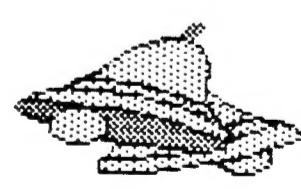
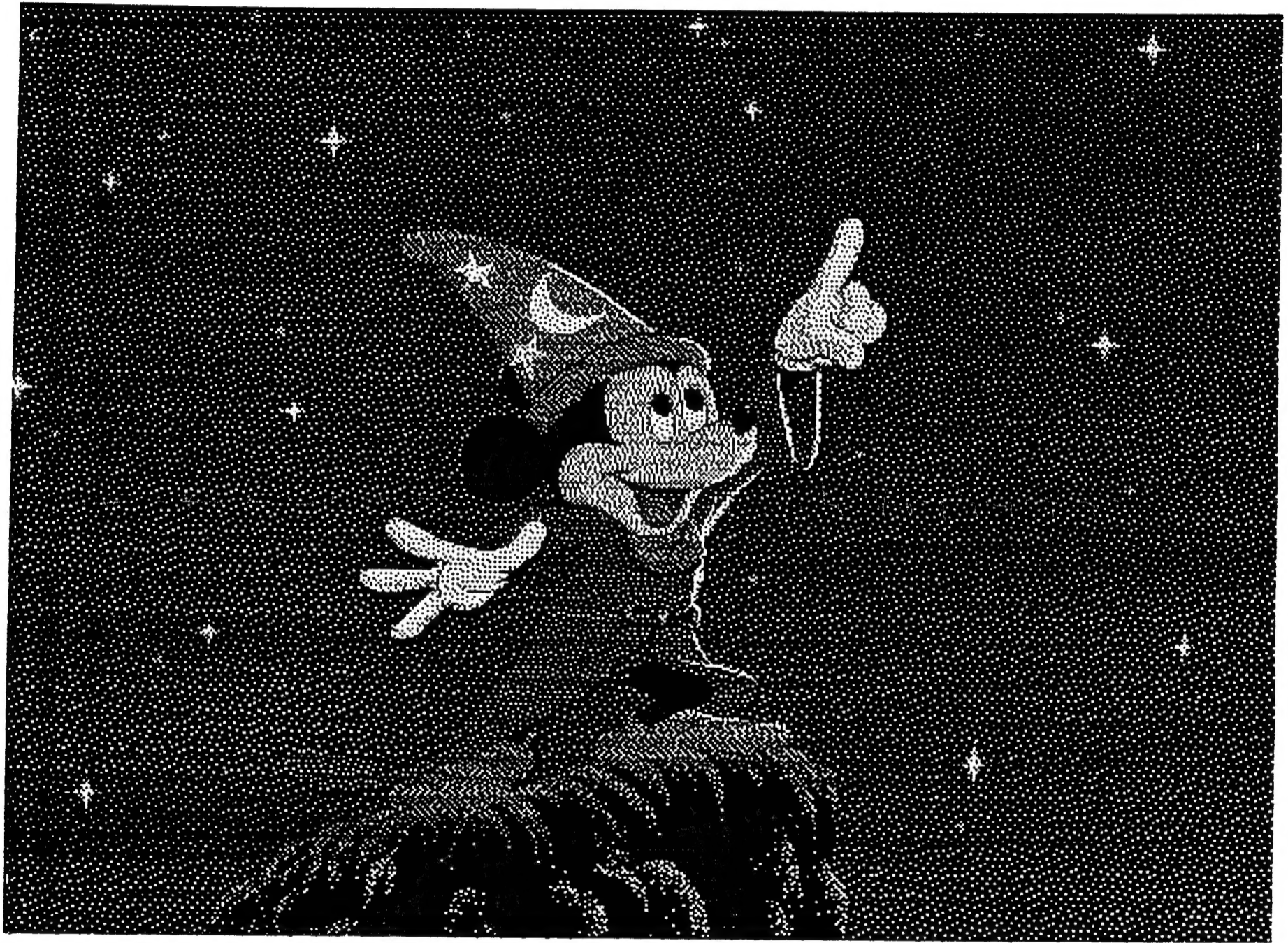


SINC - LINK



NOV - DEC '93 VOL 11-6



**PEACE
ON
EARTH**

TORONTO TIMEX - SINCLAIR USERS CLUB



SINC - LINK



NOV - DEC '93 VOL 11-6

SINC-LINK IS A PUBLICATION OF THE TORONTO TIMEX-SINCLAIR USERS CLUB AND IS ISSUED 6 TIMES A YEAR. CLUB MEMBERS RECEIVE FREE COPIES AS PART OF THE \$20.00 ANNUAL MEMBERSHIP FEE.

NEWSLETTERS ARE EXCHANGED, FREE OF CHARGE, WITH OTHER TIMEX-SINCLAIR USER GROUPS.

PLEASE CREDIT THIS PUBLICATION AND THE AUTHOR IF YOU COPY MATERIAL.

THE TS2068 & ZX-81 GROUP MEETS ON THE FIRST WEDNESDAY OF EACH MONTH AT 14 RICHOME COURT, SCARBOROUGH, ONT. 7PM START.

THE QL SIG WILL MEET AT 586 ONEIDA DRIVE, BURLINGTON, ONT. 7PM START. NEXT MEETING TO BE ANNOUNCED.

SINC-LINK IS PRODUCED ENTIRELY ON SINCLAIR AND TIMEX-SINCLAIR COMPUTERS.

SEND CORRESPONDANCE TO:

SINC-LINK EDITOR, TORONTO TIMEX-SINCLAIR USERS CLUB, 14 RICHOME COURT, SCARBOROUGH, ONTARIO, CANADA M1K 2Y1.

EXECUTIVE OFFICERS:

PRESIDENT:	RENE BRUNEAU (531-9749)
TREASURER:	BILL LAWSON (444-8772)
SECRETARY:	GEORGE CHAMBERS (751-7559)
ACTIVITIES:	LOU LAFERRIERE (820-3725)
QL CONTACT:	HUGH HOWIE (634-4929) NOTE NEW AREA CODE 905
NEWSLETTER:	JEFF TAYLOR (244-8583)
LIAISON OFFICER:	GEORGE CHAMBERS, 14 RICHOME COURT,
(Out-of-town members)	SCARBOROUGH, ONTARIO M1K 2Y1
	(416-751-7559)



TORONTO TIMEX-SINCLAIR USERS CLUB

TORONTO TIMEX - SINCLAIR USERS CLUB

EDITORIAL

News of our demise has been somewhat premature, to twist a famous quote. The club is not folding despite what some readers may have heard or inferred. The format of the newsletter has changed, of course, as ZX81 and TS2068 interest wanes, but we will be back in 1994 for our twelfth year of publication. Depending on support we may, or may not, put out six issues as customary. It all depends on you, the readers (and writers, I hope!). Let's not give up on our marvellous little wonders.

On behalf of the Club Executive, I'd like to wish you all the best in the coming year. Happy Computing!

J.T.

INDEX VOLUME 11-6

- page 2 - Club Stuff
- page 3 - Index, Editorial
- page 4 - 24 Pin Bit-Image Graphics
- page 7 - Qlips
- page 8 - Did You Know?
- page 9 - Shopping in the U.S.A.
- page 10 - Softly As I Leave You - Music
- page 12 - John Juergens Writes
- page 14 - XCHANGE Info
- page 15 - Did You Know? cont.
- page 22 - Howard Clase Plays With Disks
- page 30 - Season's Greetings



Merry Christmas



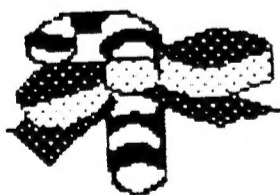
FOR SALE FOR SALE FOR SALE

One of our club members is moving from the Timex world and has a quantity of TS2068 material to sell. It is too lengthy to list here but you can send him a SASE, and ask for a list.

His stuff includes a TS2068 w/Spectrum ROM, Aerco printer interface, Larken disk system, Larken RAMdisk, TS2050 modem. Also some TS2068 books, programs, games, etc.

I don't have any prices; I suggest you make him an offer, either for items which interest you or for the whole.

Address: Keith Worrall, 22...2nd Avenue North, St. Mathieu, Quebec, CANADA JOL 2H0



JOY



2068 24-PIN BIT IMAGE GRAPHICS
FOR 24-PIN OR BUBBLE JET PRINTERS IN EPSON EMULATION

Larry Crawford / 357 Reynolds Rd / London Ontario Canada N6K 2P8
(519) 657-9119 PUBLIC DOMAIN 05 Nov 93

If you went for a 24-pin or bubble jet printer and had thoughts of trying to print some graphics in the 24-pin bit image mode, you were probably somewhat taken aback by the apparent complexity of it all, just as I was.

This article should take some of the mystery out of the process. The demo program will give you the information needed to develop your own applications.

BACKGROUND

The 24 print head pins are arranged into three groups of eight. In effect, each group acts as a separate 8-pin head covering 1/3 of a print line. Thus, with one pass it is possible to achieve the vertical definition of three 8-pin passes with only one pass. Furthermore, it is possible to achieve a density of 360 dots per inch in Hex Density mode.

These attributes make the printing very fast and should allow a CAD program to produce printed circuit board layouts with good solid lines without the need for multiple passes. Unfortunately, the existing CAD programs that I have seen are all based on 72 dots per inch (dpi) horizontal density and 8/72" vertical line spacing. The 24-pin printer does not have the 72 dpi option. Instead, it uses multiples of 60 dpi and 1/360" paper feed. This makes the size of the printed image larger than that produced by an 8-pin printer.

Consequently, it cannot be used by software programs such as Pixel Print or CAD in their present form: a) two columns of 64 characters will not fit side by side on a page and b) the socket hole spacing on a pcb layout would be too great. It should be possible to modify the calculations performed by a CAD program to get the scale of the final image correct.

There are numerous other applications, of course, so feel free to make use of the information that follows.

As a demonstration, we will print the first line of the screen in 24-pin double density bit image graphics. The data in the conversion chart that follows is essential to the process.

Type in the following program (without the parenthetic comments, of course):

```
5 REM 24-pin bit-image demo
10 CLS: PRINT "QWERTYUIOPASDFGHJKLZXCVBNM123456"
   [this will put a single line on the screen to copy]
20 IF IN 127<>236 THEN INPUT;: PRINT #0;"PUT PRINTER ON LINE":
   PAUSE 2: GO TO 20
   [An important reminder since the screen will go blank and
   nothing will happen if the printer is not ON.
   <INPUT;> is a simple way to clear out the bottom of the
   screen]
30 RESTORE 30: GO SUB 500: DATA 27,65,8,999 [set Line Feed to
   8/60"]
40 LET y=175: RESTORE 40: GO SUB 500: FOR x=0 to 255: DATA
   27,42,33,0,2,999 [y points to top of screen; codes are sent
```

```

to set printer for 24-pin bit-image double density: it will
expect 2x256 bytes of data: the x loop will point to all 256
pixel columns across the screen]
50 LET b1=224*POINT (x,y)+28*POINT (x,y-1)+3*POINT (x,y-2)
["b1" is the "top" byte. Following the chart, if the top 3
pixels of the screen are INK then the value assigned to "b1"
will be 224+28+3=255. Therefore, all top 8 pins of the print
head will fire]
60 LET b2=128*POINT (x,y-2)+112*POINT (x,y-3)+14*POINT (x,y-4)
+POINT (x,y-5)
["b2" is the "middle" byte. Following the chart, if the
pixels 3rd, 4th, 5th, & 6th from the top of the screen are
all INK then "b2" is assigned a value of 128+112+14+1=255.
Therefore, the middle 8 pins will all fire]
70 LET b3=192*POINT (x,y-5)+56*POINT (x,y-6)+7*POINT (x,y-7)
["b3" is the bottom byte. Following the chart, if the pixels
6th, 7th & 8th from the top of the screen are INK then "b3"
will be assigned a value of 192+56+7=255. Therefore, the
bottom 8 pins will all fire]
80 RESTORE 80: GO SUB 500: NEXT x: DATA b1,b2,b3,b1,b2,b3,999
[send the 3 bytes to the printer twice for double density]
90 INPUT;: PRINT #0;x: NEXT x [clears the bottom of the screen
then prints the pixel column # to let you know that the
computer is calculating the data bytes. Gets the next col #]
100 RESTORE 100: GO SUB 500: STOP: DATA 13,10,999
[send CARRIAGE RETURN and Line Feed]
500 READ a: IF a=999 THEN RETURN
[999 is a dummy value to signal the end of current data]
510 IF IN 127<>236 THEN GO TO 510 [if the printer is busy, wait
until it is ready for data. (<INPUT;> is a simple way of
clearing the bottom 2 lines of the screen)]
520 OUT 127,a: GO TO 500 [send data to the printer]
9999 RANDOMIZE USR 100: SAVE "11dem.B1"

```

Now <GO TO 9999> to save it to disk and then <RUN>

It takes nearly a minute to get to the actual printing because of all the calculations to be done in BASIC.

If you want to print a whole screen, make the following changes:

```

LINE 10: Replace with: 10 LET ctr=0: RANDOMIZE USR 100: LOAD
"screen name" SCREEN$
["ctr" will keep track of the print line being
processed]
LINE 40: Replace <LET y=175> with <FOR y=175 TO 7 STEP -8: LET
ctr=ctr+1:>
[This will set up a loop to deal with all 22 screen
lines and increment the line counter]
LINE 90: Add <ctr;"", "> immediately after <PRINT #0;>
LINE 100: Add <NEXT y:> immediately ahead of <STOP:>
LINE 9999: Change program name

```

LINES 40 and 100 set up a loop to look at all 22 lines of the upper screen, outputting the data to the printer at the end of each line. A full screen takes over 20 minutes to copy.

To print out in triple density, change the data in line 30 to 27,42,39,0,3 and add another set of b1,b2,b3 to the data in line

80. The printer will then expect 3x256 bit image data bytes.

 A much faster version gets the data from the screen file instead of the screen. It puts the data into 1 or 22 line files, ready to be loaded from disk then sent to the printer by a 205-byte m/c routine. It takes about 5 minutes to create and save the 22 files and less than one minute to print them all in triple density.

George Chambers has a copy of this program with documentation on a LARKEN club disk. If you would like an Oliger version, send me \$5.00 cash or money order. (Larry C.)

 A full screen produces an image 4.25" by 2.9" (10.8 by 7.4 mm) and is proportioned so that squares are square and circles are round. Different screen images can be printed consecutively on the same sheet so that a composite image can be 4.25" by any length.

 8-PIN TO 24-PIN CONVERSION CHART
 FOR A 24-PIN PRINTER WITH EPSON EMULATION

		8-PIN		24-PIN		
		PIN#	PIN#	CODE		
TOP (BIT 7)>			1			
		1	2	224		
			3			
			4			"TOP BYTE"
		2	5	28		
			6			
			7			
		3	8	3		
		1	128			
		2				
	4	3	112			
		4			"MIDDLE" BYTE	
		5				
	5	6	14			
		7				
		8	1			
	6	1				
		2	192			
		3				
	7	4	56		"BOTTOM" BYTE	
		5				
		6				
	8	7	7			
		8				

Q L I P S

by Hugh Howie

For some time now I have been experiencing a problem with one of my disk drives on my #2 QL, in that I could not always format a disk in #2 drive. I would get a "Failed" message, or the disk would be formatted with less than the correct number of sectors. I was able to use this drive to write to, and to read, with no apparent problem, but format was unreliable.

This particular setup has three drives on the Trump Card:-

#1	5 1/4	720
#2	3 1/2	DD
#3	5 1/4	360

I am able to use three drives on the Trump Card as I obtained from Miracle Systems a small adapter (card) which permits the Trump or the Gold Cards, to handle up to four drives. This configuration means that in conjunction with my #1 QL, I can copy the TorQLib Library to just about any configuration that may be presented to me by a member.

Recently I had occasion to be copying some disks for someone who sent me 5 1/4 disks, and in the process of copying I found one of his disks "Not Found" which usually means it has not been formatted. I reformatted it in #1 a number of times with varying results, none of them correct. I returned it to him, and he replied that he had no problem with that disk, that it formatted, was written to, and was readable with no difficulty.

I then recalled another member who had a setup similar to mine, and he was having trouble of a like nature. I began to ponder if the problem could be in the small Disk Adapter, so I made a few test runs to format a disk in #2 drive, the one most likely to give me trouble.

I started up the system, and, using a new 3 1/2 disk I formatted it four times, and there was no instance where I got other than 1440/1440. I tried it again, using the command:-

```
for i=1 to 4:format flp2_:end for i
```

This time I got one report of less than

normal. Subsequent runs gave varying results. So I tried another two disks with similar results.

I then switched off the unit, removed the Disk Adapter, and after many runs of all disks, I had no errors. Replaced the Adapter, and started getting errors again.

I checked the adapter to see if there were any wires touching where they should not, and pried any doubtful ones away from the board. It should be noted that the board is open and it is possible for undue pressure to be applied to its components

Next day I thought I would retry the experiment, and the results were exactly the same.

Again, the initial results were fine, but after the QL had been running (warming up the Adapter?) for a short period of time, I began to get less than satisfactory results.

Conclusions:-

1. The Disk Adapter does allow up to four disk drives to run from Trump Card or the Gold Card.
2. With the adapter, flp2_ appears to be faulty in formatting.
3. Flp1_ would appear to be OK except for that one disk which I mention.
4. Until the Adapter "warms up", it would appear to be reasonably reliable, but not always.
5. Without the Adapter, the unit formats fine.

If there is anyone who has had similar experiences in using this Adapter, I would be glad to hear from them.

This is my story, This is my song,
If you have any ideas, Just send them
along.

Hugh H. Howie, 586 Oneida Dr.
Burlington, Ont. Canada, L7T 3V3

DID YOU KNOW ? That there are two different controller chips used in the LKdos EPROM I/F board? They are WD1770 and WD1772. The machine code in the EPROM on the cartridge board is slightly different for each one. If you have the WD1770 chip your version 3 EPROM will be marked L3. If you have the WD1772 controller you will find that your EPROM is marked L3f.

The disassembly that has been appearing in Sinc-Link was made from a version 3f EPROM. The disassembly was done by Ken Shoenberger using his TSDB disassembler program. It was 35 pages long and he had analyzed it making handwritten remarks. Before I submitted it to Sinc-Link I retyped it in MScript so I could print it with a smaller font and type in his remarks. I was able to reduce it to 21 pages. As of writing this article only the first eight pages have been published.

To verify that I hadn't left anything out or botched the typing I proofed it against my chip. I have an EPROM reader/burner that will make a disk copy of the EPROM. It works on an IBM clone so I converted it to a LKdos disk using the Larken Disk Editor. When I am ready to proof a page I load the LKdos code at address 40,000, load TSDB and select an offset of 25536. When I disassemble I get the correct addresses on the screen. Since I have the (earlier?) WD1770 chip I have the L3 EPROM. That is where I discovered the difference between the two chips. L3f has two additional RRA instructions at addresses 2336 and 2337. There is a 2 byte difference in all addresses from there up to 4364. So the code between 2336 and 4364 in the disassembly will be different if you have the L3 version. (That is, pages 7 to 12 of the disassembly.)

Reading the code with the EPROM reader also verifies that the "forbidden addresses" (98 to 111) are accurate as they appeared on the first page of the disassembly.

One of the reasons for submitting the entire disassembly was to answer part of the questions raised by Robert Shade in his letter published in Vol. 11, #1, about increasing the size of the RAMdisk. In the RAMdisk assembly instructions Larry Kenny says "The order of the ramchip selection is not linear because of a small PCB error". A lookup table at 4175 to 4222 compensates for this. I haven't studied the possibility of adding more RAM, but it is obvious that two things would have to occur before this could even be considered: 1) correct the circuit board error, and 2) revise the code in the vicinity of 4153 where the lookup table is used. (See page 12 of the disassembly.)

Pages 1 through 13 were done in MScript. Pages 14 through 21 were done on a MacIntosh. The file was transferred to a MSdos disk where I could then read it back into MScript using the Larken Disk Editor. My only problem with this process was the difference in how tabs are handled. The Mac program uses a 9 to indicate a tab and I discovered that MScript "pads" the tabbed areas with spaces. I changed all the 9's to spaces (32) and then had to add more to return to the desired configuration. My original reason for doing this was to save typing time because I could type columns on the Mac. This turned out to be a "pay me now or pay me later" exercise. We live and learn! +LSC+

SHOPPING in the U.S.A.

I read the comments by Louis Laferriere in the last issue of Sinc-Link, and of his experience of making a purchase in the U.S.A., and how the Post Office collected the GST, (General Sales Tax to the uninitiated), then charging him \$5 for the privilege of doing so.

My experience is similar, but much more expensive.

I phoned J. D. Hannam Inc. in Anaheim, California, and asked them to send me an ED disk drive. I forgot to tell them to send it by Mail.

I couple of days after I placed the order I realised my omission, but as I thought the goods had probably been dispatched, I did nothing about it. I had a good idea as to what was going to happen. Sure enough it did.

A few days later I had a phone call from UPS in Windsor, saying they had a package for me, and that the tax would be \$xx, I said OK, and was told that there also would be a brokerage fee of \$20. (Shipping charge at source, \$5.70 US)

To travel to Windsor and clear the package myself, would have cost me more in gasoline than brokerage fee, so I had to go along with this exorbitant larceny.

From the size of the UPS truck/trailer combinations I have seen in my travels, heading north to Canada, I can only conjecture that if every package is subjected to the same brokerage, then UPS could quite possibly transport goods free, and live on the brokerage fees.

I had a similar experience with UPS a couple years ago, but in this case the Port of Entry was Fort Erie, so I went there and cleared the package with no problem, and it only took me five minutes with Customs to do so.

Since the Fort Erie experiences, I have asked my suppliers to send the packages by Mail, and have had no problems. I am notified when there is a package at the Post Office to be collected, I do so, and if all I am charged is \$5, then that is better than \$20. I still agree that the \$5 fee Louis was charged by the Post Office would be onerous to me.

I can only say to anyone making a purchase from the States, that they request the supplier to send it by Mail. If they do otherwise, then refuse to accept delivery.

I would ask all USA suppliers sending anything to Canada, to send it by Mail. The time taken by the Mail Services is not that much different from the Parcel Services - and cheaper.

Hugh Howie.

August 15, 1993

John E. Juergens
18 Bryce Canyon Way
Pacifica CA 94044-3723

Hugh H. Howie
QL Contact
586 Oneida Drive
BURLINGTON ONT.
CANADA L7T 3V3

Dear Hugh,

This is in response to your QLIPS note on EXCHANGE. Would like to talk you out of a copy if I could.

Am using the original, bundled Archive for our checking and tax program and it has worked just fine for us for many, many years. My only complaint with it is that, like all of the programs in the Psion suite, it did not come with a start-up macro.

Long ago I augmented Quill with Turbo +. Did not know at the time of purchase that we were also getting macros with it; particularly, CTRL-S which has saved a lot of time and keystrokes when starting Quill.

I have used Psion's PC-4 group and it, as apparently does the EXCHANGE group, uses TSL. TSL, at least as far as the PC-4 group is concerned, is really ONLY a start-up macro.

It would have been nice if the KeyDefine program included a start-up macro as well - then I wouldn't be asking you for a favor.

Anyway, the only reason I am interested in EXCHANGE is for the TSL. However, it may turn out to be more of a hassle than pushing that single key to invoke the KeyDefine macro at the Archive opening screen. We'll see.

On another note, have been using QL-PC FileServer since writing you recently. Have been using it to spellcheck and edit a QL _doc.

Developed 2 macros. using Turbo +: The first takes my standard page(s) of text and exports it, via a printer_dat set up for only CRs, to the PC drive D: (a 100K ram/vdisk).

The PC is started with a disk in drive A: and, via the AUTOEXEC.BAT file without further ado, it starts the PC side of QL-PC FileServer; the PC then pauses for a keypress.

Once the QL _doc file is sent over to the vdisk on drive D:. I press any key on the PC. When this happens, it loads PFS:1st Choice and then by a 1st Choice macro the QL file in D: is loaded and spellchecking begun.

When spellchecking and editing are finished another 1st Choice macro sends the file back to drive D: and ready for import into the QL.

The second QL macro does just that, actually the reverse of the first, getting the file back from the PC's drive D: and importing it into Quill by line, and re-setting the margins, Justification, etc.

Although somewhat complicated to explain, it works rather well and much easier than it appears above. PFS:1st Choice is, if you don't know, also a suite like EXCHANGE comprising a Word Processor, Spreadsheet (one of the worst I've seen), a Data Base that makes Archive look like it cost \$500, and a Telecommunications program, etc., etc.

The Word Processor is not bad. It contains both a spellchecker and a Thesaurus. The only reason I bought it (US\$40 at Damark) was the price. You are probably finding out, as you now have access to a PC, software vendors are closing out their DOS products to make shelf space available for their Windows versions. Old fuds like myself, who shall probably never use Windows, are finding bargains amongst their close-outs.

Am enclosing my usual US\$1.00, trust it's sufficient.

Thanks in advance,

jej

NO TORONTO FEST IN 1994

With many regrets I have to announce there will NOT be a Sinclair Fest in Toronto in 1994.

I would like to express my appreciation to the many who took time and trouble to send in the questionnaire which was in the last issue of Sinc-Link, stating they would like to come and whom they would like to see at such a Fest.

To others who were unable to reply owing to the deadline, I also give my thanks.

I am sorry to have to write this letter, but I feel that to go ahead with this project would be too much for me to accomplish on my own.

Thank you, one and all.

Hugh H. Howie.

November 1/93

XCHANGE QUESTION ANSWERED

In the September issue I mentioned that I thought XCHANGE, the recently released to Public Domain program, was multi-tasking, that I had heard it was not really a multitasking program and could anyone enlighten me.

The other night I received a call from Howard Clase and in a few short words he showed me the difference between such as Taskmaster and Xchange.

Taskmaster, which is a true multitasker, keeps plugging along when the program is changed, whereas Xchange stops work on a program when you switch to another.

So that means you can xchange one program for another, but operation of the first is suspended while you are away from it.

Taskmaster works in the background while you are away on something else.

My question is answered, and a big Thank You to Howard Clase for the call.

Hugh Howie

X C H A N G E

Complete program now in library

I have just received from Howard Clase a copy of XCHANGE, the Psion program recently released to Public Domain.

This is the latest PD version from the West Midlands Quanta Sub Group, and is on a DD disk, utilising all space on that disk, (3/1440 sectors).

It has been placed in our library bearing the title of "Xch-1" Should anyone wish a copy the usual rules apply, send a formatted disk plus return post and packing, and I will get it back to you as soon as possible.

I have just added an ED drive, so can supply programs on 3 1/2 DD, HD, ED, and also on 5 1/4 360 or 720. So that should cover whatever you might have.

On behalf of the club and its members,
Many thanks Howard.

Hugh Howie

**DID
YOU
KNOW
?**

I have enjoyed the quality of our newsletter and I have been awed by the knowledge of some of our contributors. I guess that I have assumed that everyone else has all the information that I have available to me. Our last newsletter and the out of town letter from George reminded me what assuming does. The request for information about the code for our DOS and the comment that a schematic is unavailable triggered me. Ken Schoenberger has written a good disassembler and he printed out the code and added remarks as he analyzed it. It is 38 pages long with two columns per page. Rather than print out a copy to send in (I have nearly worn mine out and Ken's comments would be lost) I have started to key it in to MScript so that it can be printed out at 15 CPI with narrow spacing. My Christmas present was a Canon Bubble Jet Printer and it seems readable at this smaller type. It is labor intensive so this installment will cover the first third of the code. The code in the "forbidden area" was given to us and I have not verified it. I hope to do this before the last installment and I will report back later.

The second item, the cartridge and I/F schematic, was something I was curious about when I first got my Larken system. I had looked over the cartridge and had started to put it on paper. To verify that I had it right I built a unit on a discarded Zebra board. After one false start with one wiring error it worked.

So the cartridge schematic should be correct. I did not recognize the marking on one diode, so I merely recorded what was on it. I used a diode that looked like it and it worked.

I do have to apologize for not using a Timex to do the schematic, but I thought it would be clearer done on a Laser printer.

I have not done anything on the disk I/F. If anyone is seriously interested in the schematic for it I could work on it after I complete the disassembly of the cartridge.

If any one knows of a m/c patch that would allow me to send the disassembly to memory instead of screen or printer, I would be interested in trying that rather than retyping the remaining pages.

My typing skills are far from perfect, so in case anyone is interested in verifying a particular piece of code here is how it was done:

1. Clear 39999
2. Load in TSDB at 60000
3. Key in a short basic routine to do a LKdos peek and poke to 40000 + address
4. Set offset so that proper addresses show in print out (send disassembly to screen or)
5. Set margin for left column
6. set printer on in TSDB
7. disassemble until page is full
8. reset margin to print right column
9. disassemble until page is full
10. repeat 5 thru 9 for next page

Les Cottrell 108 River Heights Drive Cocoa, FL 32922-6630

THIS IS THE CONCLUSION OF LES'S DISASSEMBLY. SEE ISSUES 11-2,11-3 & 11-5
ED.

name	Dec	Hex	Instr-Dec	Remarks	(16)	name	Dec	Hex	Instr-Dec	Remarks
	5710	164E	JR C, 5719			5900	170C	LD A, (16063)	; bottom	
	5712	1650	CP 165			5903	170F	CP B		
	5714	1652	JP NC, 5985			5904	1710	JR C, 5910		
	5717	1655	LD A, 35	; #		5906	1712	CALL 6023		
	5719	1657	LD HL, 0			5909	1715	RET		
	5722	165A	LD L, A			5910	1716	LD A, B		
	5723	1656	ADD HL,HL			5911	1717	SUB 8		
	5724	1657	ADD HL,HL			5913	1719	LD (16066), A		
	5725	1658	ADD HL,HL			5916	171C	RET		
	5726	165E	EX DE, HL			5917	171D	LD HL, (8218)	; chars	
	5727	165F	LD HL, (8218)	; chars		5920	1720	INC H		
	5730	1662	ADD HL, DE			5921	1721	LD C, (HL)		
	5731	1663	LD A, (HL)			5922	1722	LD B, 8		
	5732	1664	LD (16079), A			5924	1724	AND A		
	5735	1667	PUSH HL			5925	1725	SLA C		
	5736	1658	CALL 5860			5927	1727	JR NC, 5931		
	5739	166B	POP HL			5929	1729	DJNZ 5924		
	5740	166C	LD DE, 16080			5931	172B	LD A, (16060)	; left	
	5743	166F	LD BC, 8			5934	172E	LD C, A		
	5746	1672	LDIR			5935	162F	LD A, (16065)	; X	
	5748	1674	LD (16077), DE			5938	1732	SUB B		
	5752	1678	LD BC, (16065)	; X		5939	1733	CP C		
	5755	167B	CALL 5967			5940	1734	RET C		
	5759	167F	LD (16076), A ; Y ?			5941	1735	LD (16065), A ; X		
	5762	1682	LD (16068), HL ; screen address			5944	1738	RET		
	5765	1685	LD B, 8			5945	1739	PUSH AF		
	5767	1687	PUSH BC			5946	173A	PUSH HL		
	5768	1688	LD HL, (16077)			5947	173B	LD A, D		
	6771	168B	LD A, (HL)			5948	173C	RRC A		
	5772	168C	DEC HL			5950	173E	RRC A		
	5773	168D	LD (16077), HL			5953	1740	RRC A		
	5776	1690	LD L, A			5954	1742	AND 3		
	5777	1691	LD A, (16076)			5956	1744	OR 88	; X	
	5780	1694	AND A			5958	1746	LD H, A		
	5781	1695	JR Z, 5793			5959	1747	LD L, E		
	5783	1697	LD B, A			5960	1748	LD A, (16074)	; win attr	
	5784	1698	LD H, 0			5963	174B	LD (HL), A		
	5786	169A	SRL L			5964	174C	POP HL		
	5788	169C	RR H			5965	174D	POP AF		
	5790	169E	AND A			5966	174E	RET		
	5791	169F	DJNZ 5786			5967	174F	LD A, (23632)	; chans (hi)	
	5793	16A1	LD DE, (16068)	; screen address		5970	1752	CP 93	;]	
	5797	16A5	LD A, (DE)			5972	1754	JR C, 5980		
	5798	16A6	XOR L			5974	1756	LD DE, 9731	; scrmb1 (2068)	
	5799	16A7	LD (DE), A			5977	1759	JP 19		
	5800	16A8	CALL 5945			5980	175C	LD DE, 8874	; (spectrum)screen address	
	5803	16AB	LD A, (16076)			5983	175F	JR 5977		
	5806	16AE	AND A			5985	1761	POP AF		
	5807	16AF	JR Z, 5814			5986	1762	POP AF		
	5809	16B1	INC DE			5987	1763	POP BC		
	5810	16B2	LD A, (DE)			5988	1764	POP DE		
	5811	16B3	LD A, H			5989	1765	POP HL		
	5812	16B4	LD (DE), A			5990	1766	LD (15994), A		
	5813	16B5	DEC DE			5993	1769	LD (15996), DE		
	5814	16B6	LD HL, (16068)	; screen address		5997	176D	LD A, (23632)	; chans(hi)	
	5817	16B9	CALL 5841			6000	1770	CP 93	;]	
	5820	16BC	LD (16068), HL ; screen address			6002	1772	JR C, 6009		
	5823	16BF	POP BC			6004	1774	LD DE, 1280	; sendtv(2068)	
	5824	1600	DJNZ 5797			6007	1777	JR 6012		
	5826	1602	LD A, (16079)			6009	1779	LD DE, 2548	; send char to scn(specy)	
	5829	16C5	LD A, (16065)	; X		6012	177C	PUSH DE		
	5832	16C8	LD B, A			6013	177D	LD A, (15994)		
	5833	16C9	LD A, (16070)	; char width		6016	1780	LD DE, (15996)		
	5836	16CC	ADD A, B			6020	1784	JP 186	; jpout	
	5837	16CD	LD (16065), A ; X			6023	1787	LD A, (16060)	; left	
	5840	16D0	RET			6026	178A	LD B, A		
	5841	16D1	PUSH AF			6027	178B	LD A, (16062)	; right	
	5842	16D2	LD A, H			6030	1781	SUB B		
	5843	16D3	DEC H			6031	178F	SRL A		
	5844	16D4	AND 7			6033	1791	SRL A		
	5846	16D6	JR NZ, 5858			6035	1793	SRL A		
	5848	16D8	LD A, L			6037	1795	INC A		
	5849	16D9	SUB 32			6038	1796	LD (13434), A		
	5851	16DB	LD L, A			6041	1799	XOR A		
	5852	16DC	JR C, 5858			6042	179A	LD (13435), A		
	5854	16DE	LD A, H			6045	179D	LD A, (16063)	; bottom	
	5855	16DF	ADD A, 8			6048	17A0	LD B, A		
	5857	16E1	LD H, A			6049	17A1	LD A, (16061)	; top	
	5858	16E2	POP AF			6052	17A4	SUB B		
	5859	16E3	RET			6053	17A5	SCF		
	5860	16E4	LD B, 8			6054	17A6	CCF		
	5862	16E6	LD A, (16079)			6055	17A7	RRA		
	5865	16E9	LD C, A			6056	17A8	RRA		
	5866	16EA	AND A			6057	17A9	RRA		
	5867	16EB	SRL C			6058	17AA	LD (13443), A		
	5869	16ED	JR NC, 5874			6061	17AD	LD BC, (16060)	; left	
	5871	16EF	DEC B			6065	17B1	LD (16072), BC		
	5872	16F0	JR 5866			6069	17B5	LD A, 8		
	5874	16F2	LD A, E			6071	17B7	LD (13442), A		
	5875	16F3	LD (16070), A ; char width			6074	17BA	LD BC, (16072)		
	5878	16F6	LD A, (16065)	; X		6078	17BE	CALL 5967		
	5881	16F9	ADD A, B			6081	17C1	EX DE, HL		
	5882	16FA	JR C, 5890			6082	17C2	LD BC, (16072)		
	5884	16FC	LD B, A			6086	17C6	LD A, B		
	5885	16FD	LD A, (16062)	; right		6087	17C7	SUB 8		
	5888	1700	CP B			6089	17C9	LD B, A		
	5889	1701	RET NC			6090	17CA	LD (16072), BC		
	5890	1702	LD A, (16060)	; left		6094	17CE	PUSH DE		
	5878	16F6	LD (16065), A ; X			6095	17CF	CALL 5967		
	5896	1708	LD A, (16066)	; Y		6098	17D2	POP DE		
	5899	170B	LD B, A			6099	17D3	LD BC, (13434)		

name	Dec	Hex	Instr-Dec	Remarks	(17)	name	Dec	Hex	Instr-Dec	Remarks
6103	17D7		LD (13436), HL			6311	18A7		INC A	
6106	17DA		PUSH HL			6312	18A8		LD (13434), A	
6107	17DB		PUSH DE			6315	18AB		XOR A	
6108	17DC		PUSH BC			6316	18AC		LD (13435), A	
6109	17DD		LDIR			6319	18AF		LD A, (16063) ; bottom	
6111	17DF		POP BC			6322	18B2		LD B, A	
6112	17E0		POP DE			6323	18B3		LD A, (16061) ; top	
6113	17E1		POP HL			6326	18B6		SUB B	
6114	17E2		DEC H			6327	18B7		SCF	
6115	17E3		DEC D			6328	18B8		CCF	
6116	17E4		LD A, (13442)			6329	18B9		RRA	
6119	17E7		DEC A			6330	18BA		RRA	
6120	17E8		AND A			6331	18BB		RRA	
6121	17E9		JR Z, 6128			6332	18BC		INC A	
6123	17EB		LD (13442), A			6333	18BD		LD (13443), A	
6126	17EE		JR 6106			6336	18C0		LD BC, (16060) ; left	
6128	17F0		LD A, (13443)			6340	18C4		LD (16072), BC	
6131	17F3		DEC A			6344	18C8		LD A, 8	
6132	17F4		AND A			6346	18CA		LD (13442), A	
6133	17F5		JR Z, 6140			6349	18CD		LD BC, (16072)	
6135	17F7		LD (13443), A			6353	18D1		CALL 5967	
6138	17FA		JR 6069			6356	18D4		EX DE, HL	
6140	17FC		LD B, 8			6357	18D5		LD BC, (16072)	
6142	17FE		LD HL, (13436)			6361	18D9		LD A, B	
6145	1801		PUSH HL			6362	18DA		SUB 8	
6146	1802		LD A, (13434)			6364	18DC		LD B, A	
6149	1805		LD (HL), 0			6365	18DD		LD (16072), BC	
6151	1807		INC HL			6369	18E1		CP 200	
6152	1808		DEC A			6371	18E3		JR C, 6383	
6153	1809		JR NZ, 6149			6373	18E5		PUSH DE	
6155	180B		POP HL			6374	18E6		POP HL	
6156	180C		DEC H			6375	18E7		LD BC, 32	
6157	180D		PUSH HL			6378	18EA		OR A	
6158	180E		DJNZ 6146			6379	18EB		SBC HL, BC	
6160	1810		POP HL			6381	18ED		JR 6388	
6161	1811		LD A, (16064) ; scroll count			6383	18EF		PUSH DE	
6164	1814		DEC A			6384	18F0		CALL 5967	
6165	1815		LD (16064), A ; scroll count			6387	18F3		POP DE	
6168	1818		AND A			6388	18F4		LD BC, (13434)	
6169	1819		RET NZ			6392	18F8		LD (13436), HL	
6170	181A		LD A, (16079)			6395	18FB		PUSH HL	
6173	181D		LD (13438), A			6396	18FC		PUSH DE	
6176	1820		LD HL, 6202 ; * SCROLL ??			6397	18FD		PUSH BC	
6179	1823		LD B, 10			6398	18FE		CALL 6476	
6181	1825		LD A, (HL)			6401	1901		POP BC	
6182	1826		PUSH AF			6402	1902		POP DE	
6183	1827		PUSH BC			6403	1903		POP HL	
6184	1828		PUSH HL			6404	1904		DEC H	
6185	1829		CALL 5667			6405	1905		DEC D	
6188	182C		POP HL			6406	1906		LD A, (13442)	
6189	182D		POP BC			6409	1909		DEC A	
6190	182E		POP AF			6410	190A		AND A	
6191	182F		INC HL			6411	190B		JR Z, 6418	
6192	1830		DJNZ 6181			6413	190D		LD (13442), A	
6194	1832		LD A, (13438)			6416	1910		JR 6395	
6197	1835		LD (16079), A			6418	1912		LD A, (13443)	
6200	1838		JR 6217			6421	1915		DEC A	
6202			* SCROLL ??	<pointer @ 6176		6422	1916		AND A	
6217	1849		CALL 7841			6422	1917		JR Z, 6430	
6220	184C		LD A, 191			6425	1919		LD (13443), A	
6222	184E		IN A, 254			6428	191C		JR 6344	
6224	1850		CP 31			6430	191E		LD HL, (16060) ; left	
6226	1852		JR Z, 6217 ; not n/l, l, k,j or h			6433	1921		LD (16065), HL ; X	
6228	1854		LD A, (16063) ; bottom			6436	1924		CALL 6442	
6231	1857		LD B, A			6439	1927		JP 4499	
6232	1858		LD A, (16061) ; top			6442	192A		LD A, (16071)	
6235	185B		SUB B			6445	192D		AND A	
6236	185C		RRA			6446	192E		JR Z, 6454	
6237	185D		RRA			6448	1930		CP 1	
6238	185E		RRA			6450	1932		JR Z, 6459	
6239	185F		LD (16064), A ; scroll count			6452	1934		JR 6464	
6242	1862		LD A, (16060) ; left			6454	1936		LD DE, 16000 ; window 0 info	
6245	1865		LD (16065), A ; X			6457	1939		JR 6467	
6248	1868		JP 6140			6459	193B		LD DE, 16020 ; window 1 info	
6251	186B		LD (8222), A			6462	193E		JR 6467	
6254	186E		AND A			6454	1936		LD DE, 16040 ; window 2 info	
6255	186F		JR Z, 6263			6467	1943		LD HL, 16060 ; current window info	
6257	1871		CP 1			6470	1946		LD BC, 19	
6259	1873		JR Z, 6268			6473	1949		LDIR	
6261	1875		JR 6273			6475	194B		RET	
6263	1877		LD HL, 16000 ; window 0 info			6476	194C		LD (HL), 0	
6266	187A		JR 6276			6478	194E		XOR A	
6268	187C		LD HL, 16020 ; window 1 info			6479	194F		LD (DE), A	
6271	187F		JR 6276			6480	1950		CALL 5945	
6273	1881		LD HL, 16040 ; window 2 info			6483	1953		INC HL	
6276	1884		LD DE, 16060			6484	1954		INC DE	
6279	1887		LD BC, 20			6485	1955		DEC BC	
6282	188A		LDIR			6486	1956		LD A, C	
6284	188C		RET			6487	1957		OR B	
CLEAR	6285	188D	RST 32 ; next char			6488	1958		JR NZ, 6476	
6286	188E		CALL 5369			6490	195A		RET	
6289	1891		CP 3			VERIFY	6491	195B	XOR A	
6291	1893		JP NC, 4554			6492	195C		LD (8221), A ; curtrk	
6294	1896		CALL 6251			6495	195F		CALL 126 ; track	
6297	1899		LD A, (16060) ; left			6498	1962		CALL 123 ; loadbf	
6300	189C		LD B, A			6501	1965		LD HL, (8324) ; varofset	
6301	189D		LD A, (16062) ; right			6504	1968		LD A, H	
6304	18A0		SUB B			6505	1969		ADD A, A	
6305	18A1		SRL A			6506	196A		LD H, A	
6307	18A3		SRL A			6507	196B		LD (8245), HL ; temp6	
6309	18A5		SRL A			6510	196E		CALL 123 ; loadbf	

name	Dec	Hex	Instr-Dec	Remarks	(18)
	6513	1971	CALL 129	; nexttr	
	6516	1974	LD HL, (8245)	; temp6	
	6519	1977	PUSH HL		
	6520	1978	LD A, L		
	6521	1979	CP 2		
	6523	197B	JR Z, 6528		
	6525	197D	CALL 129	; nexttr	
	6528	1980	LD A, (8221)	; curtrk	
	6531	1983	POP HL		
	6532	1984	CP H		
	6533	1985	JR C, 6510		
	6535	1987	JP 4499		
	6538	198A	RST 32	; next char	
	6539	198B	CALL 144	; eval	
	6542	198E	LD (8241), BC	; temp2	
	6546	1992	LD A, B		
	6547	1993	AND A		
	6548	1994	JR NZ, 6585	; integer out of range	
	6550	1996	RST 32	; next char	
	6551	1997	CALL 144	; eval	
	6554	199A	LD A, B		
	6555	199B	AND A		
	6556	199C	JR NZ, 6585	; integer out of range	
	6558	199E	LD A, C		
	6559	199F	LD (8242), A	temp2+1	
	6562	19A2	LD HL, (23645)	; chadd	
	6565	19A5	LD A, (HL)		
	6566	19A6	CP 44	; ,	
	6568	19A8	JR NZ, 6583	; parameter error	
	6570	19AA	RST 32	; next char	
	6571	19AB	CALL 144	; eval	
	6574	19AE	LD A, B		
	6575	19AF	AND A		
	6576	19B0	JR NZ, 6585	; integer out of range	
	6578	19B2	LD A, C		
	6579	19B3	LD (8222), A		
	6582	19B6	RET		
	6583	19B7	RST 8 ERR 26	; parameter error	
	6585	19B9	RST 8 ERR 11	; integer out of range	
TABLE	with <pointer	@ 7197			
	6587	00 00	00 00 00 00 00 00		
	6595	FF FF	FF FF FF FF FF FF		
	6603	55 AA	55 AA 55 AA 55 AA		
	6611	CC CC	33 33 CC CC 33 33		
	6619	C0 C0	0C 0C C0 C0 0C 0C		
	6627	C0 C0	00 00 0C 0C 00 00		
	6635	88 44	22 11 88 44 22 11		
	6643	11 22	44 88 11 22 44 88		
	6651	FF 08	08 08 FF 80 80 80		
	6659	18 24	42 81 81 42 24 18		
DRAW	6667	1A0B	LD HL, (23677)	; coords	
	6670	1A0E	LD A, H		
	6671	1A0F	ADD A, 15		
	6673	1A11	LD H, A		
	6674	1A12	LD (8249), HL		
	6677	1A15	CALL 6538		
	6680	1A18	CALL 141	; endln	
	6683	1A1B	LD HL, (8241)	; temp2	
	6686	1A1E	XOR A		
	6687	1A1F	CP H		
	6688	1A20	JP Z, 4499		
	6691	1A23	CP L		
	6692	1A24	JP Z, 4499		
	6695	1A27	JP 6821		
	6698	1A2A	LD BC, (8247)		
	6702	1A2E	LD A, 191		
	6704	1A30	SUB B		
	6705	1A31	RET C		
	6706	1A32	LD B, A		
	6707	1A33	AND 192		
	6709	1A35	RRA		
	6710	1A36	RRA		
	6711	1A37	RRA		
	6712	1A38	LD H, A		
	6713	1A39	LD A, B		
	6714	1A3A	AND 7		
	6716	1A3C	OR H		
	6717	1A3D	OR 64		
	6719	1A3F	LD H, A		
	6720	1A40	LD A, C		
	6721	1A41	RLCA		
	6722	1A42	RLCA		
	6723	1A43	RLCA		
	6724	1A44	AND 199		
	6726	1A46	LD L, A		
	6727	1A47	LD A, B		
	6728	1A48	AND 56		
	6730	1A4A	OR L		
	6731	1A4B	RLCA		
	6732	1A4C	RLCA		
	6733	1A4D	LD L, A		
	6734	1A4E	LD A, C		
	6735	1A4F	AND 7		
	6737	1A51	LD B, A		
	6738	1A52	LD A, (8222)		
	6741	1A55	LD C, A		
	6742	1A56	LD A, 1		
	6744	1A58	INC B		
	6745	1A59	RRCA		
	6746	1A5A	RLC C		
	6748	1A5C	DJNZ 6745		

name	Dec	Hex	Instr-Dec	Remarks
	6750	1A5E	RET	
	6751	1A5F	BIT 0, C	
	6753	1A61	JR Z, 6758	
	6755	1A63	OR (HL)	
	6756	1A64	LD (HL), A	
	6757	1A65	RET	
	6758	1A66	CPL	
	6759	1A67	AND (HL)	
	6760	1A68	LD (HL), A	
	6761	1A69	RET	
	6762	1A6A	LD A, (8239)	
	6765	1A6D	AND A	
	6766	1A6E	RET Z	
	6767	1A6F	CALL 6698	
	6770	1A72	CALL 6751	
	6773	1A75	EX DE, HL	
	6774	1A76	LD HL, 8239	
	6777	1A79	DEC (HL)	
	6778	1A7A	RET Z	
	6779	1A7B	LD HL, 8247	
	6782	1A7E	INC (HL)	
	6783	1A7F	LD A, (HL)	
	6784	1A80	AND 7	
	6786	1A82	AND A	
	6787	1A83	JR NZ, 6767	
	6789	1A85	EX DE, HL	
	6790	1A86	INC HL	
	6791	1A87	LD A, (8239)	
	6794	1A8A	SUB 8	
	6796	1A8C	JR C, 6767	
	6798	1A8E	CP 8	
	6800	1A90	JR C, 6767	
	6802	1A92	LD (8239), A	
	6805	1A95	LD A, (8222)	
	6808	1A98	LD (HL), A	
	6809	1A99	LD A, (8247)	
	6812	1A9C	ADD A, 8	
	6814	1A9E	LD (8247), A	
	6817	1AA1	AND A	
	6818	1AA2	JR NZ, 6790	
	6820	1AA4	RET	
	6821	1AA5	LD A, (8242)	
	6824	1AA8	LD (8240), A	; temp1
	6827	1AAB	LD HL, 6587	
	6830	1AAE	LD A, (8222)	
	6833	1AB1	AND A	
	6834	1AB2	JR Z, 6860	
	6836	1AB4	CP 10	
	6838	1AB6	JR NZ, 6845	
	6840	1AB8	LD HL, 23540	
	6843	1AB8	JR 6860	
	6845	1ABD	CP 11	
	6847	1ABF	JP NC, 6585	
	6850	1AC2	LD B, A	
	6851	1AC3	XOR A	
	6852	1AC4	ADD A, 8	
	6854	1AC6	DJNZ 6852	
	6856	1AC8	LD C, A	
	6857	1AC9	LD B, 0	
	6859	1ACB	ADD HL, BC	
	6860	1ACC	LD (8243), HL	; temp4
	6863	1ACF	LD HL, (8249)	
	6866	1AD2	LD A, H	
	6867	1AD3	AND 7	
	6869	1AD5	LD (8328), A	; datablock
	6872	1AD8	LD HL, (8243)	; temp4
	6875	1ADB	LD C, A	
	6876	1ADC	LD B, 0	
	6878	1ADE	ADD HL, BC	
	6879	1ADF	INC HL	
	6880	1AE0	LD (8245), HL	; temp6
	6883	1AE3	LD A, (HL)	
	6884	1AE4	LD (8222), A	
	6887	1AE7	LD A, (8241)	; temp2
	6890	1AEA	LD (8239), A	
	6893	1AED	LD BC, (8249)	
	6897	1AF1	INC B	
	6898	1AF2	LD A, B	
	6899	1AF3	CP 192	
	6901	1AF5	JR NC, 6949	
	6903	1AF7	LD (8249), BC	
	6907	1AFB	LD (8347), BC	
	6911	1AFF	LD A, (8328)	; datablock
	6914	1B02	INC A	
	6915	1B03	CP 8	
	6917	1B05	JR NZ, 6926	
	6919	1B07	LD H, (8243)	; temp4
	6922	1B0A	LD (8245), HL	; temp6
	6925	1B0D	XOR A	
	6926	1B0E	LD (8328), A	; datablock
	6929	1B11	LD HL, (8245)	; temp6
	6932	1B14	LD A, (HL)	
	6933	1B15	INC HL	
	6934	1B16	LD (8245), HL	; temp6
	6937	1B19	LD (8222), A	
	6940	1B1C	CALL 6767	
	6943	1B1F	LD HL, 8240	; temp1
	6946	1B22	DEC (HL)	
	6947	1B23	JR NZ, 6887	
	6949	1B25	JP 4499	
CIRCLE	6952	1B28	CALL 6538	

name	Dec	Hex	Instr-Dec	Remarks	(19)	name	Dec	Hex	Instr-Dec	Remarks
	6955	1B2B	LD HL, (8241)	; temp2			7153	1BF1	DEC B	
	6958	12BE	LD E, 16				7154	1BF2	LD (8243), BC	; temp4
	6960	1B30	ADD A, H				7158	1BF6	EX DE, HL	
	6961	1B31	LD H, A				7159	1BF7	CALL 6702	
	6962	1B32	CP 191				7162	1BFA	LD A, (HL)	
	6964	1B34	JR C, 6968				7163	1BFB	EX DE, HL	
	6966	1B36	RST 8 ERR 11	; integer out of range			7164	1BFC	CP 255	; end of track map
	6968	1B38	LD (8249), HL				7166	1BFE	RET Z	
	6971	1B3B	LD (8247), HL				7167	1BFF	LD BC, (8243)	; temp4
	6974	1B3E	CALL 6698				7171	1C03	CALL 6702	
	6977	1B41	LD D, A				7174	1C06	AND (HL)	
	6978	1B42	AND (HL)				7175	1C07	AND A	
	6979	1B43	AND A				7176	1C08	JR Z, 7189	
	6980	1B44	JP NZ, 4499				7178	1C0A	LD BC, (8243)	; temp4
	6983	1B47	XOR A				7182	1C0E	DEC C	
	6984	1B48	LD (8240), A	; temp1			7183	1C0F	LD (8243), BC	; temp4
	6987	1B4B	LD (8239), A				7187	1C13	JR 7171	
	6990	1B4E	JP 7197				7189	1C15	LD HL, (8243)	; temp4
	6993	1B51	CALL 6698				7192	1C18	LD A, L	
	6996	1B54	AND (HL)				7193	1C19	LD (8239), A	
	6997	1B55	AND A				7196	1C1C	RET	
	6998	1B56	JR NZ, 7013				7197	1C1D	LD HL, 6587	
	7000	1B58	LD HL, (8247)				7200	1C20	LD A, (8222)	
	7003	1B5B	INC L				7203	1C23	AND A	
	7004	1B5C	LD A, L				7204	1C24	JP Z, 7230	
	7005	1B5D	AND A				7206	1C26	CP 10	
	7006	1B5E	JRZ, 7024				7208	1C28	JR NZ, 7215	
	7008	1B60	LD (8247), HL				7210	1C2A	LD HL, 23540	
	7011	1B63	JR 6993				7213	1C2D	JR 7230	
	7013	1B65	LD HL, (8247)				7215	1C2F	CP 11	
	7016	1B68	DEC L				7217	1C31	JP NC, 6585	
	7017	1B69	LD A, L				7220	1C34	LD B, A	
	7018	1B6A	CP 255	; end of track map			7221	1C35	XOR A	
	7020	1B6C	RET Z				7222	1C36	ADD A, 8	
	7021	1B6D	LD (8247), HL				7224	1C38	DJNZ 7222	
	7024	1B70	LD HL, (8247)				7226	1C3A	LD C, A	
	7027	1B73	LD A, L				7227	1C3B	LD B, 0	
	7028	1B74	AND 7				7229	1C3D	ADD HL, BC	
	7030	1B76	CP 7				7230	1C3E	LD (13434), HL	
	7032	1B78	JR Z, 7059				7233	1C41	LD (13436), HL	
	7034	1B7A	CALL 6698				7236	1C44	LD A, (8248)	
	7037	1B7D	LD D, A				7239	1C47	AND 7	
	7038	1B7E	AND (HL)				7241	1C49	LD (8328), A	; datablock
	7039	1B7F	RET NZ				7244	1C4C	LD HL, (13434)	
	7040	1B80	LD A, D				7247	1C4F	LD C, A	
	7041	1B81	CALL 6751				7248	1C50	LD B, 0	
	7044	1B84	CALL 7096				7250	1C52	ADD HL, BC	
	7047	1B87	LD HL, (8247)				7251	1C53	LD (13434), HL	
	7050	1B8A	LD A, L				7254	1C56	LD A, (HL)	
	7051	1B8B	AND A				7255	1C57	LD (8222), A	
	7052	1B8C	RET Z				7258	1C5A	XOR A	
	7053	1B8D	DEC L				7259	1C5B	LD (8240), A	; temp1
	7054	1B8E	LD (8247), HL				7262	1C5E	CALL 6993	
	7057	1B91	JR 7024				7265	1C61	LD A, (8239)	
	7059	1B93	CALL 6698				7268	1C64	AND A	
	7062	1B96	LD A, (HL)				7269	1C65	JR Z, 7325	
	7063	1B97	AND A				7271	1C67	LD A, (8328)	; datablock
	7064	1B98	JR NZ, 7034				7274	1C6A	INC A	
	7066	1B9A	LD A, (8222)				7275	1C6B	CP 8	
	7069	1B9D	LD (HL), A				7277	1C6D	JR Z, 7291	
	7070	1B9E	LD (8245), HL	; temp6			7279	1C6F	LD (8328), A	; datablock
	7073	1BA1	CALL 7134				7282	1C72	LD HL, (13434)	
	7076	1BA4	LD HL, (8247)				7285	1C75	INC HL	
	7079	1BA7	XOR A				7286	1C76	LD (13434), HL	
	7080	1BA8	LD A, L				7289	1C79	JR 7301	
	7081	1BA9	SBC A, 8				7391	1C7B	LD HL, (13436)	
	7083	1BAB	CP 255	; end of track map			7294	1C7E	LD (13434), HL	
	7085	1BAD	RET Z				7297	1C81	XOR A	
	7086	1BAE	LD L, A				7298	1C82	LD (8328), A	; datablock
	7087	1BAF	LD (8247), HL				7301	1C85	LD A, (HL)	
	7090	1BB2	LD HL, (8245)	; temp6			7302	1C86	LD (8222), A	
	7093	1BB5	DEC HL				7305	1C89	LD A, (8239)	
	7094	1BB6	JR 7062				7308	1C8C	LD HL, (8247)	
	7096	1BB8	LD A, (8239)				7311	1C8F	LD L, A	
	7099	1BBB	AND A				7312	1C90	INC H	
	7100	1BBC	RET NZ				7313	1C91	LD (8247), HL	
	7101	1BBD	LD A, (8240)	; temp1			7316	1C94	XOR A	
	7104	1BC0	LD BC, (8247)				7317	1C95	LD (8239), A	
	7108	1BC4	CP 1				7320	1C98	LD A, H	
	7110	1BC6	JR Z, 7115				7321	1C99	CP 191	
	7112	1BC8	INC B				7323	1C9B	JR C, 7262	
	7113	1BC9	JR 7116				7325	1C9D	LD A, 1	
	7115	1BCB	DEC B				7327	1C9F	LD (8240), A	; temp1
	7116	1BCC	LD (8243), BC	; temp4			7330	1CA2	LD HL, (8249)	
	7120	1BD0	CALL 6702				7333	1CA5	DEC H	
	7123	1BD3	AND (HL)				7334	1CA6	LD (8247), HL	
	7124	1BD4	AND A				7337	1CA9	LD A, H	
	7125	1BD5	RET NZ				7338	1CAA	AND 7	
	7126	1BD6	LD HL, (8243)	; temp4			7340	1CAC	LD (8328), A	; datablock
	7129	1BD9	LD A, L				7343	1CAD	LD HL (13436), A	
	7130	1BDA	LD (8239), A				7346	1CB2	LD C, A	
	7133	1BDD	RET				7347	1CB3	LD B, 0	
	7134	1BDE	LD A, (8239)				7349	1CB5	ADD HL, BC	
	7137	1BE1	AND A				7350	1CB6	LD (13434), HL	
	7138	1BE2	RET NZ				7353	1CB9	LD A, (HL)	
	7139	1BE3	LD A, (8240)	; temp1			7354	1CBA	LD (8222), A	
	7142	1BE6	LD BC, (8247)				7357	1CBD	LD HL, (13436)	
	7146	1BEA	CP 1				7360	1CC0	LD BC, 7	
	7148	1BEC	JR Z, 7153				7363	1CC3	ADD HL, BC	
	7150	1BEE	INC B				7364	1CC4	LD (13436), HL	
	7151	1BEF	JR 7154				7367	1CC7	CALL 6993	

name	Dec	Hex	Instr-Dec	Remarks	(20)	name	Dec	Hex	Instr-Dec	Remarks
	7370	1CCA	LD A, (8239)				7550	1D7E	LD DE, (15990)	
	7373	1CCD	AND A				7554	1D82	LDIR	
	7374	1CCE	JR Z, 7430				7556	1D84	POP BC	
	7376	1CD0	LD A, (8328)	; datablock			7557	1D85	LD HL, (23627)	; vars
	7379	1CD3	DEC A				7560	1D88	DEC HL	
	7380	1CD4	CP 255	; end of track map			7561	1D89	LD (23627), HL	; vars
	7382	1CD6	JR Z, 7396				7564	1D8C	RET	
	7384	1CD8	LD (8328), A	; datablock		MERGE	7565	1D8D	CALL 138	; cmdck
	7387	1CDB	LD HL, (13434)				7568	1D90	LD HL, 8226	; progmn
	7390	1CDE	DEC HL				7571	1D93	CALL 132	; indir
	7391	1CDF	LD (13434), HL				7574	1D96	LD A, (8224)	; errnu
	7394	1CE2	JR 7407				7577	1D99	CP 10	
	7396	1CE4	LD HL, (13436)				7579	1D9B	JP Z, 147	; nofil
	7399	1CE7	LD (13434), HL				7582	1D9E	CALL 135	; movdir
	7402	1CEA	LD A, 7				7585	1DA1	XOR A	
	7404	1CEC	LD (8328), A	; datablock			7586	1DA2	LD (8248), A	
	7407	1CEP	LD A, (HL)				7589	1DA5	LD HL, 8250	; directory
	7408	1CF0	LD (8222), A				7592	1DA8	INC HL	
	7411	1CF3	LD A, (8239)				7593	1DA9	LD A, 253	; 1st block
	7414	1CF6	LD HL, (8247)				7595	1DAB	CP (HL)	
	7417	1CF9	LD L, A				7596	1DAC	JR NZ, 7592	
	7418	1CFA	DEC H				7598	1DAE	LD (8245), HL	; temp6
	7419	1CFB	LD (8247), HL				7601	1DB1	LD HL, (8245)	; temp6
	7422	1CFE	XOR A				7604	1DB4	INC HL	
	7423	1CFF	LD (8239), A				7605	1DB5	LD (8245), HL	temp6
	7426	1D02	LD A, H				7608	1DB8	LD A, (HL)	
	7427	1D03	AND A				7609	1DB9	CP 249	; name end
	7428	1D04	JR NZ, 7367				7611	1DBB	JP Z, 7815	
	7430	1D06	JP 4499				7614	1DBE	LD (8221), A	; curtrk
	7433	1D09	LD HL, (15990)				7617	1DC1	CALL 126	; track
	7436	1D0C	INC HL				7620	1DC4	CALL 123	; loadbf
	7437	1D0D	INC HL				7623	1DC7	LD A, (8224)	; errnu
	7438	1D0E	LD C, (HL)				7625	1DCA	CP 25	
	7439	1D0F	INC HL				7628	1DCC	JP Z, 195	; dterr
	7440	1D10	LD B, (HL)				7631	1DCF	LD A, (8248)	
	7441	1D11	ADD HL, BC				7634	1DD2	CP 50	; 2
	7442	1D12	INC HL				7636	1DD4	JR Z, 7746	
	7443	1D13	LD (15990), HL				7638	1DD6	LD HL, 5090	; track length
	7446	1D16	RET				7641	1DD9	LD (8241), HL	; temp2
	7447	1D17	LD HL, (23635)	; prog			7644	1DDC	LD HL, 8328	; datablock
	7450	1D1A	LD (15990), HL				7647	1DDF	LD (8236), HL	; start
	7453	1D1D	LD HL, 13430				7650	1DE2	LD HL, (8241),	; temp2
	7456	1D20	LD D, (HL)				7653	1DE5	LD BC, 5	
	7457	1D21	INC HL				7656	1DE8	SBC HL, BC	
	7458	1D22	LD E, (HL)				7658	1DEA	JR C, 7722	
	7459	1D23	LD (15992), DE				7660	1DEC	LD HL, (8236)	
	7463	1D27	JR 7474				7663	1DEF	LF A, (HL)	
	7465	1D29	CALL 7433				7644	1DF0	CP 40	; (
	7468	1D2C	LD A, (HL)				7666	1DF2	JP NC, 7815	
	7469	1D2D	CP 40	; (7669	1DF5	INC HL	
	7471	1D2F	JR C, 7474				7670	1DF6	INC HL	
	7473	1D31	RET				7671	1DF7	LD C, (HL)	
	7474	1D32	LD HL, (15990)				7672	1DF8	INC HL	
	7477	1D35	LD B, (HL)				7673	1DF9	LD B, (HL)	
	7478	1D36	INC HL				7674	1DFA	INC BC	
	7479	1D37	LD C, (HL)				7675	1DFB	INC BC	
	7480	1D38	LD HL, (15992)				7676	1DFC	INC BC	
	7483	1D3B	OR A				7677	1DFD	INC BC	
	7484	1D3C	SBC HL, BC				7678	1DFE	LD A, B	
	7486	1D3E	RET C				7679	1DFE	CP 5	
	7487	1D3F	LD A, H				7691	1E01	JR NC, 7784	
	7488	1D40	OR L				7683	1E03	LD HL, (8241)	; temp2
	7449	1D41	JR NZ, 7465				7686	1E06	OR A	
	7491	1D43	LD HL, (15990)				7687	1E07	SBC HL, BC	
	7494	1D46	PUSH HL				7689	1E09	JR C, 7722	
	7495	1D47	INC HL				7691	1E0B	LD (8241), HL	; temp2
	7596	1D48	INC HL				7694	1E0E	LD HL, (8236)	
	7497	1D49	LD C, (HL)				7697	1E11	LD DE, 13430	
	7498	1D49	INC HL				7700	1E14	LDIR	
	7499	1D4B	LD B, (HL)				7702	1E16	LD (8236), HL	; start
	7500	1D4C	POP HL				7705	1E19	CALL 7509	
	7501	1D4D	INC BC				7708	1E1C	LD HL, (8241)	; temp2
	7502	1D4E	INC BC				7711	1E1F	LD A, H	
	7503	1D4F	INC BC				7712	1E20	OR L	
	7504	1D50	INC BC				7713	1E21	JR NZ, 7650	
	7505	1D51	CALL 192	; shrink			7715	1E23	XOR A	
	7508	1D54	RET				7716	1E24	LD (8248), A	
	7509	1D55	LD HL, (23627)	; vars			7719	1E27	JP 7601	
	7512	1D58	INC HL				7722	1E2A	LD BC, (8241)	; temp2
	7513	1D55	LD (23627), HL	; vars			7726	1E2E	LD HL, (8236)	
	7516	1D5C	CALL 7447				7729	1E31	LD DE, 13430	
	7519	1D5F	LD HL, 13430				7732	1E34	LDIR	
	7522	1D62	INC HL				7734	1E36	LD (8238), DE	
	7523	1D63	INC HL				7738	1E3A	LD A, 50	; 2
	7524	1D64	LD C, (HL)				7740	1E3C	LD (8248), A	
	7525	1D65	INC HL				7743	1E3F	JP 7601	
	7526	1D66	LD B, (HL)				7746	1E42	LD HL, 8328	
	7527	1D67	LD HL, (15990)				7749	1E42	LD DE, (8238)	; datablock
	7530	1D6A	INC BC				7753	1E49	LD BC, 2000	
	7531	1D6B	INC BC				7756	1E4C	LDIR	
	7532	1D6C	INC BC				7758	1E4E	XOR A	
	7533	1D6D	INC BC				7759	1E4F	LD (8248), A	
	7534	1D6E	PUSH BC				7762	1E52	LD HL, 13430	
	7535	1D6F	PUSH BC				7765	1E55	LD A, (HL)	
	7536	1D70	XOR A				7766	1E56	CP 40	; (
	7537	1D71	OUT 84, A				7768	1E58	JR NC, 7815	
	7539	1D73	CALL 189	; ld#1			7770	1E5A	INC HL	
	7542	1D76	LD A, 8				7771	1E5B	INC HL	
	7544	1D78	OUT 84, A				7772	1E5C	LD C, (HL)	
	7546	1D7A	POP BC				7773	1E5D	INC HL	
	7547	1D7B	LD HL, 13430				7774	1E5E	LD B, (HL)	

name	Dec	Hex	Instr-Dec	Remarks	(21)
	7775	1E5F	INC BC		
	7776	1E60	INC BC		
	7777	1E61	INC BC		
	7778	1E62	INC BC		
	7779	1E63	LD A, B		
	7780	1E64	CP 5		
	7782	1E66	JR C, 7786		
	7784	1E69	RST 8 ERR 16	; no room for line	
	7786	1E6A	PUSH BC		
	7787	1E6B	POP HL		
	7788	1E6C	LD BC, (8241)	; temp2	
	7792	1E70	OR A		
	7793	1E71	SBC HL, BC		
	7795	1E73	PUSH HL		
	7796	1E74	POP BC		
	7797	1E75	LD HL, 8328	; datablock	
	7800	1E73	ADD HL, BC		
	7801	1E79	LD (8236), HL	; start	
	7804	1E7C	LD HL, 5090	; length of track	
	7807	1E7F	OR A		
	7808	1E80	SBC HL, BC		
	7810	1E82	LD (8241), HL	; temp2	
	7813	1E85	JR 7705		
	7815	1E87	LD A, (23622)	; ppc(hi)	
	7818	1E8A	CP 255	; end of track map	
	7822	1E8C	JP NZ, 4499		
	7823	1E8F	LD HL, (8321)	; linenu	
	7826	1E92	LD (23618), HL	; newppc	
	7829	1E95	LD A, H		
	7830	1E96	CP 255	; end of track map	
	7832	1E98	JR Z, 7835		
	7834	1E9A	XOR A		
	7835	1E9B	LD (23620), A	; nsppc	
	7838	1E9E	JP 4499		
	7841	1EA1	PUSH AF		
	7842	1EA2	LD A, 127		
	7844	1EA4	IN A, 254		
	7846	1EA6	CP 30	; is it "break"	
	7848	1EA8	JR Z, 7852		
	7850	1EAA	POP AF		
	7851	1EAB	RET		
	7852	1EAC	RST 8 ERR 13	; break - cont repeats	
	7854	1EAE	LD IX, 16090	; mwide	
	7858	1EB2	LD A, (IX+3)	; ppas	
	7861	1EB5	LD B, A		
	7862	1EB6	AND A		
	7863	1EB7	JR NZ, 7952	; tokens	
	7865	1EB9	POP AF		
	7866	1EBA	CP 13	; enter?	
	7868	1EBC	CALL Z, 8016		
	7871	1EBF	CP 6	; print comma?	
	7873	1EC1	JP Z, 8056		
	7876	1EC4	CP 10	; cursor down?	
	7878	1EC6	JR Z, 7925		
	7880	1EC8	CP 12	; delete?	
	7882	1ECA	JR Z, 7941		
	7884	1ECC	CP 16	; ink?	
	7886	1ECE	JR C, 7928		
	7888	1ED0	CP 23	; tab?	
	7890	1ED2	JR C, 7928		
	7892	1ED4	JP Z, 8050		
	7895	1ED7	CP 32	; space?	
	7897	1ED9	JR C, 7928		
	7899	1EDB	CP 123		
	7901	1EDD	JR C, 7925		
	7903	1EDF	CP 128		
	7905	1EE1	JR C, 7941		
	7907	1EE3	CP 165		
	7909	1EE5	JP NC, 8075		
	7912	1EE8	LD B, 79		
	7914	1EEA	SUB B		
	7915	1EEB	CALL 8080		
	7918	1EEE	LD A, 8		
	7920	1EF0	CALL 8080		
	7923	1EF3	LD A, 95	; -	
	7925	1EF5	CALL 8001		
	7923	1EF8	POP BC		
	7929	1EF9	POP HL		
	7930	1EFA	JP 186	; jput	
	7933	1EFD	POP AF		
	7934	1EFE	JR 7925		
	7936	1F00	CALL 8080		
	7939	1F03	JR 7928		
	7941	1F05	PUSH AF		
	7942	1F06	LD A, (23632)	; chans(hi)	
	7945	1F09	CP 93	;]	
	7947	1F0B	JR C, 7933		
	7949	1F0D	POP AF		
	7950	1F0E	JR 8075		
TOKENS	7952	1F10	POP AF		
	7953	1F11	BIT 5, B		
	7955	1F13	JR NZ, 7936		
	7957	1F15	BIT 7, B		
	7959	1F17	JR NZ, 7928		
	7961	1F19	BIT 6, B		
	7963	1F1B	JR NZ, 7974		
	7965	1F1D	DEC (IX+3)	; lfeed	
	7968	1F20	JR 7928		
	7970	1F22	LD A, 5		
	7972	1F24	JR 7996		
	7974	1F26	LD B, A		

name	Dec	Hex	Instr-Dec	Remarks
	7975	1F27	LD A, (IX+0)	; width
	7978	1F2A	CP B	
	7979	1F2B	JR NC, 7988	
	7981	1F2D	LD B, 0	
	7983	1F2F	PUSH BC	
	7984	1F30	CALL 8069	
	7987	1F33	POP BC	
	7988	1F34	LD A, (IX+1)	
	7991	1F37	CP B	
	7992	1F38	JR NZ, 7983	
	7994	1F3A	LD A, 1	
	7996	2F3C	LD (IX+3), A	; ppas
	7999	1F3F	JR 7928	
	8001	1F41	CALL 8080	
	8004	1F44	LD HL, 16090	; mwide
	8007	1F47	LD A, (HL)	
	8008	1F48	INC HL	
	8009	1F49	INC (HL)	
	8010	1F4A	LD C, (HL)	
	8011	1F4C	CP C	
	8012	1F4C	CALL C, 8016	; lf
	8015	1F4F	RET	
LF	8016	1F50	LD A, 13	; do a line feed
	8018	1F52	CALL 8080	
	8021	1F55	LD A, (IX+2)	; lfeed
	8024	1F58	CP 10	; lfeed + carr ret?
	8026	1F5A	JR NZ, 8031	
	8028	1F5C	CALL 8080	
	8031	1F5F	LD (IX+1), 0	; put 0 in lfeed
	8035	1F63	LD B, (IX+4)	; margin
	8038	1F66	XOR A	
	8039	1F67	CP B	
	8040	1F69	RET Z	
	8041	1F69	LD A, 32	; space
	8043	1F6B	CALL 8080	
	8046	1F6E	DJNZ 8041	; repeat until marg done
	8048	1F70	XOR A	
	8049	1F71	RET	
	8050	1F72	LD (IX+3), 64	
	8054	1F76	JR 7928	
	8056	1F78	CALL 8069	
	8059	1F7B	LD A, (IX+1)	
	8062	1F7E	AND 15	
	8064	1F80	JR NZ, 8056	
	8066	1F82	JP 7928	
	8069	1F85	LD A, 32	; space
	8071	1F87	CALL 8001	
	8074	1F8A	RET	
	8075	1F8B	POP BC	
	8076	1F8C	POP HL	
	8077	1F8D	JP 5990	
	8080	1F90	PUSH AF	
	8081	1F91	CALL 7841	; check for break
	8084	1F94	LD A, (IX+6)	; printer type
	8087	1F97	AND A	
	8089	1F98	JR Z, 8100	; 0=Aerco I/F
	8090	1F9A	DEC A	
	8091	1F9B	JR Z, 8129	; 1=Tasman I/F
	8093	1F9D	DEC A	
	8094	1F9E	JR Z, 8112	; 2=A&J I/F
	8096	1FA0	LD HL, (8216)	; ptdrv (user defined)
	8099	1FA3	JP (HL)	
Aerco	8100	1FA4	IN A, 127	; printer check
	8102	1FA6	BIT 4, A	
	8104	1FA8	JR NZ, 8081	; not ready
	8106	1FAA	POP AF	
	8107	1FAB	OUT 127, A	; print!
	8109	1FAD	IN A, 127	
	8111	1FAF	RET	
A&J	8112	1FB0	IN A, 65	
	8114	1FB2	AND 4	
	8116	1FB4	JR NZ, 8081	; not ready
	8118	1FB6	POP AF	
	8119	1FB7	OUT 66, A	; print!
	8121	1FB9	LD A, 4	
	8123	1FBB	OUT 65, A	
	8125	1FBD	XOR A	
	8126	1FBE	OUT 65, A	
	8128	1FC0	RET	
Tasman	8129	1FC1	IN A, 191	
	8131	1FC3	BIT 0, A	
	8133	1FC5	JR NZ, 8141	
	8135	1FC7	IN A, 251	; printer check
	8137	1FC9	BIT 0, A	
	8139	1FCB	JR NZ, 8081	; not ready
	8141	1FCD	XOR A	
	8142	1FCE	OUT 251, A	; motor on
	8144	1FD0	DEC A	
	8145	1FD1	OUT 123 A	
	8147	1FD3	OUT 251, A	
	8149	1FD5	POP AF	
	8150	1FD6	OUT 123 A	
	8152	1FD8	LD A, 247	
	8154	1FDA	OUT 251, A	
	8156	1FDC	LD A, 255	
	8158	1FDE	OUT 251, A	
	8160	1FE0	RET	
	8161	1FE1	NOP	
			thru	all NOP's
	8191	1FFF	NOP	

Playing with disks.

After reading Bryan Davies' remarks about disk formatting using an ED drive and Gold Card in his trouble shooter column in the May '93 issue of QL World I decided to play about a bit myself. As I explored I found myself writing a number of little SuperBASIC routines to help me, and copies of these are being sent to Hugh for inclusion in the Toronto User Group QL Library.

In my language, a "routine" is a short piece of rough and ready code designed for a specific job. It works, but to turn it into a "program" I would need a lot of work on things like pretty screens, foolproofing, user friendliness etc. You are supposed to get into EDIT (or ED) and adapt them to suit your own purposes and systems. I have tried to put most of the things that might need changing into the first PROCedure, Set_up, with REMs indicating the changes you might want to make. Since there is not much error trapping, (in fact I cannot work out how to reliably trap the no-disk-in-drive error for direct sector reading,) if you do create an error, or BREAK the program for any reason you may leave the disk OPEN, so enter "close" or "finish" at the keyboard before running the routine again. The routines are: disk_sector_reader, disk_PC_formatter, and disk_rename (two versions)

Bryan Davies mentioned drilling a HD hole in a DD disk, so I took a copy of each format to my workshop, selected a drill bit of about the same size of the square hole in the HD disk and made a round hole in the same place on my DD disk. Back to the old QL, and sure enough 2880 sectors appeared when I formatted it. He mentions an undocumented GG command "flp_density D" to impose DD format (H for HD and E for ED), when I tried this on my doctored disk my experience was the same as his: "format failed". "So", I thought, "the disk drive is detecting the hole in some way, perhaps it's shining a light through?" I put a bit of sticky paper over the top of the hole and still got 2880. Holding it up to the window I

could see that the paper wasn't totally opaque, so I tried filling the hole with a blob of "blu-tack" (An opaque blue substance a bit like sticky plasticene which is available in the UK and has all sorts of uses.) Result 1440: but then I thought, "there's another way to test for the presence of holes, and that's to try to poke a solid object through it. Perhaps it's poking its fingers up from the bottom." and sure enough the piece of paper stuck on the bottom also gave 1440 sectors. At this point reason got the better of empiricism and I realised that since the holes are near the outer edge of the disc case I could probably see the mechanism if I lifted up the drive's dust flap, and sure enough, there were three strategically placed brass pins: one on the left for the write protect tab, and two on the right for ED and HD disk testing. (According to a note in the latest International QL Report you can set the jumpers on your drive so that it doesn't physically test the disk, but believes what the software tells it, and FLP_DENSITY seems to work under these conditions.)

The next thing to try, obviously, was an ED hole in a DD disk. This worked almost through to the center giving me about 5900 sectors - how do I know where it failed? - you can hear when a cylinder fails to format since the regular sequence of roughly 1 sec "thunks" from the disk drive breaks up into a quicker pattern. The centre, where the sectors are physically shorter, is obviously the most challenging place for the magnetic medium.

It is of course possible that some other drive manufacturers use the optical method, but a bit of opaque paper on the bottom should cope with both systems. This is so that you can use the disk in it's original format if you don't trust the new one or if it doesn't format properly.

In practise, I wouldn't recommend formatting a DD disk as ED, you are pushing things a bit to close to the limit, and the result is likely to be unreliable. On the other hand I have

heard it said that the only difference between some manufacturers DD and HD disks is the hole, the label and the price - it's not worth their while to have two different grades of magnetic material. If this is correct the only problem I can see here is the danger of getting bits of plastic turnings inside the sleeve.

Disk Formats

While I am on the subject of disks I should add a bit about the three different formats of 3.5" disks available to Gold Card users. For the absolute beginner I should explain the terms as I understand them. Firstly we have the "tracks", these are a series of concentric circles on which the data are recorded magnetically. You cannot see them, the material is uniform, you depend upon your disk drive mechanism moving the heads to exactly the right place. According to the words on the side of my disk box there are 135 tracks per inch, and since there are a total of 80 of them on a side the total width used is only about 0.6 of an inch. Most 3.5" disks are double sided, two heads read the corresponding tracks on both side of the disks at the same time, this combination is called a "cylinder". Since computers generally start counting at zero rather than one the tracks are numbered from 0 to 79 from the outside in, and the sides are numbered 0 and 1.

Each track is divided into a number of sectors, and this is where the formats differ. In DD (double density) format each track is divided into 9 sectors, each capable of holding 512 bytes (or 0.5 kbyte - the same as a microdrive sector)) so the whole disk has $9*2*80 = 1440$ sectors (or 720 kbytes).

High density (HD) disks squeeze every thing up closer fitting 18 x 512 kbyte sectors into the same space, thus doubling the capacity. But this doesn't mean that you can have 1440 or 2880 files on each disk since in both these formats the sectors are assigned to files in blocks of three, generally interweaving them as 1-4-7, 2-5-8, 3-6-9 etc - but

other formats are possible, it's all to do with the time it takes to perform the saving operations. This means that if you save only one byte there are immediately 1536 fewer bytes available for other files. A few sectors in track zero are also reserved for disk information, such as format information and the disk directory. So you can have about 500 and 1000 small files on DD and HD disks respectively. Apart from the reserved information these formats are exactly the same as those used by IBM PCs, and you can in fact read the information from an IBM disk on your QL, although not in a very convenient form!

Disk_sector_reader, will enable you to explore the contents of a QL disk (& IBM - DD & HD only) sector by sector.

However, when Miracle went to ED (Extra high density) format they decided to squeeze every bit onto them that they could and adopted a non-IBM format. Each track is split into 10 sectors, and each sector now contains 2048 bytes, so when it tells you that it is formatted to 6400 sectors it is telling a lie - it's really only 1600, but since they are four times as big as the standard QL sector size the total amount of memory available is the same. Now, however, sectors are assigned to files individually, so you can put over 1500 small files onto one disk. Since disk rotation speed is the same whatever the format this means that the QL has to be able to read ED disks at over four times the speed of a DD disk. Hard disks are faster still.

As an exception to the rule of counting from zero, tracks are numbered from 1 to 9, 18 or 10 for DD, HD or ED respectively.

Formatting IBM disks on your QL

As I mentioned above, the magnetic format of QL and IBM DD and HD disks is the same. In each disk drive interface software there is a toolkit of commands and functions specifically directed to disk operations (these are sometimes thought of as part of Toolkit 2, but they are really quite separate.) One of these enables you to read and write disk

sectors directly, irrespective of the disk's filing system, and it is this that enables you to copy the vital sectors from a ready formatted IBM disk into your QL and then copy them back onto a preformatted QL disk, so that it will behave exactly as though it had been formatted on one of the original IBoMinables. Don't think that you will then be able to read your QL files off it with a PC though, you only get an empty formatted disk, any files on the disk will be lost. It also only produces a data disk, not a system disk.

I got the basic idea from a little routine that Simon Goodwin sent me, but I've modified it quite a bit. The program as listed is incomplete, it needs a set of data statements, but since it also contains a procedure to create these there's no need to worry. The first time you use the program as supplied you should have ram1_ available (if not use flp2_ or mdv1_ making sure that you have a disk or tape available and change all references to ram1_ to flp2_ or mdv1_ in the program.) Put an empty, but formatted PC disk into flp1_, and then enter "make_data" at the keyboard and follow instructions to create the necessary DATA statements in a file called ram1_data. (Make_data is a stand alone procedure - really a separate program since it is not invoked at all when you RUN the main program, it can only be called directly from the keyboard.) Then type "MERGE ram1_data" to add its lines to the program and save it in its expanded form.

It can then be used to change the format of a QL disk without further reference to a genuine IBM disk. I've done it this way so as not to risk infringing anyone's copyright.

How it works: 1. Make data.

The vital information on the format of the disk (QL or IBM) is found in sector 1, track 0, side 0; this can always be read whatever the format, otherwise we would be in a catch 22 situation. It is known as the boot bloc and is read into a string variable b\$ - all 512 bytes of it - in line 925 of the program. The first statement of this line is one of those disk toolkit specials. Instead of just

OPENing a specific file on the disk it opens the whole disk at once - a dangerous procedure if you aren't sure what you are doing! The parameter, which must be in quotes, opens the disk on flp1 as a double density (second d) disk with 512 kbyte sectors (the 2) - you need to know this in advance - but all QL and IBM DD disks use this format. When a disk is OPENed like this all you can do is to read and write (that's what makes it dangerous) whole sectors, each individual sector is identified by a number (the sector id) made up as follows: (sector + 256*side + 65536*track). Since the first side and the first track are both numbered 0, 1 always refers to the boot bloc sector. The GET function is next used to read the sector into b\$; the syntax is:

```
GET[channel number]\sector_id,string_variable
```

only the channel number is optional, but you will almost always need to put it in, since it is very unlikely that you will ever use the command with the default channel, #1! The effect then in line 925 is to dump all 512 bytes of the boot bloc sector from the sector into b\$. Since that is all that is required from the disk it is immediately closed again.

The next line, 930, OPENS a conventional file in a ram disk (you could use flp2 or mdv1 here remember), dumps the sector bloc from b\$ into that, and closes it again. It is immediately reOPENed in line 935, but for input only, and another new file, ram1_data, created. The loop from 940 to 965 copies the 512 bytes from the boot bloc into ram1_data, changing the format as it goes. It first prints a line number and the word DATA to the file (945), takes the bytes one at a time from channel 3 (using BGET) as integers this time not characters (b% - the % sign declares the variable an integer) and prints them to channel 4 followed by a comma except for the last one (960). The program has automatically written a line of BASIC for you: a DATA statement with sixteen numbers. This is repeated 15 more times, and everything CLOSED (970).

(Those readers with a bit of programming experience will have asked themselves why I used the intermediate file `ram1_boot_bloc` rather than just read the characters back out of `b$` and convert them to integers using coercion. I wondered that too later; but I left it as it is since it illustrates more of the file handling commands.)

You merge this set of DATA statements in to complete the program and you can save it for future use. The line numbers for the DATA statements are carefully chosen so as not to conflict with other lines in the program, so don't RENUMBER until it is complete. (I couldn't make the MERGEing automatic since MERGE will not work from within a PROCEDURE)

How it works: 2. The Formatter

This is basically `make_data` in reverse. The integers are READ back from the DATA lines into `b$` as the corresponding characters (180) and PUT back into the first sector. (PUT is just like GET except that the information goes the other way). When I examined the rest of the sectors in the first cylinder of a freshly IBM formatted, virgin disk I found that most of them were filled with zeros, (byte 0 not character 0) except that 2 and 5 had 249,255,255 as their first three bytes, so lines 240 to 280 duplicate this on the new disk, this is also true of a preformatted IBM disk I have checked. Simon's routine did something different here which didn't work for me, so I suspect there may be other formats around. If this routine doesn't work for you have a look at the contents of these sectors on a newly formatted virgin disk, and duplicate them (`disk_sector_reader` will help here.) I have no idea what any any of these codes do, I'm just a copycat. Once again, `disk_sector_reader` will help if you want to explore the details.

Changing the name of a QL disk.

If you alter the above program to print out `b$` (or use "`disk_sector_reader`") you will find that the name of a QL format

disk occurs fairly early along the string - actually in bytes 5 to 15 (counting from zero remember!). As I wrote these notes I was reminded of a routine I had once written to change the name on a disk see "`disk_name`". As it stands it will only cope with a DD disk in `flp1_`, but it was easily extended to deal with the other formats and as many drives as you have. This extended version is also being sent to the QL Library, and should be available from Hugh. If you have understood my explanations of "`PC_formatter`" you should be able to work out how these work for yourself, the principle is very similar, it just reads the boot bloc, alters it a bit, and then writes the modified version back onto the disk.

Disk sector reader

This is probably too long to be printed in the magazine. It will deal with all three QL formats and the two compatible PC ones (DD and HD). It first asks for the drive and disk format (on your own head be it if you give wrong information here!) and then which track(s) you want to examine. It then dumps the contents of each sector in order onto the screen, if you want to examine it in more detail press "D" and it will display the sector byte by byte giving the byte's position in the sector, its normal character (if printable) and its ASCII code. As supplied it is designed to work on a TV, if you use a monitor and want to see more at once alter the values of "`cols`" and "`batch`" to suit your set up in PROCEDURE `Set_up`. (Batch should divide nicely into 512.) This program may work even on a partially corrupted disk and could be adapted to act as a file rescue program.

Howard Clase
Tel (709) 753-6415
403 Torbay Rd SS#3
St John's
Nfld A1A 5C9
e-mail hclase@morgan.ucs.mun.ca

DISK PC FORMATTER

```

100 nm$=          "Disk_PC_formatter"
110 REMark SG modified 1992.07.22 hjc
120 REMark ~~~~~Main
130 Set_up: Do_it: Finish: STOP
140 REMark ~~~~~æ
150 DEFine PROCedure          Set_up
155 CLS:b$="": RESTORE
160 PRINT "This program converts a QL format disk to\"PC format, by
inserting the boot bloc for\"a PC disk to track 0, and initialising
an\"empty PC directory.  DD disks only!"
180 FOR i=0 TO 511: READ b%: b$=b$&CHR$(b%)
190 END DEFine
200 REMark ~~~~~æ
210 DEFine PROCedure          Do_it
220 PRINT\"Insert a formatted QL disc & press any key\": PAUSE(-1)
230 OPEN#3,\"flp1_*d2d\": PUT#3\1,b$
240 c$=CHR$(249)&CHR$(255)&CHR$(255)
250 c$=c$&FILL$(CHR$(0),509)
260 FOR s=2,5: PUT#3\s,c$
270 FOR s= 3,4, 6TO 9,257TO 261
280  PUT#3\s,FILL$(CHR$(0),512): END FOR s
290 END DEFine
300 REMark ~~~~~æ
310 DEFine PROCedure          Finish
320 CLOSE#3: PRINT\"Conversion complete"
330 END DEFine
340 REMark ~~~~~DATAæ
900 REMark ~~~~~
910 DEFine PROCedure          make_data
920 LOCAL b$,i,j: CLS
930 PRINT"Put a formatted PC disk into flp1_,\"then press any key."
940 PAUSE(-1)
950 OPEN#3,'flp1_*d2d': GET#3\1,b$: CLOSE
960 OPEN_NEW#3,raml_boot_bloc: PRINT#3,b$;:CLOSE
970 OPEN_IN#3,raml_boot_bloc:OPEN_NEW#4,raml_data
980 FOR i=0 TO 511 STEP 16
990  PRINT#4,350+i!'DATA ';
1000  FOR j=0 TO 15
1010    BGET#3,b%: PRINT#4,b%;
1020    IF j<>15: PRINT#4,', ';
1030  END FOR j:PRINT#4:END FOR i
1040 CLOSE
1050 PRINT"Now enter 'MERGE raml_data' at the \"keyboard, and resave the
program."
1060 END DEFine

```

DISK SECTOR READER

```

5 nm$= "Disk_sector_reader"
10 REMark PD H.J. Clase 1993.09.09 ver1.3
15 REMark Displays contents of disk sectors
20 REMark of QL (DD,HD&ED) and MSDOS (DD&HD only)
25 REMark disks.
30 REMark N.B. Requires SuperToolkit II
35 Set_up: Read_sectors: Finish
100 REMark ~~~~~
105 DEFine PROCedure Set_up
110 CLS: FLP_SEC 2
115 pd$="1,2": REMark Numbers of drives available
120 pf$="D,H,E": REMark possible formats
125 dev$=Get_flp$: fmt$=Get_fmt$
130 Get_tracks Ftrack,Ltrack
135 REMark For "detailed" display, TV values
140 REMark for wider screens try 6 & 64
145 cols=3: batch = 32 :REMark <--- <--- <--- <---
150 END DEFine
200 REMark ~~~~~
205 DEFine PROCedure Read_sectors
210 LOCAL a$,k$,si,se,tr
215 ch%=FOPEN(dev$&fmt$)
220 FOR tr=Ftrack TO Ltrack
225 FOR si=0,1
230 FOR se = 1 TO maxsec
235 GET#ch%\ (se+256*si+2^16*tr),a$
240 PRINT"Track"!tr!"side"!si!"sector"!se\a$
245 PRINT\\"Press D for detailed display"
250 PRINT"or any other key to continue"\\
255 IF INKEY$(5000)="d": Details
260 END FOR se:END FOR si: END FOR tr
265 END DEFine
300 REMark ~~~~~
305 DEFine PROCedure Finish
310 CLOSE#ch%
315 END DEFine
400 REMark ~~~~~
405 DEFine FuNction Get_flp$
410 LOCAL d$,n$,lp,p$: d$="flp1_": p$="("&pd$&")"
415 REPEAT lp
420 PRINT"Which drive is the disk in? ":p$,
425 n$=INKEY$(-1): PRINT "flp";n$
430 IF n$ INSTR(pd$) AND n$<>"," : Pip: EXIT lp
435 PRINT "Drive ";n$;" not available.":Bip
440 END REPEAT lp
445 d$(4)=n$: RETURN d$: END DEFine
500 REMark ~~~~~
505 DEFine FuNction Get_fmt$
510 LOCAL f$,p$,r$: p$="("&pf$&")"
515 REPEAT lp
520 PRINT"Disk format? ";p$,,,
525 f$=INKEY$(-1):PRINT f$&"d"
530 IF f$ INSTR(pf$) AND f$<>"," : Pip: EXIT lp
535 PRINT "Format ";f$;"d unknown.":Bip

```

```

540 END REPEAT lp
545 f = CODE(f$): SELECT ON f
550   = 68,100: r$="*D2D": maxsec=9
555   = 69,101: r$="*D4E": maxsec=18
560   = 72,104: r$="*D2H": maxsec=10
565 END SELECT : RETURN r$: END DEFINE
600 REMARK ~~~~~
605 DEFINE PROCEDURE          Get_tracks(Ft,Lt)
610 LOCAL lp
615 REPEAT lp
620   INPUT"First track (0-79)",,,Ft
625   IF Ft<80 AND Ft>=0: Pip: EXIT lp
630   PRINT "0-79 only, try again!": Bip
635 END REPEAT lp
640 REPEAT lp
645   INPUT"Last track ("&Ft&"-79)",,,Lt
650   IF Lt<80 AND Lt>=Ft: Pip: EXIT lp
655   PRINT Ft;"-79 only, try again!": Bip
660 END REPEAT lp: END DEFINE
700 REMARK ~~~~~
705 DEFINE PROCEDURE Pip: BEEP 100,30: END DEFINE
710 DEFINE PROCEDURE Bip: BEEP 4000,40: END DEFINE
800 REMARK ~~~~~
805 DEFINE PROCEDURE          Details
810 LOCAL c$,i,l$,mb,n%: l$=CHR$(10): mb=256: n%=1
815 IF maxsec=10: mb=2048
820   PRINT"Position, character, code"\\
825 FOR i=1 TO mb
830   f$='### # ### '
835   c$=a$(i): IF c$=l$:c$=" ": REMARK LF trap
840   PRINT_USING f$,i,c$,CODE(a$(i))
845   IF NOT n% MOD cols: PRINT
850   IF NOT n% MOD 32
855     PRINT\\"Press any key when ready."
860     PRINT\\"Position, character, code"\\
865     PAUSE(2000)
870   END IF :n%=n%+1
875 END FOR i: PRINT \\": END DEFINE

```

DISK RENAME

```

5 nm$=          "Disk_rename"
10 REMARK h.j.c. 1993.6.25 version 0.9
15 REMARK N.B. Uses TK II
20 Rename_disk: Finish: STOP
100 REMARK ~~~~~
105 DEFINE PROCEDURE          Rename_disk
110 CLS: FLP_SEC 2: a$="": n$=""
115 OPEN#3,'f1p1_*d2d': GET#3\1,a$
120 PRINT "Old name = ":a$(5 TO 14)
125 INPUT "New name (ENTER to keep old) = "!n$
130 IF n$<>"": a$(5 TO 14) = n$: PUT#3\1,a$
135 END DEFINE
200 REMARK ~~~~~
205 DEFINE PROCEDURE          Finish
210 CLOSE: WHEN ERROR :REPORT
215 END DEFINE

```

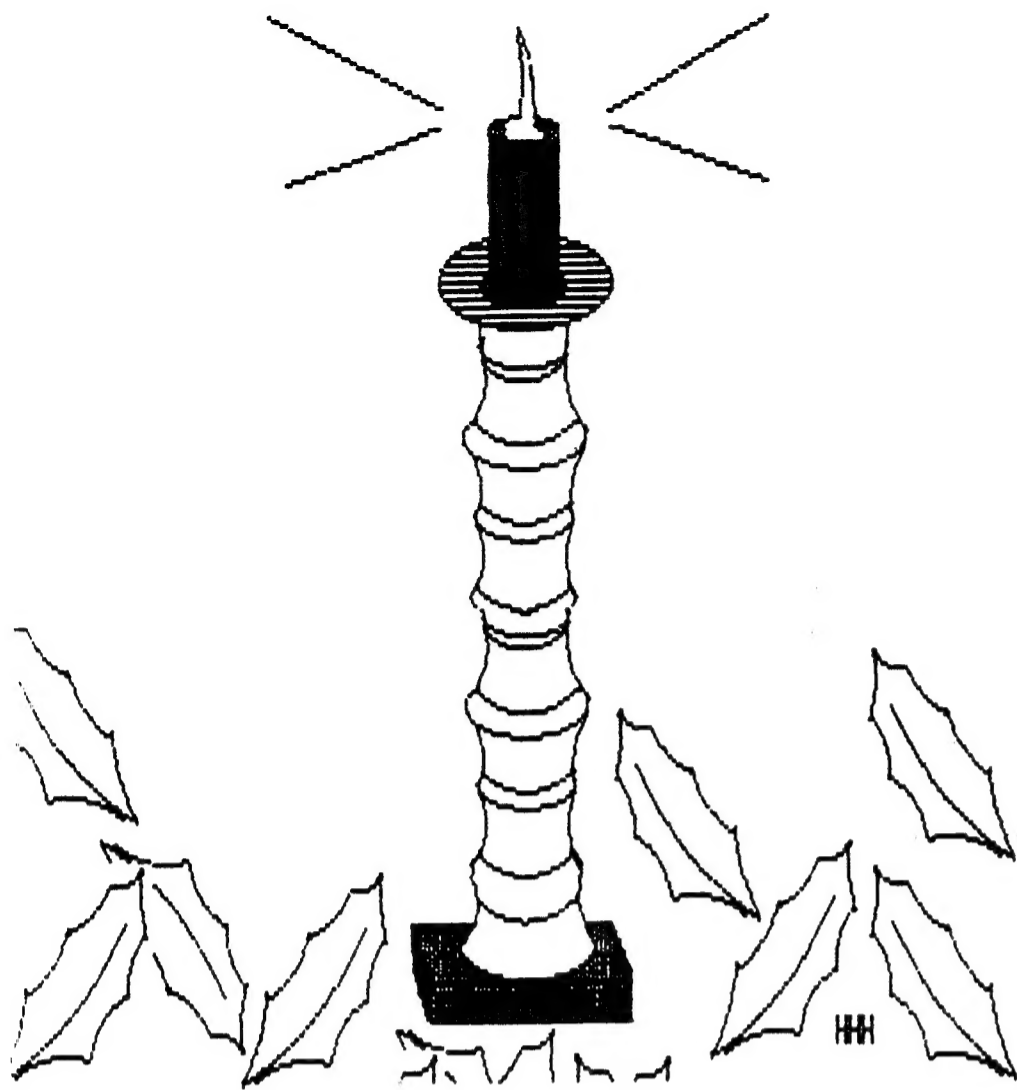
DISK RENAME2

```

5 nm$= "Disk_rename2"
10 REMark h.j.c. 1993.8.18 version 2.0
15 REMark N.B. Requires TK II
20 Set_up: Rename_disk: Finish: STOP
100 REMark ~~~~~
105 DEFine PROCedure Set_up
110 CLS: FLP_SEC 2
115 pd$="1,2": REMark Numbers of drives available
120 pf$="D,H,E": REMark possible formats
125 dev$=Get_flp$: fmt$=Get_fmt$
130 END DEFine
200 REMark ~~~~~
205 DEFine PROCedure Rename_disk
210 LOCAL a$,n$: a$="": n$=""
215 ch%=FOPEN(dev$&fmt$)
220 GET#ch%\1,a$
225 PRINT"\Old name = ";a$(5 TO 14)
230 INPUT "New name (ENTER to keep old) = "!n$
235 IF n$<>"": a$(5 TO 14) = n$: PUT#3\1,a$
240 END DEFine
300 REMark ~~~~~
305 DEFine PROCedure Finish
310 CLOSE#ch%
315 END DEFine
400 REMark ~~~~~
405 DEFine FuNction Get_flp$
410 LOCAL d$,n$,lp,p$: d$="flp1_": p$="("&pd$&")"
415 REPEAT lp
420 PRINT"Which drive is the disk in? ";p$,
425 n$=INKEY$(-1): PRINT "flp";n$
430 IF n$ INSTR(pd$) AND n$<>"": Pip: EXIT lp
435 PRINT "Drive ";n$;" not available."
440 BEEP 4000,30: END REPEAT lp
445 d$(4)=n$: RETURN d$: END DEFine
500 REMark ~~~~~
505 DEFine FuNction Get_fmt$
510 LOCAL f,f$,p$,r$: p$="("&pf$&")"
515 REPEAT lp
520 PRINT"Disk format? ";p$,,,
525 f$=INKEY$(-1):PRINT f$&"d"
530 IF f$ INSTR(pf$) AND f$<>"": Pip: EXIT lp
535 PRINT "Format ";f$;"d unknown."
540 BEEP 4000,30: END REPEAT lp
545 f = CODE(f$): SELEct ON f
550 = 68,100: r$="*D2D"
555 = 69,101: r$="*D4E"
560 = 72,104: r$="*D2H"
565 END SELEct : RETURN r$: END DEFine
600 REMark ~~~~~
605 DEFine PROCedure Pip: BEEP 100,30:END DEFine

```

Howard Clase, Tel (709) 753-6415, 403 Torbay Rd SS#3. St John's, Nfld
A1A.5C9 e-mail hclase@morgan.ucs.mun.ca



No man,
when he hath lighted a
candle,
putteth it in a
secret place,
neither under a bushel,
but on a candlestick,
that they which come in
may see the light.

The Northern Light of Sinclair Computing
bring you
SEASONS GREETINGS
and wish one and all
A MERRY CHRISTMAS
and
A HAPPY AND PROSPEROUS NEW YEAR

TORONTO TIMEX-SINCLAIR USERS CLUB

NOV/DEC 1993

Dec 24, 1993

Dear Out-of-Town Members,

Another year has just about passed and TTSUC is still here. Jeff Taylor has some comments in his editorial, which you might take to heart.

Larry Crawford, one of our members from London, Ontario asks, in a letter I just received, why not put out a list of members names & addresses (possibly phone numbers, why not?), so that members could communicate directly with each other as interests them. I think this is a good idea. Possibly at the same time I could write a notation as to their main Timex computer interest. How about this idea. If anyone does not want to be on the list they could drop me a line, requesting deletion.

I have spoken only to Jeff Taylor about this. He is favourable to the idea. We will have to consider it at our next meeting; maybe there's an aspect that I have not thought of. If it is a go, I suggest that the next newsletter may bring the information.

Our 2068 Larken library has had some additions which you Larken owners might be interested in. Firstly, Larry Crawford has been working Tasword over considerably. Incorporating many club members suggestions into it. Well, it can cope with I/O's as varied as tape/LKDOS/OLIGER. And there's a host of other refinements. Some of you may have an earlier vintage of Larry's work. Ask for a revised Larken library Disk #42.

Larry has also been refining his Interbank Database program on Larken library disk #30. The demonstration on the library disk contains the Sinc-Link Index. You need a Larken RAMDISK to make use of it. If you already have an earlier version, do get the updated copy.

Larry C. has really been busy! He has also reworked an earlier version of his AROS utility. This program also makes use of the Larken RAMDISK. Using this utility allows you to store several programs, one on each of seven 32K chips of the RAMDISK. The utility allows you to bankswitch any desired program into the 2068, from the RAMSISK chips. Ask for it on library disk #56.

Wow, our Larken 2068 library is up to disk #57. Let's describe some of them. Our printed catalogue is up to Disk #50. We'll start from there:

- #51...MSCRIPIT SUPPORT PROGRAMS
- #52...TIMEX INFORMATION FILES
- #53...MUSIC FORM SPECTRUM GAMES TAPES
- #54...SPECTRUM/TS2068 ROM CONVERSION
- #55...STEPHEN GUNHOUSE COLLECTION
- #56...AROS DEMONSTRATION

Is anyone waiting for anything club-wise, from me? If so, I've overlooked you. Sorry about that. write and remind me.

Shall close wishing you the very best of the New Year.

Sincerely

George Chambers.

