Redial & Character Filter). Should you fail to connect when you place a call, with Auto Redial set "On", Adam will automatically redial your number every 45 seconds up to a maximum of ten times. The character filter will eliminate the control characters the other computer is sending out which Adam doesn't understand when you download files.

The FILE option is where you are able to select whether you want to receive or transmit files directly to or from tape or disk. Once you are connected with another computer, to receive a file you hit the WILD CARD key to switch to the commamd mode. Next you select the RECEIVE option and the drive you will be using. A file is opened by typing a file name. A message tells you when you can begin to pull in the file. Hitting WILD CARD again returns you to the terminal mode. On the status line will be indicated the number of pages you can store on your tape or disk. Control s and Control q respectively are used to stop and re-start transmission of data in most cases. All the information displayed on the screen is stored. Adam stores slightly more than a screen (equivalent to one half page) at a time. While it is storing the display on the screen stops. You can monitor the number of pages being stored by watching the page counter on the status line. Don't try to store more than 14 pages in one file as that seems to be the maximum Adam can handle. If the file you are receiving exceeds the available storing capacity of your tape or disk, reception is stopped and your file is closed automatically. When you have received the complete file you requested, you must close the file by pressing the UNDO key. It is imperative that you close your file otherwise, you won't be able to access it later nor will you be able to store anything else on that tape or disk.

Transmitting a stored file is accomplished very much the same way except this time you select the TRNSMT option. Switching between terminal and command mode once again is done via the WILD CARD key. After selecting the drive in which your tape or disk is in, you type the file name of the entry you wish to transmit. The ability to get the directory and use the arrow keys to select your file (like you would using SmartWriter) would have been preferable. When the file is located, a message will advise you to press the STORE/GET key while in the terminal mode to activate data transmission. The UNDO key is used to end transmission completely at any time.

The status line displays various error messages if anything goes wrong during file reception or transmission. To avoid unnecessary complications when downloading be sure to use a tape or disk which contains less than four files or preferably none at all. And don't forget to close the file once received. Remember that using the SmartBasic CLOSE command to fix the file won't always work. A warning is included in the manual pointing out that you can't safely edit and save a text file captured by AdamLink. What this means is that modifying a captured file using SmartWriter and then attempting to save the edited version will not always work. You can however print a hard copy of it. This seems to be a rather major inconvenience since the need to edit the files you download will be frequent. When uploading files the most common problems that could arise are: Adam not being able to find a file by the name you indicated or it cannot access and/or read the requested file. Make sure you enter the name of the file exactly as it should be (i.e. with the correct upper and lower case letters) and that the file is one which can be easily accessed.

One thing which puzzles me is why this version of Adamlink wasn't included with the modem in the first place. I don't see it as another marketing tactic to exploit the consumer as it is being offered to the Adam user at a reasonable price. I can only speculate that both internal and market pressures forced the release of a compromise program that wasn't fully functional in all aspects at the time. Whatever the reasons, Adamlink II is a reality and a full-featured modem program at last.

Two final AdamLink notes:
1) AdamLink II does not send CP/M files.
2) If you're still using the original software look for this strange occurrence. After loading the program instead of the normal screen you sometimes get the following.

```
-------------------------------------------------------------
LABEL KOKO PROCESS EVE                          l   V   l  VI  l
                                                l TOUCH l      l
THE CORNER                                      l  HERE l FACE l
-------------------------------------------------------------
```

This seems to occur randomly and is very rare. We have not seen it with the new software yet. Nevertheless, it is something to look for. Now if only someone can tell us what it means.

Basic Bonanza Review Part II

Last issue, we reviewed 7 of Martin Consulting's 15 Basic Bonanza programs. The following short summaries represent the remainder of this valuable software package.

i) TRY ME

Two learning games are included here for people looking for good educational programs for their children. The basic skills of arithmetic and spelling are stressed. The first is a combination guessing and spelling game where clues are given and the child must guess what animal the computer is "thinking of". In addition to the right guess, correct spelling is required but if the child is not successful in his or her attempt the proper spelling is displayed. The second game is a mathematical test where the player answers math questions in order to fill a racer's fuel tank.

ii) FINANCE

Finance consists of three programs: monthly budgeting, savings and loans interest projections, and metric conversions. Popular home finance applications that let Adam do the number-crunching for you.

iii) FILER & iv) LABELS

Filer is a general purpose database program that you can use to maintain files or records of any type electronically. You can use this program to store names on a mailing list and then use the Labels program to print out address labels. The usual filing options (add, delete, edit, search, print) are incorporated and you define the fields to meet your specific needs. Labels allows you to print out your selected entries in different formats in addition to the regular mailing label format.

v) DESIGN

Design gives you an insight into how computer games are created by allowing you to design figures in high resolution graphics through the use of a grid. The program allows you to save your pictures and utilize them in other basic programs of your own.

vi) MAGIC

Magic consists of two programs or illusions designed to impress your friends with the remarkable powers of your computer. MENTAL creates a situation where Adam can actually respond intelligently to questions. It's the old hand is quicker than the eye routine using Adam, the keyboard and code letters built into the program. ANALYSIS lets your Adam play the role of a wise analyst. After asking for some personal information, Adam will then provide the player with a personal analysis.

vii) ASSEMBLER & viii) DISASSEM

These two programs are directed at the more advanced computer users who are familiar with assembly language. Assembler is a menu driven program that translates assembly language to machine code while Disassembler translates machine language into assembly language. With it you can study the machine language programs residing in memory. One interesting use for Disassembler is to disassemble the Basic interpreter. This is the program which permits you to communicate with Adam in SmartBasic.

Electronic Flashcard Maker

Everyone is familiar with flashcards. We've all used them at one point or another to help us study important facts for school tests or for lectures we had to give or simply to play trivia or other quiz games. If you thought that flashcards were a thing of the past in today's computer-oriented society, look again! The programmers at Coleco thought otherwise and have ressurrected flashcards into electronic form.

With Electronic Flashcard Maker you can create, edit and use as many as 30 decks each holding up to 200 cards. Anyone can create personalized study aids that make learning facts more fun and rewarding. You can use flashcards to memorize speeches or lines for an acting role. And they're ideal for creating educational and entertaining family games.

Loading the program from data pack takes almost a minute and a half. In the meantime, we are treated to a musical interlude. At the entry screen you select whether you will make, edit, or use the cards using the Smart keys or the corresponding button on the controller keypad. Selecting "Make Cards" will cause the screen to display both sides of a card. You enter your question or notes on one side and then enter the corresponding answer or related information on the other side. Adam numbers the cards automatically. If you need to go back to a card to make changes, you can run through the deck either in decreasing or increasing order. To facilitate reordering cards already in your deck, two decks are at your disposal. Deck 1 is the main deck while deck 2 is a temporary holding place for the cards you want to reorder. Extensive provisions are provided for sorting. These will be most useful when editing and sorting decks containing many cards.

When you're ready to use your flashcards, you can go through them manually or have them flash automatically on the screen. Either way, you can have them appear randomly or in sequential order. In addition, you can have all the cards that you mark as problem cards set aside and reshown later in a special deck of misses. If you don't choose this option , any cards you mark as misses will be recycled into the main deck. You can answer orally, mentally or by typing the answer, in which case it will appear on the screen in the answer area below the card. The program does not determine whether you answered correctly or not, you must mark the cards that you missed. This is done by pressing the Smart key so labelled. In the auto mode you can specify the length of time the card will remain on the screen, a handy feature to have when you want to put yourself through a speed drill on material you're comfortable with. When you've answered the last card Adam reports how well or how bad you did. It displays the number of cards you marked as misses and the percentage of those answered correctly. To add a degree of flexibility when using your flashcards for game playing or for studying, you will want to "flip the deck". This will change the side of the card which is displayed first. If you were previously shown the questions first, you will now see the answers and vice versa.

The program has a pause control built into it. You will really

appreciate this if you get caught up in a marathon session and want to take a break once in awhile. Activating the pause causes a music melody to be played, much like the pause button found on most games. In fact, you can use the hand controller to operate Flashcard Maker and there is even a special keypad overlay included for such a purpose.

One of the most interesting features found in this program is its ability to generate accent marks for foreign languages. A slew of other special characters in addition to exponents, subscripts, musical notes and math symbols are possible. This does not limit the program to certain users and makes it as useful to the math or science student as to the languages student. A major drawback, however, is the fact that none of these special characters or accent marks can be printed out. A special daisy would have been necessary to accommodate this. The Wild Card key is used to bring up the accent marks & the Control key the many other characters.

Flashcard Maker is not only useful for creating flashcards electronically, but it's great for other home applications as well. It can be easily used to catalog collectibles such as coins, stamps, books and records. It's also a filing system for recipes, addresses and telephone numbers. When cataloging you can enter the major characteristics of the item on the A side and the less important details on the B side. Remember each deck or mini database can hold up to 200 cards or records. Setting up a filing system using flashcards will work similarly. You will only be able to file numerically by card number which is not too much of a problem since the program's reordering features allow you to sort and edit quite easily.

If you get bored with making your own flashcards, you can buy one from Coleco's Flash Facts Series. One of these sold in Canada is Flash Facts Vocabulator. Here you can choose from 25 different decks covering a broad range of vocabulary-building subjects including definitions, expressions, homonyms, synonyms, antonyms, and much more.

Game Reviews

Dambusters Update: Last issue we gave you a lot of "how-to" information. This helped many of you to get a handle on the game. Since then we have discovered things that might make the game even more compehensible. Many of you have expressed consternation with the altimeter. Well we can confirm it now. The altimeter is just plain wrong. The large hand moves in 5 foot increments not the 10, as stated in the manual. Fig. 2 is wrong, 1000 ft is not at what on a clock would represent the 3 o'clock position but between 1 and 2 o'clock. It is impossible to get to where the hands are pointed in fig. 2. Before you get there the plane spins out. These altimeter readings are crucial for the bomb drop so note these differences well. While we're on the subject... try this. Press keypad #5 (the navigator screen). Now move your navigational cursor into another quadrant. Now press the side fire button. Nice the way the map flips back to where the plane is. The only problem is that when your plane is out of the first quadrant and you try this you still get the first quadrant.

Title: 2010: Strategy

Cartridges made to cash in on the success of popular fantasy science-fiction movies seldom make very good games. Coleco's "Star Trek" was, except for the theme music, a disappointment. Parker Brothers' "Star Wars" was similarly devoid of innovation and excitement. Manufacturers seem to believe that a box office success can carry a bad cartridge. And sometimes it can.

"2010: Strategy" is, of course, based on the book (by Arthur C. Clarke) and movie of the same name. "2010" continues the adventure started in "2001: A Space Odyssey". In the movie (and game) a reconnaisance ship, the "Leonov" is sent out to recover the ailing "Discovery". The "Discovery's" crew has disappeared and the mystery lies within the ship's onboard computer "HAL". Time is of the essence. The ship must be repaired and restarted before it crashes into the surface of the Jovian moon Io.

Little of the mystery and wonder of Clarke gets translated into the cartridge. Above I gave a bare bones sketch of the plot. If you've seen the movie or read the book you will remember the tense cat and mouse game with HAL or the metaphysical slant of the movie. The cartridge cuts out all of this. You must repair the ship before it crashes. That's like saying that Vonnegut's "Slaughterhouse Five" was about some guy who knew how to time travel.

"2010" could have been vastly improved by adding some randomness into the game. Then decisions based on particular situations could be made. That would have been strategy. As it is, "2010" is more of a mediocre action game than any sort of strategy game.

This is how you play "2010". Life support, communications, reactor, and engine circuits are all burnt out. Individual section and total system monitors are all provided. An altimeter shows the constant pull of Io's icy grip. Skill level one starts you off at the highest altitude above the surface with the HAL system operational. Using the paddle you select the circuit that you want to repair. Pressing the side button switches you to a full screen circuit repair diagram. You must now move a "spark" through all the CTI's (cryotronic interphases) in order to repair the circuit. The action is very similar to the game "Oilswell". Caution must be used in order to avoid the MFF's (floating magnetic flux fields). If your spark should be in a CTI when a flittering MFF crosses your path - KABOOM! - your CTI blows. You must then repair it with your minature repair drone "Waldo" and start activating your circuit once again. Crossing a CTI more than once can also blow it for you.
On higher skill levels you must repair HAL circuits. These use CSI's (cryosynaptic interphases) and are a little more difficult to fix. Directional switches are used to get the spark through the circuit. However, in this case you must avoid the CTI's. Passing through a CTI or getting hit by a MFF when in a HAL circuit means another call on Waldo.

Once all systems are up, you can blast out of orbit. Contrary to the description in the manual the take off is not spectacular.

The ship simply moves off the right hand of the screen and leaves. That's it, end of game.

So to sum it up, for an action game "2010" is not too innovative but it's not too bad. But to call it a strategy game is simply misleading. If you get it, buy it for its action. Or even better get it for its pause feature. When you press pause you hear this surrealistically absurd circus version of HAL's favorite song "Daisy". Someone has a great sense of (black) humor.

P.S. We've just received word that Coleco will be releasing a text version of "2010". In all probability this will be a real strategy game. More next issue.

Members' Programs

Here are some of our members' programs for this month. They are printed here for the following reasons: They illustrate a particular programming technique, (pokes, machine language routines, etc.) They are useful or entertaining programs. They are not too long.
We are still building up a library of non-copyrighted programs. Programs of note that are simply too long for the newsletter will be accumulated therein. Details of how this library will be made available to members will be published in the next newsletter. Until then we encourage members to continue sending in programs. Some of you have sent your programs in on tape or disk along with a hardcopy listing. We appreciate this immensely. Since we run each program before we publish it (to make sure it works) this speeds up our reviewing process considerably. If you do send a tape or disk in, don't worry, they're always sent back. The safest way to send in material to us is by registered or insured mail. That way you'll be sure that we receive it. Should it get lost when we send it back, we'll even replace it.

P.S. B. Crepeau asks us to add this line to his progam found in issue 1.1 page 28: 205 if f=1 then goto 220

```
 10 &     RENUMBER LINES ONLY
 11 &     D.Lelievre
 12 &
 13 &  This routine renumbers lines in programs it is called from:
 14 &  You have to manually change the line # references in your
 15 &  program's GOTO's, GOSUB's, etc. The starting line # and the
 16 &  increment between lines are in low/high byte format in
 17 &  locations 1101/1103 & 1105/1107 respectively. The default
 18 &  values give line #'s 10,20,30,40, etc.
 19 &  To use this routine, run this program, then type in
 20 &  BSAVE RENUMBER,A1090,L40 Then load any program you want
 21 &  renumbered, put in line # 65535 REM . Then type in
 22 &  BLOAD RENUMBER and CALL 1090 . Watch out, it's super fast!
 23 FOR i = 1090 TO 1130
 24 READ by%: POKE i, by%
 25 sum = sum+by%: NEXT
 26 IF sum <> 3624 THEN  PRINT "ERROR IN DATA STATEMENTS": END
 27 DATA   221,42,217,62,62,255,22,0,30,4,46, 10,38,0,14,10,6,0
 28 DATA   221,190,1,32,6,221,190,0,32,1,201,221,117,0,221,116,1
 29 DATA   237,74,221,25,24,233
65535 REM
```

```
 10 &  VERTICAL MESSAGE SCROLL
 20 &  D.Lelievre
 30 &
 40 &  This program has two machine language routines that
 50 &  read character patterns from VRAM and set them up
 60 &  as 0 or 1 print flags to generate 8 row x 8 col. characters
 70 &  on the screen (and printer).
 80 LOMEM :28300
 90 DIM me$(255)
100 FOR i = 28000 TO 28078
110 READ by%: POKE i, by%: sum = sum+by%
120 NEXT i
130 IF sum <> 6440 THEN  PRINT "Error in data statements": STOP
140 DATA   46,0,38,0,6, 8, 221, 33, 197,109
142 DATA   125,211,191,124,211,191,0,0,0,0
144 DATA   219,190,221,119,0, 35,221,35,16,236
146 DATA   201,0,0,0,0,0,0,0,0,0
148 DATA   46,197,38,109,221,33,41,110,6,8
150 DATA   197,6,8,126,203,39,56,6,221,54
152 DATA   0,0,24,4,221,54,0,1,221,35
154 DATA   16,238,193,16,1,201,35,24,227
160 TEXT: PRINT: PRINT: PRINT "Type in message:": PRINT: PRINT
165 INPUT ""; me$
170 ln% = LEN(me$)
175 IF ln% = 0 THEN  STOP
180 FOR i = 1 TO ln%
185 me$(i) = MID$(me$, i, 1)
190 NEXT i
194 INPUT "PRINT-OUT(1=YES) ?"; p$
196 IF p$ = "1" THEN  PR #1
200 FOR i = 1 TO ln%
210 IF me$(i) = " " THEN 310
220 GOSUB 500: &   Put 8 bytes from vram to 28101-8
230 CALL 28040: &   Set 64 print flags at 28201-64
240 b$ = me$(i)
250 FOR j = 28201 TO 28264 STEP 8
260 FOR k = j TO j+7
270 IF PEEK(k) THEN  PRINT b$; : GOTO 290
280 PRINT " ";
290 NEXT k
300 PRINT: NEXT j
310 PRINT: PRINT: NEXT i
320 PR #0: GOTO 160: &   Restart
499 &   Put 8 bytes from vram to 28101-8
500 ad% = ASC(me$(i))*8
510 h% = ad%/256: l% = ad%-h%*256
520 POKE 28001, l%: POKE 28003, h%
530 CALL 28000
540 RETURN
```



```
 1 &  FACTORS
 5 &  S. Harper
 7 &
10 &  This program
11 &  factorizes
12 &  a given number
13 &
15 PRINT
20 PRINT " Enter Number"
25 PRINT " to be Factored"
30 PRINT " Hit '0'to end"
35 PRINT
40 INPUT x
45 PRINT
50 IF x = 0 THEN 999
60 FOR y = 2 TO INT(SQR(x))
70 a = x/y
80 b = INT(a)
90 IF b = a THEN 300
100 NEXT y
110 PRINT " "; x;
115 PRINT " is a prime no."
130 GOTO 15
150 FOR y = 2 TO INT(SQR(x))
160 a = x/y
170 b = INT(a)
180 IF b = a THEN 350
190 NEXT y
200 PRINT " and "; x
210 GOTO 15
300 PRINT " The factors"
305 PRINT " of "; x; " are"
325 PRINT
350 PRINT " "; y; ",";
360 x = x/y
370 GOTO 150
999 END
```

```
  5 REM   Graph Part 1
 10 DEF FN o(x) = TAN(x)
 20 DEF FN p(x) = 1
 30 DEF FN q(x) = FN o(x)/FN p(x)
 50 INPUT "a partir de quel endroit?"; a
 55 PRINT " "
 60 INPUT "jusqu'a quel endroit? "; h
 65 i = (h-a)/256
 70 DIM z(256)
 75 DIM asymp(256)
 80 init = a
 85 IF FN p(a) < 1E-03 THEN  a = a+i: GOTO 85
 90 min = FN q(a)
100 max = FN q(a)
103 REM      CALCUL DES POINTS
105 a = init
110 FOR w = 0 TO 255
115 IF ABS(FN p(a)) < 1E-07 THEN  asymp(w) = 1: GOTO 130
118 b = FN q(a)
120 z(w) = b
125 r = r+b*b
130 a = a+i
135 NEXT w
138 REM      ECART TYPE
140 s = SQR((r-r/256)/255)
142 REM      DETECTION DES ASYMPTOTE
143 a = init
145 FOR w = 0 TO 255
148 IF asymp(w) = 1 THEN 180
150 b = FN q(a)
155 IF ABS(b) > 2*s THEN  asymp(w) = 1: GOTO 170
160 IF b > max THEN  max = b
165 IF b < min THEN  min = b
170 a = a+i
180 NEXT w
185 REM         **********************
190 REM         "plottage" des points
195 REM         **********************
200 HGR2
203 IF init < 0 AND init+255*i > 0 THEN  GOSUB 1500
205 HCOLOR  = 2
210 a = init
220 FOR w = 0 TO 255
223 IF asymp(w) = 1 THEN  GOTO 2000
225 c = INT(191*(min-FN q(a))/(max-min))+189
227 IF c > 191 THEN  c = 191
228 IF c < 0 THEN  c = 0
230 HPLOT w, c
240 a = a+i
250 NEXT w
300 IF min < 0 AND max > 0 THEN  GOSUB 1000
400 GET c$
405 REM   Graph Part 2
410 IF c$ = "" THEN 400
420 IF c$ = "r" THEN  TEXT: RUN
430 IF c$ = "l" THEN  TEXT: LIST -20: END
440 IF c$ = "p" THEN  TEXT: GOTO 500
450 TEXT: END
500 PRINT " ": PRINT " "
510 INVERSE
520 PRINT "                              "
530 PRINT "      P A R A M E T R E S     "
540 PRINT "                              "
545 NORMAL
550 PRINT " ": PRINT " "
560 PRINT "LE MIN EST DE  :"; min
565 PRINT " "
570 PRINT "LE MAX EST DE  :"; max
575 PRINT " ": PRINT " "
580 PRINT "LA FCT VA DE   :"; init
585 PRINT " "
590 PRINT "JUSQU'A        :"; h
595 PRINT " ": PRINT " "
600 PRINT "L'INCREMENT EST:"; i
610 END
970 REM         **********************
980 REM        dessin de l'axe des x
990 REM         **********************
1000 d = INT(191*(min-0)/(max-min))+189
1005 IF c > 191 THEN  c = 191
1008 IF c < 0 THEN  c = 0
1010 HCOLOR  = 3
1020 HPLOT 0, d TO 255, d
1040 RETURN
1470 REM         **********************
1480 REM        dessin de l'axe des y
1490 REM         **********************
1500 r = ABS(init/i)
1510 HCOLOR  = 3
1520 HPLOT r, 0 TO r, 191
1540 RETURN
2000 REM       ***************************
2010 REM         dessin d'un assymptote
2020 REM       ***************************
2030 HCOLOR  = 8
2040 FOR k = 0 TO 191 STEP 20
2050 FOR l = 0 TO 10
2055 HCOLOR  = 2
2060 HPLOT w, k+l
2070 NEXT l
2080 NEXT k
2090 HCOLOR  = 2                Jocelyn Doire
2100 GOTO 240                   Quebec
```