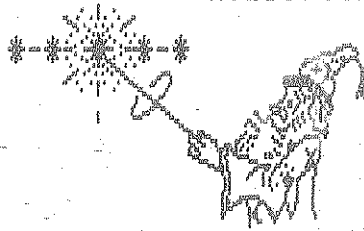


PORT FE

SORCERERS USERS' GROUP
(Toronto)

P.O. Box 1173 Sta. 'B'
Downsview, Ontario,
Canada. M3H 5V6

SORCERER Newsletter



The Toronto Sorcerer User Group was founded in the Spring of 1978, a handful willing and eager to learn members.

This newsletter shall at all times keep in mind the goal of its conception. To spread the seeds of knowledge.

Articles printed in this newsletter shall be free for all Sorcerer Users' groups to reprint or comment on as they see fit.

Articles submitted for this newsletter must be in no later than the beginning of the month of every month.

May 1981 ISSUE

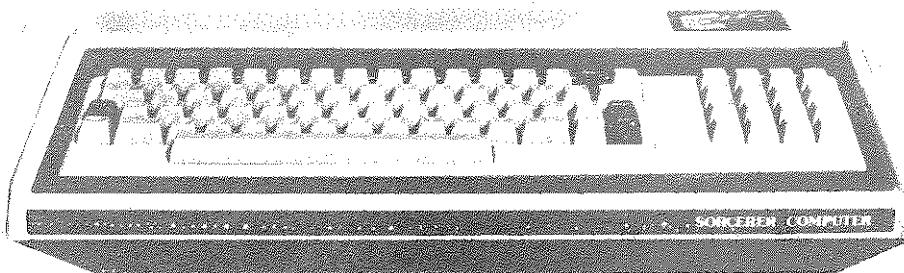
TABLE OF CONTENTS

1. - TRS80 Level Two Basic (for the Sorcerer ?!#@...)
2. - Editor's Note
- Program Review - SNAKE
3. CP/M SECTION:
- QUARTER SQUARE PLOTTING
4. - EXBASIC PLOTTING PROGRAM LISTING
- MAKML.BAS, PLOT.BAS
7. - WORDSTAR (continued)
- Wordstar command listings
- Wordstar DOT commands
- NOTICE TO MEMBERS
9. SORCERER TECHNICAL TIPS
- 48k RAM ADDED to the Sorcerer I (update)
10. - MEMBERSHIP APPLICATION FORM

NEW MEETING PLACE

Location : Bathurst Heights Library - Date: Thurs. May 14 - 7:00 PM
3170 Bathurst St.

One block north of Lawrence on the west side of Bathurst.



The Exidy 19K Extended Cassette Basic described in the last two newsletters now has a stiff competitor.

In comparison to the Sorcerer Standard Basic, the newly converted R/S (Microsoft) 16K Extended Basic is a superb piece of software. It is almost exactly the same interpreter as the 19K Exidy version but chopped down to 12K, although my extended commands close the gap to 16K.

If you have a 32K Sorcerer and use cassette as main storage, this is a dream come true for any basic programmer that has been slaving away at the old ROMPAC Basic. Machine language programs are not quite compatible because of the memory mapped I.O. and therefore require changing.

The following are the new commands found in the extended basic in addition to the regular ones.

AUTO LINE #, increment	CLOAD?"filename" -R/S format tape verify
DELETE LINE # - LINE #	LIST LINE # - LINE #
EDIT LINE # + 16 sub comds	TRON & TROFF - Trace functions
SYSTEM - Monitor mode	PRINT @ Position, item list
PRINT USING String, item list	
PRINT # -1 (or 2) - I/O of variables & strings to cassette at	
INPUT # -1 (or 2) 1200 baud Sorcerer format.	
DEFINT letter range	DEFSNG letter range
DEFDBL letter range	DEFSTR letter range
ERROR code#	ON ERROR GOTO line#
RESUME line#	ELSE statement or line#
INKEY\$	STRING\$ (n, "character" or #)
COBL(x)	CINT(x)
CSNG(x)	FIX(x)
RANDOM	RND(x)
SET(x,y)-turn on pixel	RESET (x,y)-turn off pixel
CLS	POINT (x,y)
ERL	MEM
VARPTR variable name	LLIST - list or print to
USR(x)	LPRINT - centronics printer

Along with these commands, come five new error codes:

L3 - Disk basic only	NR - No resume
FD - Bad file(input #-1, etc.)	UE - Unprintable error
RW - Resume without error.	

One of the better features is the extension of variable types and format:

- A% - Integer in the range -32767 to 32767
- A! - Single precision of 6 digit accuracy
- A# - Double precision of 16 digit accuracy
- A\$ - Variable strings of 255 characters

In comparison to the 19K Exidy Cassette Extended Basic, a number of features had to be squeezed out. Listed are some of the commands which I did not incorporate into the new R/S Basic. You will now know what you will be missing.

WHILE-WEND / INSTALL / DEF FN / OCT\$(X) / DEF USR / SPACE\$(X) / BAUD / ERASE
 LINE INPUT / NULL / OPTION BASE / SWAP / SERIAL / WAIT / WIDTH / HEX\$(X)
 AND AND

- Doesn't have 34 character variable names, only limited to two (2) letters
- PRINT @ position, item = CURSOR X,Y
- R/S Basic can have lines of maximum 255 characters in length.

RADIO SHACK BASIC INTERPRETER CHANGES

In order for any R/S program to be run on the Sorcerer, the interpreter has been MODIFIED extensively and new commands added to simulate R/S memory mapped hardware. A number of utility commands are also included to make I/O to the outside world easier, such as the Paddle input.

We shall continue this article in the next issue of PORT FE giving the interpreter changes and the variations to the commands, joystick input circuitry, cassette interface and changes in hardware etc...

Article by: Robert Lansdale (SUGT)

EDITOR'S NOTE

The TRS80 LEVEL II EMULATOR as described in our January issue of PORT FE is the level II Microsoft Basic which runs on the Radio Shack computer. This was received from Devin Trussell of the Sorcerer Computer Users Group (Australia).

The raw program has been greatly modified by Robert Lansdale, one of our Toronto members. He has been working with the TRS80 interface for approximately one and a half years, and has finally completed the work. I must congratulate him for the fine work he has done in this program.

I would also like to congratulate whomever is responsible for starting this piece of work in the first place. It really does help to have contact with other Sorcerer Users Groups around the world. This is an example of what co-operation can do between us, if one doesn't have the answers to a problem or is/has been working with a program too long, a fresh look at the problem is sometimes all that is necessary.

I wish to state further for those that have this program that they cannot SELL it. This would be a copyright infringement of Microsoft Basic. The program is still theirs regardless of the I/O changes and the additional commands that were added.

PROGRAM REVIEW - SNAKE (Basic)

The snake slowly creeps along. He is hungry. Suddenly, he spies a tasty 9. He slithers closer, closes in for the chomp. CHOMP!!! OH NO. He missed! He comes around again, carefully manoeuvring as not to take a chomp of his own long slender body. He is almost there. As he closes in for the final chomp, the digit disappears! OH NO! - But wait, another digit has appeared on his right. The snake once again tries to close in...

This was the story of the game of Snake. The game is a shoot off of the familiar blockade type game where the object is to try not to run into the walls or yourself. In this game of snake, you control a hungry Snake. The object of the game is to eat as many numbers as you can without running into yourself or the four walls. Only one digit appears at a time. If you can't get to that digit in time, it disappears and another appears somewhere else on the screen. Although this may sound rather simple, it is not. For one thing, the more digits you eat, the longer the snake gets. This makes it harder for you to survive and keep on eating more and more digits.

Snake appears to be a well written Basic game. My only criticism of the game is that the game often drags on too long for the better player and that it is too slow. Ahh, but oh well, Basic always is...

By: Paul Tan

Quarter Square Plotting on the Sorcerer.

One problem with the Sorcerer is that it is impossible to get the full resolution out of the machine because there are a lack of user-defined graphic characters. One way to get out of this problem is to settle for less resolution, which does not take up so many characters. This is the approach that I have implemented for any one of Exidy's extended versions of BASIC. When I refer to EXBASIC the same will hold true for EXCAS except that it does not have any disk features. MBASIC 5.03 will also work, but it does not have the cursor function.

By using quarter square graphic like the one on the X key, it is possible to double the standard resolution of the Sorcerer. You are able to plot 128 by 60 points and never have to worry about running out of graphics. This is the same approach that is taken on many of the home computers, eg. TRS-80 and the PET.

One might say that the resolution is limiting but if there are so many good things out for the TRS-80 we may as well use the stuff there without redoing the wheel. (eg. SUBLOGIC flight simulator). There are 4 routines in the standard graphic pack. They are ginit, set, reset, and plot.

Ginit

Parameters: None

Example: CALL GINIT

This routine initializes the user defined graphics needed for the routines. (Standard address: &h9b00)

Plot

Parameters: x,y,colour

Example: CALL (X,Y,COLOUR)

This routine will plot a point on the screen at the given x,y coordinate. If the colour is black (0), then the routine will plot a black quarter square. If the colour is white it will plot a white quarter square. Note that no error checking is done to see if the coordinate is in range. If the coordinate is far enough out of range, you could crash your system. (Standard address: &h9b2b)

Set

Parameters: x,y

Example: CALL SET(X,Y)

This routine will plot a white pixel at x,y. It works in the same way as plot. (Standard address: &h9b0c)

Reset

Parameters: x,y

Example: CALL RST(X,Y)

This routine will plot a black square at x,y. It is the complement of the set function. Since reset is a reserved word in EXBASIC (Disk Version), I usually use RST for reset. (Standard address: &h9b21)

Table Of Standard Addresses for V1.00 of Graph Pac

Function	Address	Purpose
ginit	&h9b00	Set up user defined graphics
plot	&h9b2b	Plot a point at x,y with colour
set	&h9b0c	Plot a white point at x,y
reset (rst)	&h9b21	Plot a black point at x,y

Note: &h means the following digits are hexadecimal numbers.
This is the way that hex is used in EXBASIC.

There are 3 BASIC program and 1 file containing the source to the plotting routines. These files are:-

PLOT.BAS	BASIC load module of the plot routines
PLOTDEMO.BAS	BASIC program that demonstrates the plotting.
PLOT.MAC	Source code for the plotting routines. These require M80 to assemble.
MAKML.BAS	BASIC program that will make a file like PLOT.BAS above.

Using the routines with EXBASIC or the likes is easy. First the graphic routines must be put in memory. This is done with

```
EXBASIC PLOT /m:&h9aff
```

The /m:&h9aff tells EXBASIC that it can only use up to this address for programs, and sets aside a small section for the code.

PLOT.BAS is a program that will poke the machine language program in automatically when it is run.

If you decide to reassemble PLOT.MAC, I have included a program that will make a BASIC file that contains the machine language for any area of memory. So, assemble PLOT.MAC using M80. Then using L80 set the program area to the area you wish to use for the routines, then load PLOT.REL. Using the /m switch will give you the locations of the routines. Then use the /e switch and the routine will be loaded. So to summarize:-

1. Assemble PLOT.MAC => m80 =PLOT
2. Load L80. => L80
3. Set the program area to a convenient location => /p:adr
4. Load PLOT.REL => PLOT
5. Use /m to get locations of routines => /m
6. Use /e to move into correct area => /e
7. Save the routine using any method.
8. When the routines are reloaded do not forget to protect them with /m:&hadr statement when you load EXBASIC.

The file PLOTDEMO.BAS is a sample of some of the things that can be done using the plotting routines. In Ver 1.5 of the routines, which I am working on now, there are functions for drawing vectors and setting the origin anywhere on the screen area. There is also a function for testing whether a point on or off.

Step to Ensure Correct Usage of Plotting Routines

1. Call EXBASIC correctly => EXBASIC /m:&h9aff
2. Make sure that the graphics routines are loaded in.
=>RUN"PLOT
3. Make sure that the address for the routines are correct.
4. MAKE SURE THAT ALL THE VARIABLES THAT YOU PASS ARE INTEGERS
5. Call ginit to initialize the graphics, before you use the routines.
6. Remember that you can only pass integer variables, not values.
(eg. CALL SET(1,2) will not work)
7. Make sure your values do not go out of range. This has funny effects, sometimes the graphics on the screen change.
8. Don't get the x and y coordinates mixed up. Your plot will look funny.

```
110 ' Plotting Routine Demo Driver
120 ' -----
130 '
140 ' Jacques Giraud, Spectra Electronics
150 ' April 1981
160 '
170 ' This program may be copied only for non-profit use.
180 '
```

```

190 DEFINT W-Z,B      'All variable passed to routines must
                      'be integers.
200 GINIT=&H9B00      'Initialize graphics
210 PLOT=&H9B2B       'General purpose plotting routine
220 RST=&H9B21        'Reset at X,Y
230 SET=&H9B0C        'Set point at X,Y
240 WHITE=1           'Colour for white square
250 BLACK=0           'Colour for dark pixel
260 CALL GINIT        'Set the graphics right
270 PRINT CHR$(12)    'Clear the screen
290
300 '   Plot Sine Wave using PLOT
310 '   Plot Cosine wave using SET
320
330 Y=59/2            'find position to plot X axis
340 FOR X=0 TO 127    'Draw the line
350   CALL PLOT(X,Y,WHITE)
360 NEXT X
370 X=127/2          'calculate position of Y axis
380 FOR Y=0 TO 59
390   CALL SET(X,Y)
400 NEXT Y
410 FOR A=0 TO 12.7 STEP .035
420   X1=A*10
430   Y1=SIN(A)*29+29
440   Y2=COS(A)*29+29
450   CALL PLOT(X1,Y1,WHITE)
460   CALL SET(X1,Y2)
470 NEXT A
480 CURSOR 20,28:PRINT"Hit any key to continue":A$=INPUT$(1)
490 PRINT CHR$(12);
500 A$=STRING$(1920/8," ")      'Make string to reverse screen
510 PRINT                        'Invert Screen
520 CURSOR 0,0:PRINT CHR$(1);   'Move cursor to home position
530 Y=59/2                        'Plot axis again
540 FOR X=0 TO 127
550   CALL PLOT(X,Y,BLACK)
560 NEXT X
570 X=127/2
580 FOR Y=0 TO 59
590   CALL RST(X,Y)
600 NEXT Y
610 FOR A=0 TO 12.7 STEP .035
620   X1=A*10
630   Y1=SIN(A)*29+29
640   Y2=COS(A)*29+29
650   CALL PLOT(X1,Y1,BLACK)
660   CALL RST(X1,Y2)
670 NEXT A
680 END

```

PLOT.BAS

```

10 DEFINT A-Z:ADR=-25856:READ A:WHILE A(<)-1:POKE ADR,A:ADR=ADR+1:READ A:WEND:END
10000 DATA 033,156,155,017,000,254,001,040
10010 DATA 000,237,176,201,024,009,126,050
10020 DATA 137,155,026,050,136,155,201,205
10030 DATA 014,155,062,001,050,138,155,024
10040 DATA 017,205,014,155,062,000,050,138
10050 DATA 155,024,007,205,014,155,010,050
10060 DATA 138,155,175,237,075,136,155,203
10070 DATA 057,143,203,056,143,022,001,183
10080 DATA 040,005,203,034,061,032,251,103

```

```

PLOT.BAS (continued)
10090 DATA 105,042,006,041,061,032,252,072
10100 DATA 071,009,001,128,240,009,071,078
10110 DATA 221,033,139,155,221,126,000,183
10120 DATA 040,008,185,040,006,004,221,035
10130 DATA 024,242,071,058,138,155,183,040
10140 DATA 004,120,178,024,003,122,047,160
10150 DATA 071,221,033,138,155,004,221,035
10160 DATA 005,032,251,221,126,000,119,201
10170 DATA 000,000,001,032,161,160,169,150
10180 DATA 167,166,192,149,165,168,193,170
10190 DATA 194,195,196,000,255,255,255,255
10200 DATA 240,240,240,240,255,255,255,255
10210 DATA 015,015,015,015,240,240,240,240
10220 DATA 255,255,255,255,015,015,015,015
10230 DATA 255,255,255,255,255,255,255,255
10240 DATA 255,255,255,255,000,000,000,000
10260 DATA-1

```

How to Use MAKML.BAS

MAKML is a very simple program. It will create for you the file in which to place the data, then it will make a program of the machine language area that you specify. Remember that in an input statement hexadecimal numbers are considered legit.

```

Output Filename? PLOT.BAS
Start Address of Program? &H9B00
End Address of Program? &H9BC2

```

After this there will be a lot of grinding, as the file is written onto the disk. Then with a simple run, you can reload that machine language. MAKML also puts a one line program that will load the data at the top of the program.

MAKML.BAS

```

100 A$="Make Machine Language => BASIC":PRINT CHR$(12):
PRINT A$:PRINT STRING$(LEN(A$),137):PRINT
110 DEF FNMS$(A)=RIGHT$(STR$(A),LEN(STR$(A))-1):
DEF FNP$(A)=LEFT$("000",3-LEN(FNMS$(A)))+FNMS$(A)
120 INPUT"Output Filename";F$:C=10000:OPEN "O",1,F$
130 INPUT"Start Address of Program";ST:INPUT"End Address of Program";EN
140 PRINT#1,"10 DEFINT A-Z:ADR=";ST;"":READ A:WHILE A(<)-1:POKE ADR,A:ADR=ADR+1:
READ A:WEND:END":A$=""
150 FOR X=ST TO EN STEP 8
160   FOR Y=X TO X+7
170     A=PEEK(Y)
180     A$=A$+FNP$(A)+", "
190   NEXT Y
200   A$=LEFT$(A$,LEN(A$)-1)
210   PRINT#1,C,"DATA ";A$:A$=""
220   C=C+10
230 NEXT X:PRINT#1,C+10,"DATA",-1
240 CLOSE 1
250 END

```

By: Jacques Giraud

Spectra Electronics, Software Support.
 Toronto: 656-9646 (Till May 10/81 only)
 Calgary: 403-264-7270

Unfortunately we cannot include in this issue the 'PLOT.MAC' file listing but this will be included in the June issue.

This will complete the Plotting routines that 'Jack' wrote. I find that it is very fast, but alas Basic again. Well I wonder how long it will take for someone to let us know whether or not anybody has interfaced a Hi Res Graphics board with the Sorcerer?. Sure would like to hear about that one.

Sarcastic comments by: the editor (please disregard)

As some of you might have gathered from my rantings & ravings in the last issue, I dig 'Wordstar'. As some of you will also say 'each to their own'. Well let me do a little comparison as far as complexity of the command functions go. From the brief time that I have spent with SPELLBINDER, it is definitely more of a program than WORDSTAR. It has many features that Wordstar doesn't, and hence it is more complex to use. As any type of program goes, the more you use it the more familiar you become with it.

If you're like me, I use the programs not on a daily basis but rather just once in a while. This then tends to suggest that one does not fully become familiar with all aspects of the 'finer points' of the features that each program has. It also points out that, since each person tends to take the easier route, the simpler it is to use, the more you like it.

Now then getting back to Wordstar, the command menus can be displayed on the screen at any time one wishes, I find that a lot handier than having to look up in the bible, what a particular command sequence should be. Ever try to find room for the instruction manual, or that it stays open at a particular page, well I have and believe me sometimes they start looking very ragged and torn.

Below you will find a listing of almost exactly what is shown at the top of the screen, for each command group.

^Q CURSOR S=Left side E=Top D=Right END line C=End file
 R=Start File X=Bottom P=Back to prev/cond V=Start of last F/Repl
 Delete to Del=LEFT Y=RIGHT (or entire line)
 Find/Replace F=Find a String A=Find and Substitute
 REPEAT NEXT COMMAND G=Repeat until Key Hit

^J H = Display and set HELP level M = Margins and Tabs
 F = Flags in right screen column S = Status Line
 I = Command index, Entering text R = Ruler Line
 B = Paragraph REFORM (^B command) V = Moving text
 D = Dot commands, Print controls P = Place markers

^K End EDIT/SAVE :D = Done X = Done/Exit S = SAVE, REDIT G = ABANDON
 MARK BLOCK :B = Block start K = Block end H = Hide/Display
 Block operation:V = Move C = Copy Y = Delete block W = WRITE
 ADDING FILES :R = READ W = Write J = DELETE
 PLACE MARKERS :0-9 = SET/HIDE Place markers 0-9
 PRINTING : P = Print a file

^O S = Line Spacing C = Center cursor line F = Margins/Tabs from line
 L = Left Margin X = Margin Release P = Page display off (on)
 R = Right Margin W = Word wrap off (on) D = Dot CMD display off (on)
 I = Set Tab Stop J = Justify off (on) T = Ruler display off (on)
 N = Clear Tabs V = Vartabs off (on) Space = Cancel Prefix

^P F = Phantom space T = Proportional char. Y = Enhanced on/off
 S = Underline B = Boldface D = Double strike
 R = ESC, Programing Q = 10 C.P.I X = Strikeout Begin/end
 O = Non-Break Space W = 12 C.P.I G = Phantom Rubout
 C = Pause E = 16.8 C.P.I

Return = Over print line follows Space = Cancel prefix

The following commands can be mixed into the text but will have to be at the beginning of a line. These do not count as lines on the page, but as command lines instructing Wordstar 'from point of insertion' to execute different formatting to the printer. Most of the commands are simple and as usual, saved with the file for future reference.

DOT COMMANDS

.PA New Page

.CP(n) New Page if less than (n) lines left on this page

.OP Omit page numbers when printing, starting this page

.PN Print page numbers starting this page (default)

.PN(n) Set page number to (n), Print page numbers

.PC(n) Page number column (default 33 (30 if 64 column screen))

.PO(n) Page offset: Extra indent when printed (default 8)

.Text comment (not printed)

.HE Text heading used until next .HE (default blank)

.FO Text footing (replaces Page number)(default blank)

.PL(n) Paper length: Total number of lines (default 66)

.MT(n) Margin at top: # lines, top of paper to text (default 3)

.MB(n) Margin at bottom: # lines, end of text to end of paper (default 8)

* Lines text on page is PL + MT + MB (default to 55)

.HM(n) Footing Margin: Blank lines, text to footing (default 2)

For page breaks to display as they will print, use .PL, .MT, .MB, and .LH (next frame) at file beginning only.

THE FOLLOWING WORK ONLY WITH INCREMENTAL PRINTERS

.LH(n) Line height in 48ths of an inch. (default 8)

.CW(n) Character width in 120ths of an inch, for standard or alternate pitch, whichever is in use (see ^P menu) (Default 12 for standard, 10 for alternate)

.SR(n) Subscript/Superscript roll in 48ths of an inch (3)

Prints as current page number

\ Do not interpret next character as a special character

AK Do not print following spaces if on even numbered page

Page number positioning: if a footing text is specified (.FO), The default bottom center page number is not printed

Use # in the heading or footing to position page number where desired

AK Followed by spaces followed by # may be used to produce alternating LEFT/RIGHT page numbers

FLAG CHARACTERS(right most column of screen)

< Line ends in "hard" carriage return, Entered by user

This line break arose from word wrap or paragraph REFORM, and may be moved on subsequent REFORM

+ This line of document continues on next screen line

- Next line will over print this line

? Unrecognized or incomplete DOT command

---P Page Break

. This screen line is after end of document

: This screen line is before beginning of document

This concludes the short burst of philosophy from the editor and I hope it has enlightend some of you in the realms of Wordstar.

By: H.A.Lautenbach

NOTICE TO MEMBERS

As all of you are aware, the membership dues have been kept the same now for almost two years. We have tried to keep them as low as possible since we are a NON PROFIT making organization. Due to the increased printing costs and mailing we have had to raise the dues a little. An analysis of the current costs would have put us in the red in the fall of this year.

The increase shall maintain the same quality of PORT FE as you have received since January of this year and it will also insure us of a monthly meeting place. All future meetings will be held at:

Bathurst Heights Library
3170 Bathurst St.

one block north of Lawrence on the west side of Bathurst.

The EDITOR

48K Ram Added to the Sorcerer I

In the earlier part of the year there was a Sorcerer modification passed down to us from the Australian Group. This modification has been made to TWO machines here in Toronto as of this time. At first there was a problem. One of our members (Robert England) decided to implement the changes and found some discrepancies. There were two - 1. paragraph 3 stated to bend up pins 1-6 and 8-12, this should have read 8-13, 2. Reference to I.C. 10B on the schematic should have been 10C. The following should clarify it for everyone. First let me bring one more detail to your attention. Anyone making this modification should also be aware to change (I would recommend) all rams to 150ns, just in-case this causes any problems for those of you who are using disk drives.

Directions for adding an additional 16K of RAM to the Sorcerer I.

- a. Parts required 8 - Rams 4116 150ns, 1 - 74LS10, 8 - 16 pin sockets
1. Solder 16 pin I.C. sockets onto the second row ram chips. Bend out pin 4 of the NEW 4116 RAM chips & insert into the sockets.
2. Cut pins 1,2 & 3 of I.C. 11B (74LS08), Be very careful not to damage the I.C. (this way you will not have to remove the chip) and bend up pins 1,2 & 3.
3. Bend up pins 1-6 and 8-13 of the NEW 74LS10 chip and solder the remaining pins '7' common & '14' +5v in the same orientation piggyback onto I.C. 11B.
4. Complete the wiring using wirewrap insulated wire or equivalent as shown in the shaded areas below.

All modifications only affect the 'MULTIPLEX CONTROLLER' section of the Sorcerer circuit board.

