MEMOTECH
MEMOPAK
HRG

Read Me First-
Before You Start!

MEMOTECH CORP
7550 West Yale Avenue, Suite 200,
Denver,
Colorado 80227
Telephone: (303) 986 1516
Twx: 9103202917

This equipment generates and uses radio frequency energy and if not installed and used properly, that is, in strict accordance with the manufacturer's instructions, may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

— Reorient the receiving antenna.

— Relocate the computer with respect to the receiver.

— Move the computer away from the receiver.

— Plug the computer into a different outlet so that computer and receiver are on different branch circuits.

— If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions. The user may find the following booklet prepared by the Federal Communications Commission helpful:

"How to Identify and Resolve Radio-TV Interference Problems".

This booklet is available from the U.S. Government Printing Office, Washington, DC 20402, Stock No. 004-000-00345-4.

WARNING:

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only Computers/Peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this equipment. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

Hello User,

By now you may have become quite expert at BASIC programming on the Sinclair ZX81. You're probably making good use of one of the Memotech MEMOPAKs or another commercial RAM pack. But you'd like to try graphics – programming in a new dimension. Graphics can be great fun. Apart from space zombies to be zapped without serious risk to your health, there is a new world of design, animation, geometry and presentation to try.

Again, we hope that you like our elegant design. Now let's tell you how it works.

How do I set up the Memopak HRG?

Disconnect your ZX81 power supply, and install your HRG pack between the ZX81 and your RAM pack. You need a RAM pack because the HRG routines operate primarily on the memory – a screen image is held as a 'video page' in about 6.2K of RAM. The more RAM you've got the more video pages you can use.

We recommend you use the Sinclair 1200 milliamp power supply as supplied with the Sinclair printer and recent ZX81s.

You will find all MEMOPAK units plug in firmly, but for the best connection we suggest you cut up and use the velcro tabs provided. Moreover, our HRG pack utilises the addresses between 8 and 10K, so if you have a MEMOPAK 64K you will have to make sure this area is switched out. This means you must set your switch either in MODE B (2 ON; 1, 3 and 4 OFF) or MODE D (4 ON; 1, 2 and 3 OFF). Remember ON is UP. With a 64K RAM pack you will probably want to put your video page above 49152. In this

case you should remove the little square jumper switch next to the back connector board. It's best to use a pair of tweezers or pliers (but disconnect the power first). To keep the jumper safe, you can replace it on just one pin.

Is there a quick test to show my pack is running OK?

Yes. Choose a spare memory area of at least 6.2K and set V to its start byte address. Then do our special call. So enter as follows:

LET V = 19000 (with jumper in)
or
LET V = 50000 (with jumper out)
RAND USR 8619

What you should get is MEMOTECH appearing in white on a black background. If not, check your configuration and your connectors, and try again.
When it's working, you can test the 'BASIC RE-SET' button which is located on the side of your pack. This is for returning you from your HRG display to your normal BASIC display.

Where and how are the HRG routines kept?

The MEMOPAK HRG contains a 2K EPROM where our subroutines reside and it's full to bursting. We have provided at least 30 functions in the firmware to help the user get the most out of the HRG. Our philosophy here has been in the Sinclair tradition: help the user to help himself. We have increased the scope of your activity; we leave you free to explore the new territory.

How does the HRG work?

Simply, the HRG allows you to output in dots rather than characters (1 character = 8 × 8 = 64 dots) or Sinclair pixels (1 pixel = 4 × 4 = 16 dots). Since the dot is the smallest graphics item, it offers the maximum flexibility. Any shape, character or pattern can be generated by the right combination of dots, and the possibility of smoother curves increases.
In everyday life, to fill a page with dots is a laborious process. But here the sweat is taken out of it by a number of means:

a) Subroutines which work much faster because they operate directly in machine code.
b) Full use of looping facilities in both BASIC and machine code.
c) The ability to pre-plot special characters or mix in characters from the Sinclair range.
d) Powerful rolling and scrolling facilities.

The range of subroutine calls is listed in a table in the back of the booklet. They span many functions, and are particularly powerful in their ability to make the most of your memory. They also span a number of different levels – dot, line, character, block and page.

How is the screen related to the memory?

As far as our pack is concerned, your screen is made up of 192 × 248 dots or HRG pixels (the equivalent area of 24 × 31 BASIC screen character positions). The leftmost column of the screen is ignored by the HRG – for subtle software reasons. All screen activity takes place in a 'video page' of memory which you can show on the screen at the same time. You can also manipulate the 'video page' without showing it on the screen; or you can

be looking at another video page while doing so!

How is a video page organised?

A variable is reserved by the HRG system to hold the memory address of the start byte of the video page. So always remember to set variable V to the start of the particular video page you want to reach.
Roughly speaking, one screen dot (or HRG pixel) is stored in one bit of memory. However, when placed in memory each horizontal 'line' of 248 graphics bits is preceded by 2 bytes of control data. In addition, there is a single byte of screen control data terminating the whole video page. So the arithmetic looks like this:

Bit Summary

	Line Control		Screen Graphics		HRG Memory
Line	16	+	248	=	264
Page	3072	+	47616	=	50688

1 video page = 50688 + 8 page control bits = 50696 bits.

Byte Summary

	Line Control		Screen Graphics		HRG Memory
Line	2	+	31	=	33
Page	384	+	5952	=	6336

1 video page = 6336 + 1 page control byte = 6337 bytes.

Whereabouts in memory can I put a video page?

You must choose an area which is not otherwise occupied by the system variables, the instructions, the display file or your other arrays. The simplest thing is to set a low RAMTOP leaving you enough space (6337 bytes per video page) between that and your real limit of memory. Clever users may find other ways.

How do I use the HRG subroutines?

A subroutine is called by using the Sinclair USR function. We recommend the use of an instruction such as RAND, as this has few side effects, but other instructions may do. All the subroutines can be called initially from one address (8192), so that the same call can be used for any function:

```
RAND USR 8192
```

If however you want to reserve RAND to set the seed for the randomizer, then use, say, LET A = USR 8192. To nominate a particular subroutine however you must first specify its name in a reserved variable Z$. So to call for example the simple PLOT function (which sets a bit in the video page) you must first enter the instruction:

```
LET Z$="PLOT"
```

"PLOT" in turn uses the X and Y variables to find out the co-ordinates to be plotted. So the whole step goes like this:

```
LET X=64
LET Y=100
LET Z$="PLOT"
RAND USR 8192
```

This little routine, *provided* the right initialisation steps have been taken, will set one bit in the video page specified.

How do I use the MEMOPAK HRG with a BASIC program?

The MEMOPAK HRG has been designed so that you can use it in ordinary BASIC. There are also routines which allow you to switch the screen from conventional BASIC display to the HRG display and back again. While you are developing your program, you can use the manual 'BASIC RETURN' button on the side of the pack which puts you back into BASIC display mode, without in any way endangering your graphics data. When you return back to the BASIC display mode, you may be upset to find a blank screen. This is blank simply because although your BASIC program is running, it has not been requested to output anything to the screen. You'll probably want to look at your program listing at this point, so you need to use the Sinclair BREAK and LIST functions.

What about FAST mode?

These routines work rapidly in FAST mode, at least as rapidly as graphics on the Sinclair Spectrum and with more pixels. However, in FAST mode, you lose the screen and the gratifying experience of seeing the graphics at work for you. Moreover in FAST mode the 'BASIC RETURN' button cannot work, so if you want to get back to BASIC display manually, you have to break in and enter "SLOW" even though there will be no screen response until you've done it. Now you can use the 'BASIC RETURN' button.

How do I initiate an HRG routine in my BASIC program? (Page functions)

The first thing is to set the start byte of your video page in the reserved variable V. Then you must cite and activate the "START" function. Functions must always be cited in reserved variable Z$ prior to the call. "START" does two things: initiates the HRG system and assigns the video page area (start address cited in V) to it. (Later in the program the "PAGE" function could be used to assign another video page without re-initiating the system.) (Note: the groups of instructions which follow are typical enough, but when all put together they may form a pretty absurd program. For 'real' program examples see the back of this booklet).
Your routine so far could well look like this:

```
10 LET V=40000
20 LET Z$="START"
30 RAND USR 8192
```

You may now want to clear the video page area initially (or else you may pick up what you left there before – remember an area above RAMTOP is not cleared by Sinclair).

```
40 LET Z$="CLEAR"
50 RAND USR 8192
```

Note the same USR address is used whatever the function. "CLEAR" also uses variable V to find out where the page to be cleared is located. Since we are talking about the same page that was assigned with the "START" function, there is no need to re-set V. All the activity so far has gone on in the memory video page and we haven't seen a thing. To keep your eye on what's going on, let's do an "HRG" call. This will do an HRG display of the memory page rather than display anything your BASIC program may

have output. Don't worry, your BASIC outputs are still accessible (they're still in the Display or D-FILE), but they are now invisible.

```
60 LET Z$="HRG"

70 RAND USR 8192
```

Note that as we are still talking about the same video page, we still have not re-set V. The screen is now locked into our video page and anything that happens there, we can watch. "HRGINV" will also show what is in the page, but in reverse, and without changing the bits in the memory.
So in just seven lines we are ready to start depicting something. If seven lines is too much for you, we have kindly provided you with a macro (multiple) function: "STARCH". "STARCH" sounds like a new concept in programming for laundry control but really it simply means "START"+ "CLEAR"+"HRG" and the above program can now be shortened to 3 instructions:

```
10 LET V=40000

20 LET Z$="STARCH"

30 RAND USR 8192
```

To summarise: the HRG system is initialised; a video page is assigned from 40000 onwards; the page is cleared and the contents (blanks or unset HRG pixels) displayed.
We can just mention 3 other page level functions: "PRINT" will transfer any video page to the Sinclair printer. "STRING" will let you pass a video page into a long string which can then be SAVEd on cassette by the ZX81. "UNSTRING" will unpack a string into a video page for you on re-LOADing. The video string is always S$. You can use "STRING" simply to clear space for yourself above RAMTOP. But don't try to "UNSTRING" something which you did not "STRING" first, unless you are sure you have got the control characters right. When re-loading a video page with "UNSTRING" (that has been stored and SAVED within the "STRING" function) it is necessary to:

a) make sure you do not use RUN for the re-load section but GOTO, in order to prevent S$ from being cleared.
b) make sure you do not re-dimension S$ on re-entering as this will also lose the array
c) remember to re-set V and call the "STARCH", "UNSTRING" and "HRG" subroutines to see the video page again.

Sample re-load subroutine. Enter program with GOTO 1000.

```
 990 STOP
1000 LET V=50000
1010 LET Z$="STARCH"
1020 RAND USR 8192
1030 LET Z$="UNSTRING"
1040 RAND USR 8192
1050 LET Z$="HRG"
1060 RAND USR 8192
1070 GOTO ....
```

Incidentally, if you don't like the idea of repeating the subroutine call – e.g. RAND USR 8192 – all the time, you may like to place it in a little subroutine. For some repeated calls, you might also like to set the parameters from within the routine as well.

What can I do now?

You are now free to experiment with the other subroutine functions (and their parameters) listed at the back.

They are broken down into five kinds: page functions, block functions, character functions, line functions and dot (or HRG pixel) functions. The page functions have already been described above. As for the rest, let's start small.

HRG pixel (or dot) routines

The simplest function is "PLOT" which requires X and Y co-ordinate parameters in addition to V:

```
80 LET X=50

90 LET Y=60

100 LET Z$="PLOT"

110 RAND USR 8192
```

This will place a dot on your screen. Remember that the co-ordinates in X and Y work like this:

	Axis	Range	Direction
X	Horizontal	0–247	Left to right
Y	Vertical	0–191	Bottom to top

"UNPLOT" works in exactly the same way, so:

```
120 LET Z$="UNPLOT"

130 RAND USR 8192
```

will take the dot away again. If you want to find out whether an HRG pixel is set use "TEST", this time calling the routine with the Basic LET instruction for the reply:

```
140 LET Z$="TEST"

150 LET K=USR 8192

160 IF K=0 THEN .....

170 IF K=1 THEN .....
```

Any variable may be used; if the routine returns 0 then the pixel is unset, if 1 then set; and you can take action accordingly. In this case, since the routine is still testing the old X, Y locations, which we UNPLOTted, then the reply will be zero.

With "HRG" called, your screen displays an image of the video page. However you can use the "LOCATE" function to find the absolute memory address of a pair of co-ordinates, and so manipulate the video page directly. Remember however, that the video page is not exactly the same as a screen; there are two control bytes preceding each line (the first one is a Sinclair newline byte (118) and the second one is zero) and one more byte after the last line in the page (also 118). So if you want to interfere with a video page directly, take these into account. To find out where in memory our pixel location is, enter:

```
180 LET Z$="LOCATE"

190 LET K=USR 8192
```

and K will receive the address.

Can I run two or more video pages together?

Yes, they can be located next to each other in memory and will effectively be joined vertically. In this case the first line control byte of the later video page should be the same as the last page control byte of the first video page. It is now possible to look at any 'intermediate' page made up of later lines of the earlier page and earlier lines of the later page. To call such a page make sure that your start byte value placed in V is displaced from the start bytes of the 'real' video page by a multiple of 33.

What about Geometry?

At this point we can introduce our

simple geometric functions. We could have provided more but we think it is an excellent chance for users to brush up on their skills. The only functions we provide are "LINE", "UNLINE", "BLINE" and "WLINE"; any other shape can be effected by combinations of these or the "PLOT" call, and the Sinclair maths functions.
You'll need to work on your algebraic geometry, or dust off a text-book to get algorithms for curves, sine waves and so on. Try this natural log curve:

```
200 LET Z$="PLOT"
210 FOR X=1 to 245
220 LET Y=33 * LN X
230 RAND USR 8192
240 NEXT X
```

Line routines

"LINE" will join the points specified in the pairs of co-ordinates P, Q and X, Y and you should be able to do straight-edge geometry yourself.
"UNLINE" will wipe out a line in the same way.
"BLINE" (black line) and "WLINE" (white line) are used only for vertical plotting and will draw a line upwards from X, Y until a bit of the same setting is encountered. The vertical co-ordinate of the setting is returned in Y.
Used in conjunction with the "LAUNCH" routine, some sophisticated shading routines are possible. See our example at the back.
Lastly, for the zappers, there is our "LAUNCH" routine. This will fire a laser beam vertically up the screen and detect a 'hit' (i.e. if any of the bits in its path were set). If there is a hit, it passes back the vertical co-ordinate and vanishes. Otherwise, it disappears off the top of the screen. "LAUNCH" need not be used so aggressively; it can also serve as a 'radar' function to detect the presence of the first bit set in a vertical column rising from the X, Y position cited. This can be used for shading or blocking in.

Character routines

There are two kinds of character you can plot – those you've designed yourself and those Sinclair provides. To design a simple horizontal line of bit settings, simply set up a string of 0's and 1's in C$, with a colon as terminator:

```
250 LET C$="10101:"
```

Instead of 0, you can use * to avoid disturbing the initial status of a bit. To design a 2-dimensional character you can simply use N, S, E or W (or a combination) to re-set the start location of the next series of bit settings, and continue. So we could plan a cross on the screen instead:

Type in

```
250 LET C$="1***1NE1*1NE1NW1
    *1NW1***1:"
260 LET X=30
270 LET Y=40
280 LET Z$="SKETCH"
290 RAND USR 8192
```

and the cross will appear with the bottom left-hand point located at 30,40. In this example, we built the character upwards using N, but we could have dropped it downwards using S, or missed out a line altogether using SS.
"UNSKETCH" has the effect of unsetting the bits set in the string.

"INVSKETCH" is like "SKETCH" except that it reverses the bit settings, placing a reverse image of your character in the video page itself.
"SINCH" does the same thing as "SKETCH" except that you need to cite a Sinclair character. For reverse images, you can use Sinclair's own inverse functions. A whole string of Sinclair characters can be displayed anywhere in the page, starting at the X, Y point which represents the bottom left-hand corner of the first character. To wipe out a "SINCH" display, use the Sinclair space character.

Block handling routines

For block handling routines, to save on tedious co-ordinates we have hit on the concept of north, south, east and west. For this reason, variables N, S, E and W are called.
We have two dynamic routines which can be useful for animated displays – roll and scroll. The "roll" functions take a block and shift the constituent lines up or down, taking the line that has dropped off the block at one end and adding it on at the other. This is done according the following 2-character command strings – "RU" (roll up) and "RD" (roll down), and these will use the N, S, E and W parameters.
A similar set of commands exist for the scroll functions. "Scroll" differs from "roll" in that the line that drops off will completely disappear and a blank line will step in at the other end. Scrolling can be done horizontally as well as vertically. So these are the last four command strings you've got: "SU", "SD", "SR" and "SL".
Note that when moving horizontally, E must be greater than W; when going vertically, N must be greater than S.

Scrolling and rolling is most effective when set in a loop.

Subroutine Calls

Command string (Z$)	Function	Parameters
Page routines		
START	Initiates HRG system and assigns memory video page	V
PAGE	Assigns memory video page	V
CLEAR	Clears a page	V
HRG	Displays a page	V
HRGINV	Displays a page inversely	V
STARCH	Macro = Start + Clear + HRG	V
PRINT	Prints a video page	V
PRINT1	Prints top line of video page	V
BASIC	Displays current BASIC page	NONE
STRING	Copies page into BASIC string	V, S$
UNSTRING	Copies string into video page (requires DIM S$(6337) or more)	V, S$
Block routines		
RU	Roll Block up	V, N, S, E, W
RD	Roll block down	V,N, S, E, W
SU	Scroll block up	V, N, S, E, W
SD	Scroll block down	V, N, S, E, W
SR	Scroll block right	V, N, S, E, W
SL	Scroll block left	V, N, S, E, W
Character routines		
SKETCH	Plots user-defined character	V, X, Y, C$
UNSKETCH	Unplots user-defined character	V, X, Y, C$
INVSKETCH	Reverses user-defined character in page and screen	V, X, Y, C$
SINCH	Plots ZX81-defined character	V, X, Y, C$
Line routines		
LINE	Draws a line	V, X, Y, P, Q
UNLINE	Wipes out a line	V, X, Y, P, Q

*BLINE	Draws black line 'UP' until set BIT	V, X, Y
*WLINE	Draws white line 'UP' until unset BIT	V, X, Y
*LAUNCH	Draws momentary line 'UP' until set BIT	V, X, Y

Dot routines

PLOT	Sets one BIT/HRG pixel	V, X, Y
UNPLOT	Unsets one BIT/HRG pixel	V, X, Y
*TEST	Gets a setting of a BIT	V, X, Y
*LOCATE	Gets a memory location of a BIT	V, X, Y

*These calls elicit a reply and so should be made with the LET statement:

```
LET G=USR 8192
```

will place the reply in G. "LAUNCH", "BLINE" and "WLINE" will get a value of the vertical co-ordinate or zero if none is found. "TEST" will get a value of zero for an unset, one for a set bit. "LOCATE" will get the absolute memory address of a bit set. Or you can use a logic test:

```
IF NOT USR 8192 THEN GOTO 1000
```

HRG Parameters

Z$ = Command string
V = Start byte of assigned video page
X = Horizontal co-ordinates (0–247)
Y = Vertical co-ordinates (0–191)
P = Secondary horizontal co-ordinates (0–247)
Q = Secondary vertical co-ordinates (0-191)
N = Uppermost block line (0-191)
S = Lowest block line (0-191)
E = Rightmost block line (0-247)
W = Leftmost block line (0-247)
C$ = String for "SKETCH", "SINCH" etc.
S$ = String for storing video page data

HRG error codes

L = Parameter variable not declared
M = HRG command string not known
N = Horizontal parameter too large
O = Vertical parameter too large
P = Invalid element in "SKETCH" string

Here are some examples to try:

A) To draw two waves, and shade in the areas between.

```
10 LET V=40000
20 LET Z$="STARCH"
30 RAND USR 8192
40 LET Z$="PLOT"
50 FOR X=0 to 247
60 LET Y=100+50 * SIN(X/20)
70 RAND USR 8192
80 LET Y=100+SIN(X/30)*
   COS(X/15)*50
90 RAND USR 8192
100 NEXT X
110 FOR X=0 to 247
120 LET Y=0
130 LET Z$="LAUNCH"
140 LET Y=1+USR 8192
150 LET P=USR 8192
160 IF P=0 THEN GOTO 190
170 LET Z$="BLINE"
180 RAND USR 8192
190 NEXT X
```

Run it and see. Is it a bird? Is it a Loch Ness Monster?
A few notes about the program: "LAUNCH" detects the first set bit in a vertical pattern. In line 140 we add 1 and use that as the start of "BLINE" which will draw a black line up until the next set bit. Sometimes, we don't want to draw a line up, so we test to make sure that a second bit exists as in lines 150–160.

B) To make a beautiful pattern from curves. Is it a bird? Is it a seal?

```
10 LET V=40000
20 LET Z$="STARCH"
30 RAND USR 8192
40 LET Z$="PLOT"
50 FOR A=10 TO 100 STEP 3
60 LET D=A*A
70 LET C=D*190
80 FOR X=0 TO 247
90 LET Y=C/(X*X+D)
100 RAND USR 8192
110 NEXT X
120 NEXT A
```

C) Here is a program which shows a

funny clock. We're sure you could make it better – more accurate; with hours as well. Change line 80 to make it slow down or speed up.

```
 10 LET V=40000
 20 LET Z$="STARCH"
 30 RAND USR 8192
 40 LET P=100
 50 LET Q=100
 60 FOR T=1 TO 60
 70 GOSUB 130
 80 FOR Z=1 TO 20
 90 NEXT Z
100 GOSUB 210
110 NEXT T
120 STOP
130 LET X=100+50*SIN
    (T*PI/30)
140 LET Y=100+50*COS
    (T*PI/30)
150 LET Z$="SINCH"
160 LET C$=STR$ T
170 RAND USR 8192
180 LET Z$="LINE"
190 RAND USR 8192
200 RETURN
210 LET X=100+50*SIN
    (T*PI/30)
220 LET Y=100+50*COS
    (T*PI/30)
230 LET Z$="UNLINE"
240 RAND USR 8192
250 RETURN
```

Good Luck from all at MEMOTECH

FURTHER INFORMATION ON MEMOTECH PRODUCTS

Memotech produce a range of add-on Memopaks for the ZX81. We will be pleased to send information sheets on any of the following packs:

	Please Tick
• **Plug-in Keyboard**	☐
• **64K RAM PACK**	☐
• **32K RAM PACK**	☐
• **16K RAM PACK**	☐
• **Hi-Res Graphics**	☐
• **Centronics Type Interface**	☐

COMING SOON

RS232 Interface

Digitising Tablet

We'll let you know, via our Press advertisements, when information becomes available on the above products.

Please send this page to the address given on the back cover of this leaflet.

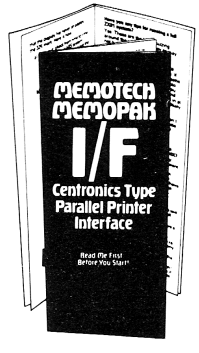
MEMOTECH
Explores the Excellence of your ZX81

WHY CHOOSE MEMOTECH?

Memotech systems are used not just in the home and in small businesses, but also in larger organisations where localised, efficient data handling is required. We feel this is due to the advantages users gain from the points listed below.

- **reliable and efficient in operation**
- **compact, stylish, high quality extruded aluminium casing**
- **forward compatibility with Memotech products**
- **extensive documentation with sample programs**
- **flexibility of operation modes**
- **full guarantees/refunds or replacement**
- **exchange options (16K)**
- **full after-sales service**
- **efficient quality control**

INSTRUCTION MANUALS



Memotech provide substantial documentation for all their products, in the form of individual booklets for each Memopak. In addition to basic information on how to use your Memopak the booklets contain program examples and material on the internal architecture and functions plus our guarantee for use in the unlikely event of any malfunction.

Guarantee

April 1982

This product is guaranteed free from defects in material and workmanship for a period of six months from the date of purchase subject to the following conditions:

1. **The guarantee does not cover any damage caused through neglect, incorrect adjustment, accident or misuse and will be invalidated if the product is modified or altered in any way or repaired by anyone other than Memotech Corporation.**
2. **Claims under this guarantee must be made by sending the product (well packed preferably in its original packing) to the address below.**
3. **This guarantee is valid only in the case of a purchaser resident in the United States.**

Memotech Corp
7550 West Yale Avenue
Suite 200
Denver, Colorado 80227

Model: Memopak HRG

Serial No: (if any).....

Date of Purchase.....

Owner's Name.....

Owner's Address.....

.....

.....

Method of Purchase*.....

Point of Purchase.....

*** Please enclose evidence of purchase (receipt etc) as without this we will be unable to give a refund or replace this item.**

WHAT THEY SAY ABOUT MEMOTECH

"a Rolls Royce add-on."
"The new RAMS from **Memotech** are beautifully designed and blend in really well with the styling of the ZX81."

ZX Computing
Aug/Sept 1982

"The **Memopak** is undoubtedly the ultimate memory expansion for the Sinclair. Try DIM A (9500)!"
"The documentation is good—answering first-time users' questions. It also gives demonstration programs."

Syntax
August 1982

"The 64K version has four switches visible in its rear which allow you to switch out the area between 8K and 16K in the memory map in 4K blocks. This is an excellent idea and I hope other manufactuers will follow this lead . . . Only the **Memotech** allows you to add something between 8K and 16K."

ZX Computing
Aug/Sept 1982

"Anyone who wishes to use his Sinclair for any form of data handling—address lists, stock control, etc., would very quickly run out of space . . . the **Memotech** 64K RAM is designed to overcome this problem."

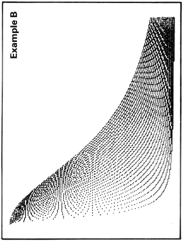
Hobby Electronics
June 1982

". . . there's no denying that the **Memotechs** are the best-styled and the best-made RAM packs on the market."

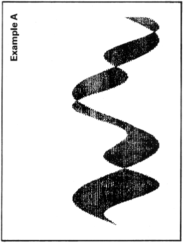
ZX Computing
Aug/Sept 1982

NOTES

NOTES



Example B



Example A