VORTEX
RAM EXPANSIONS
SP 64 – SP 512

USER HANDBOOK

vortex
COMPUTERSYSTEME

VORTEX
RAM EXPANSIONS
SP 64 – SP 512

USER HANDBOOK

ENGLISH EDITION: OCT 1985
TRANSLATION BY: STEPHEN H. JOHNS.
EXCLUSIVE U.K. IMPORTERS & DISTRIBUTION
SCREENS MICROCOMPUTER DISTRIBUTION
MAIN AVENUE, MOOR PARK, NORTHWOOD,
MIDDLESEX. ENGLAND.

Introduction

Dear CPC 464 User,

It is finally ready. Your CPC 464 memory capacity may now be expanded. Up to a maximum of 512KB. You yourself decided upon the vortex RAM-Expansion thereby selecting a thoroughly tested, high-quality product which is capable of meeting your every need and exceeding your fondest wishes.

Utilizing electronic components of the newest generation and meticulously thought out supportive software, we have succeeded in creating a peripheral device which fits completely inside your CPC 464 console.

If you've discovered the BASIC program "RAM" area to be less than adequate for your programming needs, or it disturbs you to wait while your printer is busy, or you find that 44KB is not enough for CP/M applications and Data/Wordprocessing software then we are sure that our product is just what you need solve these problems ..... Actually, it will solve EVERY problem.

Performance Characteristics:

- Cleverly designed to fit in virtually every type AMSTRAD CPC 464 console.
  Easy card installation.....no soldering, just plug it in.

- Simply by plugging in the appropriate RAM chips, you can expand all the way up to 512KB..... . No soldering required here either.

- **Intelligent** on-board software assures maximum performance at all expansion levels. (64,128,256,320,512KB)

- Usable with a floppy-disc drive (vortex 5.25" Floppy Disc Station **F1-S/D** or **Amstrad** 3" Disc Station DDI-1) or without (in this case only under BASIC).

- **Fully accessible under CP/M.** You recieve a 62KB CP/M upon which any standard CP/M program will run.
  You have the possibility of **128 directory entries.**
  Additionally, you may select a **32KB printer buffer** which will enable you to type and print at the same time. Depending upon your expansion level, you'll have up to 448KB for use as a **superfast RAM disc**.

- Under BASIC, you have up to 288KB Program space, and 256KB Data space. A slightly expanded BASIC instruction set allows you free handling of the new memory area.

- A superfast ROM-Resident Z80 Monitor which may be called from basic allowes you unlimited access to the heart of your CPC for Tracing, Dumping, Listing and Assembling and the setting of Breakpoints etc. in machine language.

- Addition of a BASIC Graphic command subset which will satisfy the wishes of any CRT artist.

- Buffered Expansion bus which is a requirement of the CPC 464 with drives, for the safe, secure operation of any future peripheral devices (RS-232, Centronics, Modem, Etc).

An additional plus is the vortex Service Pass. This pass is included in all vortex RAM Expansion packages, and not only reinforces the vortex guarantee, but insures you troublefree system upgrades, or software improvements as they occur.

We wish you complete satisfaction and enjoyment in the use of this fine product, and always optimum, troublefree use of your 464 with vortex RAM Expansion installed.

Your vortex team

IMPORTANT IMPORTANT IMPORTANT IMPORTANT IMPORTANT

Please note the particularities about our programs COPY, PARA, GRAPHIC MASTER 2.0 and SPTEST.CDM, described in the appendix at the end of this book.

Distribution and marketing rights of vortex products in U.K an elsewhere reats with Screens Microcomputer Distribution Limited.

vortex Computersysteme Vertrieba GmbH
Klingenberg 13
7106 Neuenstadt/Buerg
West Germany

Neuenstadt, August 1985

Installation of the vortex RAM Expansion in your CPC 464

In order to install the RAM Expansion card in your CPC 464, the case must first be opened. This requires a medium sized "Phillips" or "Cross-Tip" screwdriver. After opening the console, a small "Flat-Tip" screwdriver is required for the removal of the Z80 CPU and the Gate Array Logic chip.

Use of the Aluminum heatsink, which was included in the RAM Expansion package is highly recommended (Not mandatory in all installations) however, the isolation material must be installed for fail-safe operation of the card.

The following steps must be followed exactly as worded, if your RAM Expansion is to work as planned. Do not omit or skip any step during the installation procedure.

1. Turn off both the CPC 464 and monitor, as well as any other peripherals which may be connected. Disconnect all cables and connectors from the CPC Console.

2. Turn your CPC over, place it "face-down" on a towel or other soft material, and loosen all 6 fastening screws with the "Phillips" or "Cross-Tip" type screwdriver.

3. Carefully turn your CPC 464 back over, insuring that all 6 screws fall on the towel. Set your CPC 464 down in the normal "use" position, collect up the screws putting them aside, and proceed.

4. Gently separate the console halves (Top and Bottom), and tilt the keyboard towards yourself. Looking inside, you'll notice that there are 2 cable harnesses which plug into the main processor board (One at each end).
   Gently pull the right side connector (the smaller of the two) from the Main PC board, and then as gently pull the left one from the underside of the keyboard. No need to worry here, it is impossible to reconnect these 2 plugs falsely.

5. It is now possible to completely separate the two halves, and set the keyboard part aside.

6. We can now look down upon the CPC 464 main processor board itself. While the various components (Resistors, Diodes, Integrated Circuits, Capacitors, etc.) are interesting to look at, we are only going to deal with two. These being the Z80 Central Processing Unit (CPU) and the so called Gate Array.
   Both of these devices are of the 40-pin variety, that is they have 20 pins on each side and are quite large. The Z80 is located in the approximately center of the main PC board, and is mounted in a socket. The Gate Array is located in the upper-right area of the main PC board, very close to the Power and Video connectors. It too is mounted in a socket.
   A point to note about the Gate Array, is that unlike the other "chips" which may be purchased in any good Electronics parts store, the Array was designed and manufactured specifically for the CPC 464, and is only obtainable from AMSTRAD. Please ... proceed slowly.

Notice! all CPC 464 are not created equal. There are three possible component configurations for the main PC board, due to the two different Gate Array types, which AMSTRAD chose to use in the CPC 464.
The known possible configurations are:

A. Main board with one single socket for a Gate Array with heatsink (Early CPC models)

B. Main board with two optional places for a single Gate Array socket. Choice of the socket placement (position) was dependent upon the Gate Array used.

C. Main board with one single socket for a Gate Array without heatsink (Most recent model)

What all this means is that the vortex RAM Expansion card had to be designed so that it would install easily no matter what model (version) CPC 464 you had. Therefore it was necessary to provide each RAM card with two 40-pin Sockets, and three 40-pole parallel pin connectors.

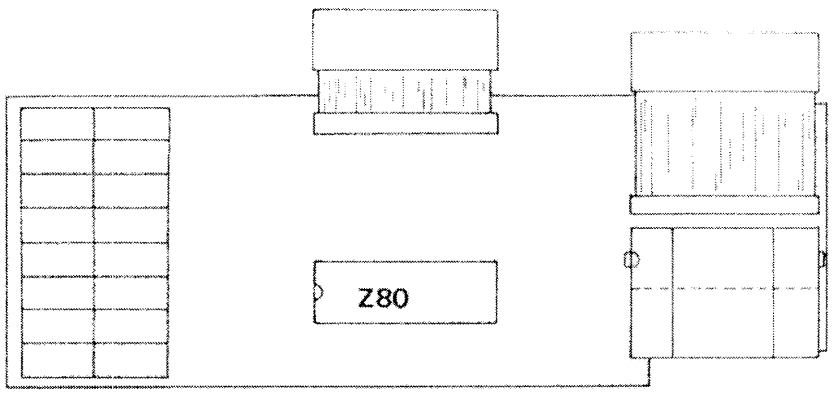
7. Locate the Z80 CPU (IC) on the main board. Insert the small "Flat-Tipped" screwdriver between it and it's socket, and gently wiggle it loose. Be very careful not to bend any of the "legs", or use excessive force. You may find a gentle "twist of the wrist" to be a great help.

8. In exactly the same manner, locate your Gate Array. Using an indelible marker mark it with an arrow pointing towards the rear of the console, so that there is no doubt in your mind how it must be re-inserted after its' removal. Using the same procedure as before, remove the Gate Array from it's socket. Remember to proceed carefully as this device is very difficult to replace.

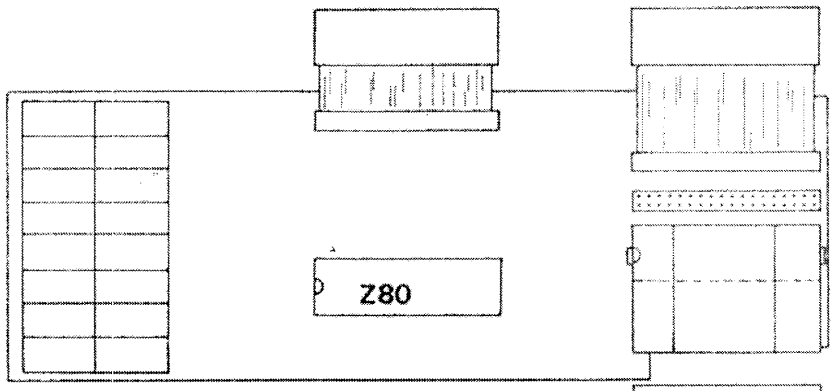
9. Now take out the black gummy cube located near the right lower edge on the PC board.

10. Now place the RAM Expansion on the towel in front of you, so that the ribbon connectors are to the rear. The ribbon connector on the right side is not soldered, but only pressed onto the 40-pole pin connector. Remove this connector now, and place it aside. In the middle of the RAM card, you'll see an empty socket. In this, as is pictured in Figs. 10, 11, and 12 you will insert the Z80 CPU IC. Be absolutely certain that all 40 "legs" are lined-up before pressing the chip onto the socket. Also note that the "half-moon" notch, which is located on one end of the chip, is correctly positioned. Check everything again before proceeding!

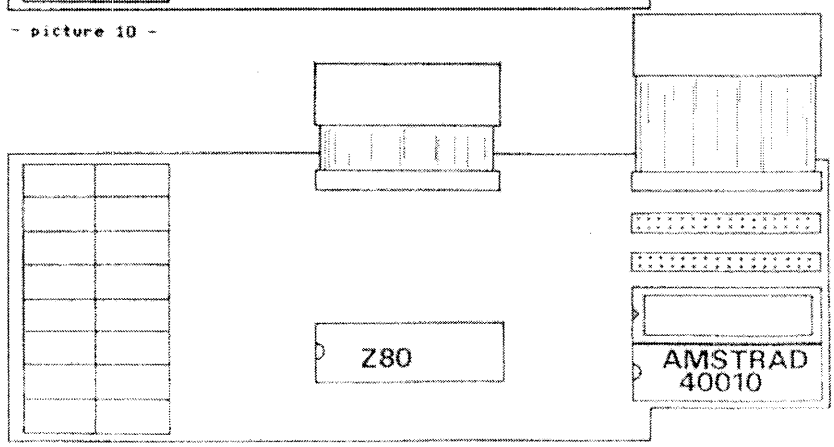
11. On the right side of the card, we will insert the Gate Array which you have previously marked. Now, dependent upon which type of Gate Array your CPC 464 was equipped with, you will either use the front or rear IC socket. If you have the Gate Array which uses no heatsink, then you will use the front socket as illustrated in picture 12. Be sure all 40 "legs" are aligned prior to pushing this chip onto its' socket.



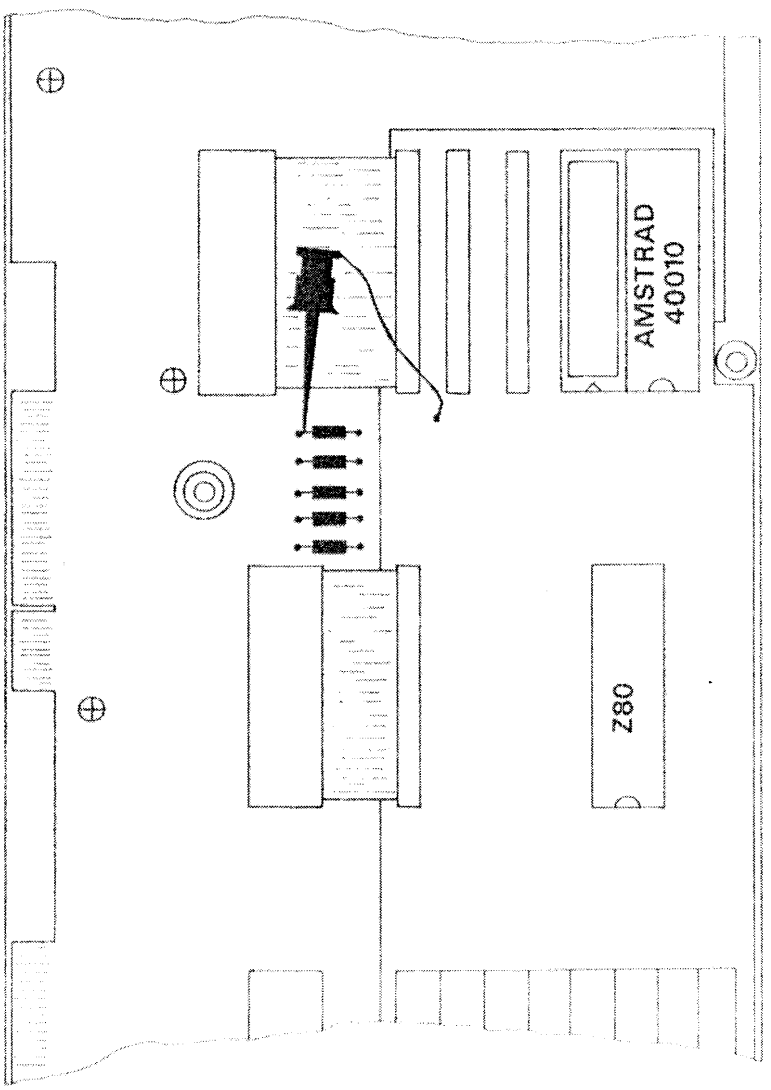
- picture 11 -



- picture 10 -



- picture 12 -



- picture 13 -

In the event your 464 is equipped with a heatsink, you must follow quite a different procedure. A Gate Array with a heatsink must be inserted into the rear IC socket as illustrated in pictures 10 and 11. Insure that the "half-moon" notch is correctly positioned before proceeding. If you've marked the IC with an arrow, then simply position it in the socket so that the arrow points towards the rear of the console. After inserting the Array, the heatsink must be replaced with that one which was packed with the RAM card. Proceed by slightly bending the clamps on the right and left side of the Array in an outward direction, so that the large square portion of the heatsink can be lifted out.
The white paste which you see is called heatsink compound, and must not be removed or wiped away. It is this compound which actually transfers the heat produced by the chip, to the aluminum for dissipation into the air. This activity "cools" the chip by continuously removing any excess heat. Once again, use this oppurtunity to insure that the "half-moon" notches are on the left side of the chip, and to insure that all 40 "legs" are properly positioned and seated in the socket. Now position the new heatsink on top of the Gate Array, and bend the clamps back to their original positions to hold it in place. Refer to pictures 10, 11 and 12 as needed, and don't proceed unless you're certain that everything up to this point is correct.

****
IMPORTANT: The "half-moon" notches on both the Z80 and the Gate Array ICs' must be properly positioned, that is as sit in front of the console and look down on the 464, the notches must be on the chips' left edges.
****

12. Now position the RAM card within your 464 console so that the center ribbon cable is aligned with the old Z80 socket on the main PC board as illustrated in picture 13. Carefully press the 40-pin ribbon connector onto the old socket.
Next, pick up the 40-pole pin/ribbon connector which was previously lain aside, and align the 40-pin (wide, flat) end of it with the old Gate Array socket on the main PC board. Carefully press it too, onto the old Gate Array socket. Double check everything up to this point. (IC pins, Ribbon Cable connections, etc.)
Now, adjust the RAM card so that its' front edge is parallel to that of the main PC board, and press the 40-pole ribbon/pin connector onto the appropriate set of 40 pins. Check your work.

13. All of the standing components (Capacitors) under the RAM card must now be gently lain on their sides. No need to worry, a Capacitor lying on its' side works as good as one standing up.

14. Now for the final connections. Attached to the RAM card is one Mini Test-Clips (Red or Black). Look again at picture 13, noting the position of the clip in relation to the 5 small resistors which are located in a row on the main PC board.
Note: Improperly connecting this clip will not hurt either the RAM card or your 464.

19. And last, but very important... Install the plastic sheet which was part of the RAM package, between the RAM card and the main PC board. This is to insure that no short circuiting can occur.

16. Now reconnect the upper and lower portions by reinstalling the keyboard cable harness as it was before. The power and diskette cable should likewise be reconnected.

17. Proceed by reading the special "What to do if it doesn't work" section, more for a self-check than anything else, rechecking every step one more time.

18. Carefully turn your CPC 464 over, and replace all 6 screws. Tighten each screw just enough to hold the console halves together. Do not overtighten!

19. Nothing further is needed for use with BASIC, however a new 62K8 System Disc must be created in order to use the card under CP/M. This task is handled by the cassette which is also a part of the RAM Expansion package.

20. When you're sure everything is in order, reconnect all cables and power-up. You should see the "RAM Sign On" in addition to the usual power-up screen.

What to do when it doesn't work ?

Is it plugged in?
Is the monitor turned on?
Is the red Power "On" indicator on the console lit?
Are the Power and Video connectors properly attached?

- Are you certain that you replaced both of the cable harnesses?
  Could you have neglected to press each fully in place?

- Are you certain that each of the 40-pin ICs' are fully inserted, that each pin is in its respective socket, that no pins are broken off or bent under, and that the half-moon notches are properly oriented to the left?

- Are you certain that the 40-pole pin connector is properly inserted, that no pin(s) is/are out of place bent or broken off? Look along the sides of the push on connector to insure that no pins are visible.

- Are you certain that each of the 40-pin ribbon connectors are fully inserted, that each pin is in its respective socket, that no pins are broken off or bent under?

- Is the plastic sheet properly positioned under the RAM board?

- Are you certain that you've properly connected the clip ?

When you can verify that all of the above are in order then

there is no reason the RAM card cannot operate as it is designed to within your system.

Creation of a CP/M workdisc

First of all, in order to be able to work with CP/M at all, you must be in posession of a disc drive. Although we most highly recommend the vortex Floppy Disc Station for your system, CP/M will work equally as well with the Amstrad 3" Disc drive. Should you not have either type Drive, then it isn't necessary for you to read the following paragraphs at all.

Let's assume that you are indeed the proud owner of some type Disc Drive system, and that you also have the CP/M system discette on hand, which you recieved with the disc station. In addition, you also recieved a number of useful utilities with your station, several of which we'll need to get started.
What we need to do first, is to create a "new" CP/M system, which will allow you full use of your new RAM Expansion card. The "new" here, is merely the modification of tracks 0 and 1 (System Tracks), which is totally automatic, and handled by a program utility called PATCH.COM.

In order for you the customer to be able to do that what is required, we wrote the PATCH program, so that it would perform the required system track changes, without alot of user interaction.

PATCH will "modify the system tracks on one of your discs, and transfer both the SPOOL and RAMDISK programs onto this disc, along with a new image file $OSC.SYS which contains a portion of the Operating system.

The program RAMDISK does to the Expansion card as the FORMAT program does to your discs... It prepares them to recieve blocks of data.

The SPOOL program causes a portion of your new RAM Expansion to function as a printer-buffer, thereby allowing you to continue typing, formatting, programming or whatever you like. The SPOOL program takes care of your printer's feedings by storing the entire file (temporarily) and then "strobing" it to the printer as it is needed. When the entire file has been sent, the SPOOL program once again steps into the background and remains there until you need to print again.

SPOOL is activated by typing A>SPOOL <CR>, and deactivated by typing it again.

The steps which must be taken in order to complete the installation are as follows:

1. Backup your "Old" System discette (FORMAT a blank disc, and then COPY it; DISCCOPY it in the case of 3" AMSTRAD Disc Station)

2. Use the CASCOPY program to transfer the PATCH program from

the cassette to disc. (CLOAD in the case of 3" AMSTRAD Disc Station)

3. Run the PATCH program. This will initialize your RAM Expansion for CP/M

4. Use MOVCPM and SYSGEN to generate a 62KB CP/M System

NOTE: After completion of this exercise, you will have created a new 62KB CP/M System discette which will allow you to work with every Standard CP/M program available, a 32KB printer-buffer which takes charge of your print operations and leaves the CPU free to do all the other tasks, and a RAM disc of up to 448KB, which is much faster then your present "Disc-to-Disc" operations because it has no moving parts, delays, wait-states, or timing bugs.

The new System (62KB) discette should immediately be backed-up, and the original write-protected, and put in a secure place.

Here's the entire process, step-by-step.

CPC 464 + vortex 5.25" F1-D/S	CPC 464 + AMSTRAD 3" DDI-1
I. Creating a copy of the Original System Discette	
A. Format a new, empty 5.25" Discette using the BASIC command |FORMAT,1 <CR>. (see Picture #1)	A. Format and copy the Original System discette under CP/M to a new 3" discette. Put Original System discette into drive A and start CP/M by typing |CPM <CR>. If you own a single drive start DISCCOPY <CR>, to format the new disc and copy the Original System discette. (see Picture #2).
B. Insert the Original System discette in drive A an type the command |CPM <CR> to start the CP/M Operating System. (see Picture #1).	
C. Copy the Original System discette to the already formatted discette by using the Program COPY <CR>. (see Picture #3). If you own the F1-S the Original and the Copy are exchanged until the whole discette is copied. If you own the F1-D, put Original into drive A an blank, new discette in drive B.	Original and Copy are exchanged until the whole discette it copied. If you own two drives, put Original Discette in drive A an blank, new discette in drive B an start under CP/M COPYDISC <CR>. NOTE: the copy must have the so called 'System-format.'
D. Return the Original System discette to a safe place.	B. Return the Original System discette to a safe place
II. Copying PATCH.COM from cassette to discette	
A. Insert the enclosed cassette into the datasette recorder and rewind it. Place	A. Insert the enclosed cassette into the datasette recorder and rewind it. Place

```
 BASIC 1.0

Ready
|FORMAT,1

Drive A: diskette inserted ? (Y/N) Y
Init side 1 track 79
Ready
|CPM
```

- picture 1 -

```
CP/M 2.2 - Amstrad Consumer Electronics plc

A>DISCCOPY

DISCCOPY V2.0

Please insert source disc into drive A then press any key:
Copying started
Please insert destination disc into drive A then press any key:
```

- picture 2 -

```
 44K CP/M vers. 2.2-01/85   vortex GmbH

A>COPY

 COPY 2.0   (C)1985 vortex GmbH

Source diskette in drive A or B  ? A
Destination diskette in drive A or B ? A

Please insert Source diskette into drive A then press any key
Reading     - side  0 track 07
Please insert Destination diskette into drive A then press any key
Writing     - side  0 track 07
Please insert Source diskette into drive A then press any key
```

- picture 3 -

```
A>CASCOPY

 CASCOPY 2.0    (C)1985 vortex GmbH

Select Copy-direction (<,>):
Diskette < Cassette

Name of Cassette file:
Name of Diskette file:
Press PLAY then any key:
Loading PATCH.COM block 9

A>
```

- picture 4 -

```
A>CLOAD

CLOAD V2.0

Press PLAY then any key:
Loading PATCH.COM block 9


CLOAD V2.0 finished

A>
```

- picture 5 -

```
A>PATCH

 CP/M Initialization for the vortex RAM Expansions  (C)1985 vortex

 (Use ^C to leave this program)

 Printer-Spooler activated at boot ?          (Y)es or (N)o: Y
 RAM disc formatted automatically at boot ?   (Y)es or (N)o: N
 System vector active ?                       (Y)es or (N)o: Y
 CP/M Restart now ?                           (Y)es or (N)o: Y
```

- picture 6 -

```
 44K CP/M vers. 2.2-09/85    vortex GmbH
 (Spooler on - RAM disc not formatted)

A>MOVCPM 250 *

CONSTRUCTING 62k CP/M vers 2.2
READY FOR "SYSGEN" OR
"SAVE.34 CPM62.COM"
A>
```

- picture 7 -

```
A>SYSGEN

 SYSGEN 2.0     (C)1985 vortex GmbH

Source diskette in drive A,B or RETURN ?
Destination diskette in drive A,B or RETURN ? A
Please insert Destination diskette into drive A then press any key

Another diskette Y/N ? N
Please insert a CP/M System diskette into drive A then press any key

A>
```

- picture 8 -

```
A>SYSGEN *

SYSGEN V2.0

Please insert DESTINATION disc into drive A then press any key:_

Do you wish to reconfigure another disc (Y/N) ? : N

Please insert a CP/M system disc into drive A then press any key:_

SYSGEN V2.0 finished

A>
```

- picture 9 -

the Copy of the Original System discette into drive A B. Start by typing CASCOPY <CR> the cassette to discette Copy program. (see Picture #6)	the Copy of the Original System discette into drive A B. Make room for the new programs on this discette Erase all but SYSGEN.COM, MOVCPM.COM and CLOAD.COM. Use for this the CP/M built-in command ERA. C. Start by typing CLOAD <CR> the cassette to discette Copy program. (see Picture #5)

III. Using the PATCH.COM program

The use of this program is identical for either vortex 5.25" Station or the Amstrad 3" Station. We should still be in CP/M mode, so let us continue by typing PATCH <CR>, this is the programm we transferred from cassette to discette. The program will load and start itself by asking you a few questions. (see Picture #6). Detailed information about these questions are given you later.

The first question:
Printer-Spooler acivated at boot ? (Y)es or (N)o
we will answer with a 'Y' for Yes

The second question:
RAM disc formatted automatically at boot ? (Y)es or (N)o
we will answer with a 'N' for No

The third question:
System vector active ? (Y)es or (N)o
we will answer with a 'Y' for Yes

The fourth question ?
CP/M Restart now ? (Y)es or (N)o
we will answer with a 'Y' for Yes

After you input your last (fourth) question, the CRT will clear the System will re-boot and then the new Boot-message will appear to tell you that the Spooler is active and the RAM disc is formatted. (see Picture #7)

By typing DIR <CR> we look at the directory of this newly created discette. You must see the three new files $OSC.SYS, RAMDISK.COM and SPOOL.COM.

Now with all that out of the way, you are ready to generate the new 62K CP/M System discette.

IV. Installing the 62K CP/M 2.2 Operating System

A. We'll begin by typing MOVCPM 250 * <CR> which is the command to re-size the Operating system to it's maximum of 62K. (see Picture #7)	A. We'll begin by typing MOVCPM 250 * <CR> which is the command to re-size the Operating system to it's maximum of 62K. (see Picture #7)
B. Next type SYSGEN <CR>	B. Next type SYSGEN * <CR>

! which puts the newly "resized" CP/M which is now in memory onto tracks 0 an 1 of the diskette in drive A. (see Picture #8). The question: Source diskette in drive A,B or RETURN ? We'll answer with pressing the ENTER key. The question: Destination diskette in drive A,B or RETURN ? We'll answer with an 'A'. When directed: Please insert Destination diskette into drive A then press any key Simply press the ENTER key. The question: Another diskette Y/N ? We'll answer with a 'N' When directed: Please insert a CP/M System diskette into drive A then press any key Simply press the ENTER key.

C. Now press CTRL+SHIFT+ESC. This will cause a System reset to occur.

D. Restart CP/M from BASIC by typing |CPM <CR>. When the CP/M Boot message appears you should see '62K' instead of '64K' in the header. This is the resized CP/M which you created with the commands PATCH, MOVCPM and SYSGEN.

! which puts the newly "resized" CP/M which is now in memory onto tracks 0 and 3 of the diskette in drive A. (see Picture #9). When directed: Please insert DESTINATION disc into drive A then press any key: Simply press the ENTER key The question: Do you wish to reconfigure another disc (Y/N) ? We'll answer with a 'N' . When directed: Please insert a CP/M system disc into drive A then press any key: Simply press the ENTER key.

C. Now press CTRL+SHIFT+ESC. This will cause a System reset to occur.

D. Restart CP/M from BASIC by typing |CPM <CR>. When the CP/M Boot message appears you should see '62K' instead of '64K' in the header. This is the resized CP/M which you created with the commands PATCH, MOVCPM and SYSGEN.

All of the required work is now complete, and you have a new '62K' CP/M System disc, which is completely compatible with your new RAM Expansion, and the utility programs RAMDISK, SPOOL, and $OSC.SYS. Now all that's left to do is to write protect this new disc, and use it as a 'Master' to generate all of your future "BIG CP/M" diskette requirements.

Printer-Buffer, Booting, RAM disc, System vector ??

In order to get right to the 'meat of the matter', we skipped over detailed into an explanation for all of the 'strange, new words' used here. Instead, we concentrated merely on the generation of a new, modified, bigger CP/M, and the steps required to accomplish that task. Now for the definitions:

Printer-Buffer

You've no-doubt noticed already, that it's impossible to use the console for anything while you're printing. That stems from the fact that the CPC 464 inventors made no provision for simultaneous printing while anything else was going on. This would require extensive programming, timing changes, and a bigger memory. Well, thanks to your new RAM Expansion card, the instant you give the 'printout' commands, your file is transferred to an 'intelligent' area of memory (32KB long), and the card takes over from there. You are free to do whatever you wish (Load, Save, Type, Copy, Erase, etc.). The 32KB area holds the equivalent of 16 DIN A4 Typewritten Pages, which we felt would be more than sufficient for your needs. This 32KB Buffer is activated by typing SPOOL<CR>, and de-activated by the same command. It is totally invisible to the operator.

Remember, with SPOOL active, all output will go to the printer unless you type CTRL+P after booting. (CTRL+P turns printout function on and off)

Booting

Whenever the word 'Boot', 'Booting', or the like appear, it means simply the loading or re-loading of CP/M into the computer's memory. A Boot occurs whenever the command |CPM <CR> is typed in (from BASIC). This causes the contents of tracks 0 and 1 to be read into the CPC 464 and started. From that point on, the computer is under control of the CP/M operating system. Should you desire to change discs, always type CTRL+C so the CP/M BDOS knows. This will prevent any data loss should you try to Save, Erase or Rename a file.

RAM disc

The RAM disc is really only a 'Pseudo Disc Drive'. A disc drive as we know it, consists of mechanical parts (motor, read/write heads, load solenoid, stepper motors, etc.) and electronics (controller, write/sense amplifiers, servos, etc).
In order to read or write even 1 bit of data, the drive must first be activated, the File control Block must be polled to find the next free spot on the disc, the stepper motors must position the read/write heads over the track to be written, the solenoids must load the heads to the disc's surface, the index and sector timing circuits must determine when to start reading/writing the data, the FCB must be updated, and the new filename entered into the directory. As you can see, this takes up alot of time. The actual data read/write operation may only take a few milli-seconds, but the 'red tape' can sometimes make working with even a very fast drive very tiring.
This problem is eliminated thru use of our RAM Expansion card which functions as a Disc drive, but without any mechanical or electronic parts. Better yet, as it has no 'wait states', timing or head positioning/loading delays, it is up to 50 times faster than a conventional drive. Under CP/M, it is defined as drive 'C', and like his electro-mechanical brothers must be formatted prior to use. Of course you must remember to transfer all data on 'C:' to a real disc before shutting off your console, as it will otherwise be lost.

To use the RAM disc most effectively, the RAMDISK program must first be used to format the 'Disc' for data and file storage. Then, PIP should be used to transfer all related files from A: or B:, to C:. This only takes a matter of seconds due to reasons mentioned above. You are now free to do any file manipulation required without any disc drive interruptions at all.

As the RAM disc holds the equivalent of 224 DIN A4 Typewritten pages (448KB) it has more than enough room for a large database, general file or wordprocessor.

When asked: 'RAM disc format at boot' ? (Y)es or (N)o
You must decide if you want the RAM disc to be re-formatted (All data lost) every time a Boot or return to BASIC (WPOS) is executed. We answered (N)o here, because we would rather not lose all 448KB of data just because we accidentally or intentionally returned to BASIC. The command 'RAMDISK (CR)' can be typed at any time, and is handy for 'clearing drive C:' once you've transferred your work to A: or B:.
IMPORTANT: A RAM disc is available with the RAM Expansion cards SP128(=64KB), SP256(=192KB), SP320(=256KB) and SP512(=448KB).
(in brackets we have the formatted capacities of the Ram disc's)

^System vector^

Your original CP/M system was only 44KB therefore all of the CP/M utilities (DDT, STAT, DUMP, etc...) were written to most effectively use the available memory, while allowing you the largest TPA (Transient Program Area) possible.
In order to satisfy these needs, many of the utility programs use what is called a "jump table", that is a specially designed area of memory, where addresses to system and ROM calls are stored. This was usually right at the top of the TPA which used to end at 44KB. Now that you've increased your system to a whopping 62KB, the jump tables are no longer at the top of TPA, which is now farther up in memory (62KB), but rather right in the middle of it (at the 44KB mark). It is easy to see how a jump to top of TPA, which remains un-modified, can result in system 'lock-up', crash or god forbid, even a Drive head load and un-controlled write operation! With literally any data residing at the old Top of TPA (44KB) a call or jump to this area could be disasterous. The System vector is simply another jump table, which 'redirects traffic' so to speak, and causes all system jumps to 'land' on their proper mark.

When asked: 'System vector active' ? (Y)es or (N)o
You have to decide if you want to use the CP/M utilities under the 62KB system. You may ask here, 'What good does it do not to be able to use utility programs under 62KB CP/M ?' Well, due to overlapping, and the fact that a jump table is now located within the TPA, especially the program DDT and the built-in command SAVE can cause some trouble if the program you manipulated with DDT is overlapping the System vector, which means that a part of this program is replaced with the System vector. Programs up to a length of 40KB are not affected of this problem.
For normal CP/M programs and Applications it is always better to use the vector.

The Expanded BASIC commands of the vortex RAM-Expansion

The vortex RAM-Expansion gives you increased RAM memory storage area for both Data and Program storage. Even with our smallest card, you get a full 64KB RAM for your programs, and 32KB storage for Data. Our biggest (512KB) offers you a full 288KB for programs, and an incredible 256KB for data. Truly enough for any requirement.

The addition of 37 new BASIC commands enables you to get the absolute most out of this new peripheral, without the slightest difficulty when used in BASIC.

The program memory area is arranged to allow you to write 9 32KB blocks of BASIC code and to put them together any way you desire. Indeed, it is possible to use the same line numbers in every 32KB bank, as the line numbers are additionally identified by the bank to which they belong. An example of your BASIC code might look like this: 120 GOSUB LINE 700, BANK 4
or: 320 IF A<>0 THEN GOTO LINE 5, BANK 3 ELSE LINE 5, BANK 5

Your program may occupy one or more banks, or any portion of any bank. That is up to you the programmer.

Specifications of the RAM-Expansion under BASIC

(1) 64...288KB Program memory area

(2) 32...256KB Data memory area

(3) 32KB Printer-Buffer area

The Printer-Buffer is turned on and off using the following commands:

SPOOL ON<CR> turns the Buffer on
SPOOL OFF<CR> turns the Buffer off

When 'ON' the Buffer is sensitive to any 'list' commands appearing on the console, and causes the output to go to the printing device. Like the SPOOL.COM function (Under CP/M) the Buffer remains invisible to the user. The command SPOOL OFF turns the buffer off.

The Data memory can function either as a storage area for Video RAM or as a "Psuedo Floppy"

In Video RAM mode, you are able to store the contents of up to 17 CRTs full of data, and play them back at up to 3 CRTs full per second. As each 'CRT full' is equal to 16KB of data, the effect is kind of like a 'penny movie'

In 'Pseudo-Floppy' mode, you have up to 256KB free memory space in which to put your relative RAM file.

THE EXPANDED BASIC COMMAND SET

BANK
BASIC
BOS
CALL
COMMON
DEV
FAST
FRAME
GCHAR
GOSUB
GOTO
GPAPER
GPEN
ID
LIST
LOAD
MASK
MON
NEW
PEEK
POKE
RAMCLOSE
RAMFIELD
RAMOPEN
RAMREAD
RAMWRITE
RECORDS
RETURN
RUN
SAVE
SCREEN.IN
SCREEN.OUT
SCREENS
SLOW
SPOOL.ON
SPOOL.OFF
UNMASK
VIDEO.ON
VIDEO.OFF

IMPORTANT:

Working under BOS 1.0 all characters after ASCII 32 are user-defined (equal to SYMBOL AFTER 32). Therefore the SYMBOL AFTER command is redundant. If you nevertheless use the SYMBOL AFTER command in programs running under BOS 1.0 you'll receive an error message.
All following BASIC commands up to section "THE ROM RESIDENT Z-80 MONITOR" are only working under BOS 1.0 (started with |BOS <CR>).

THE VORTEX BANK-BASIC EXPANSION

All of the vortex BASIC command Expansions begin with a vertical line '|' which is derived from a simultaneous pressing of the SHIFT key and the key with the circled a '@'.
The following abbreviations are used for the BASIC command explanations which appear in the remainder of this handbook:

l,m,n,o	Represents the number of a Program memory Bank. This can be any number between 0 and 8.
line	Line number of a BASIC statement within a program
addr	Address of byte in memory
parl...	Representation of a parameter to be transferred by a CALL
chn	Output channel #; chn=0 Is the DEFAULT channel (CRT output) chn=8 Is the PRINTER channel (Hardcopy output) chn=9 Is the FLOPPY/CASSETTE channel (Disc/Tape output)
i	Video RAM number. Can be any number between 0 and 15, depending on the size of your RAM- Exp. card.
string	Represents a string variable. This cannot be directly transferred, rather handled in the form @VAR$. Remember that VAR$ must have been previously defined.
reclng	Length of a data record
len1...	Represents the length of a variable in a RAMFIELD definition
n	Represents any valid number depending of the command being used
recnum	Represents Data or Record number
/.../	Represents user optional inputs
<CR>	Means press the 'ENTER' key

BANK

Format: |BANK,n

Function: Selects on of the available 32KB program memory banks (0...8)

Comments: The command |BANK,n can only be used in direct mode, and allows the programmer to switch between banks for program input or correction. The line numbers typed in at any one time, belong to the bank which is momentarily active.

The number of available banks per card type is shown below:

Type	Number of 32K Banks	Total available Program memory
SP64/M	2	64 KByte
SP64	2	64 KByte
SP128	3	96 KByte
SP256	5	160 KByte
SP320	6	192 KByte
SP512	9	288 KByte

After installation of our vortex RAM-Expansion card, the built-in RAM of your CPC 464 becomes Bank 0, while all of the other Banks (1-8) reside on the card itself.

IMPORTANT: The basic variable HIMEM is equal to 32616 when used with Bank 0, and 32767 when used with all other Banks (1-8). This value must not be increased thru use of the MEMORY command, but may be reduced without any trouble. Under certain conditions, HIMEM for Bank 0 can also be lower. (See the |VIDEO.ON command)

Example:
```
*** Bank 0 Active ***

Ready
|BANK,1<CR>

*** Bank 1 Active ***

Ready
```

BASIC

Format: |BASIC

Function: Turns off Bank orientated BASIC

Comments: Returns the BASIC interpreter to Non-Bank BASIC status. The entire system assumes 'power-up' condition. This command reverses the effects of the |BOS command.

Example: |BASIC<CR>

BOS

Format: |BOS

Function: Turns on Bank orientated BASIC

Comments: This command allows the use of the new BASIC statements. Activating the |BOS command causes all memory to be erased, therefore you should

always be sure that no revalant data is in memory before activating BANK-BASIC. The sign-on for BOS is:

--- Bank-BASIC BOS 1.0 (C)1985 by vortex ---

Next, the RAM-Expansion will be recognized, and the following information will appear:

```
+-----------------------------------------+
! vortex      RAM-Expansion      Card  ID ! 
+-----------------------------------------+
!    w          xxxK     yyyK        zzK ! 
!  Bks          Prgm     Data        Spl ! 
+-----------------------------------------+
```

w represents the number of available 32KB program memory banks (Bks = Banks)

xxx Represents the total available RAM for program storage

yyy Represents the total available RAM for Data storage

zz Represents the Printer-Buffer size in KBytes. This can either be 0KB or 32KB. The Data RAM will increase or decrease accordingly.

Example:

```
Ready
|BOS<CR>
```

CALL

Format: |CALL,n,addr/,par1,par2.../

Function: Used to call a machine code subprogram which is located in Bank n, at address addr. Additionally, the transfer of variables is supported (par1,par2...).

Comment: Use of this option should be attempted only by experienced machine code programmers, with a good understanding of the CPC 464's memory structure. Up to 30 parameters may be transferred with one |CALL.

Example: 20 |CALL,1,&6000

COMMON

Format: |COMMON/,string,1,m,n,.../

Function: Allows REAL, INTEGER, or STRING varibles beginning with a certain alphanumeric character to be defined as Local or Global or to be defined to have a certain value only on specified banks of memory.

IMPORTANT: Fields are always local defined which means only defined on one bank. To transfer field elements to other banks, you must assign them first to a STRING-, REAL- or INTEGER variable that is of the COMMON type.

Comments: With this command it is possible to assign groups of variables to each separate Bank of memory, simply because they'll share the same alphabetic character at the beginning of their names.
Likewise, a group can be assigned to work with only one Bank, two Banks, three Banks etc...

In the event only |COMMON is input then all variables defined in this bank are only valid in this bank. (Local variables).
It is therefore possible to define the variable 'HUGO$' in each bank with different values for 'HUGO$' in each definition.

|COMMON,@string$ defines a group of variables as valid for use in each (!) bank. The group criteria will be the beginning alphabetic character which was assigned to the variable string$.

The structure of the variable string$ is:

string$="b1,b2,b3-b4,b5,b6-b7..."

In this b1,b2,.. stands for the first character of the different groups of variables.

|COMMON,@string$,1,m,n,.. defines the groups of variables valid in banks 1,m,n,..

IMPORTANT: The speed of variable transfer from bank to bank depends on the number of COMMEN variables.

Example: a.) 10 |COMMON

= all variables are local defined

b.) 100 A$="B,K-P,Z"

```
                    110 |COMMON,@A$
```

- all variables beginning with a B, Z or K-P are defined in all banks

```
c.)         100 A$="B,K-P,Z"
            110 |COMMON,@A$,0,2
```

- all variables beginning with a B, Z or K-P are only defined in banks 0 and 2

For example the variable BETA has the value 123.456 on bank 0, it also has this value on bank 2. The variable BETA will not have the same value on every other bank.

DEV

Format: |DEV/,chn/

Function: Assigns an output channel for the |LIST command.

Comment: The output device can be one of the following:

chn=0 CRT (Video output)
chn=8 Printer (Hard copy device)
chn=9 Disc drive/cassette

Inputting the command |DEV by itself selects the CRT (Channel 0) as the output device. (This is the default channel)
Once selected, the channel remains active until the next |DEV command or system reset.

Example: 100 |DEV,8<CR>

GOSUB

Format: |GOSUB/,n/,line

Function: This command is used to call a subroutine in a specified bank.

Comment: With the command |GOSUB it is possible to call a subroutine which starts at location line, bank n, from within a BASIC programm, which is running in the current bank. At the end of this subroutine must stand the |RETURN command to return to the line after the |GOSUB command.
line can be constant or (!) variabel.

Example:
```
a.)         20 |GOSUB,1,125
```

```
b.)     10 INPUT A
        20 |GOSUB,1,A

- depending on A different subroutines in bank 1
can be called.
```

GOTO

Format: |GOTO/,n/,line

Function: This command is used to jump to a specified line within a specified bank.

Comment: With the command |GOTO it is possible to execute a nonreturnable jump to a specified line in another bank.
line can be constant or (!) variable.

Example:
```
a.)     10 REM
        20 |GOTO,2,1985

b.)     10 line=1000
        20 |GOTO,4,line
```

ID

Format: |ID

Function: This command is used to display the "memory statistics" of the vortex RAM-Expansion.

Comment: See ID (Identification) by the command |BOS.

Example: 30 |ID

LIST

Format: |LIST/,1,m,n,o,.../

Function: Lists a program bank by bank.

Comment: With this command, it is possible to list selected banks or all banks. If no bank is specified, then all banks containing program lines are listed.
To list program lines in one bank you also can use the "normal" LIST command.
The command |DEV can be used to specified where the list output should appear.

Example:

```
*** Bank 0 active ***

Ready
1  A$="A"
5  |COMMON,@A$
10 REM >> this programm resides in bank 0 <<
20 For A=1 TO 100
30 |GOSUB,1,20
40 NEXT A
|BANK,1

*** Bank 1 active ***

Ready
10 REM >> this programm resides in bank 1 <<
20 PRINT "The square of ";A;" is ",A^A
30 |RETURN


Ready
|LIST

------------------ BANK 0 ------------------

1  A$="A"
5  |COMMON,@A$
10 REM >>this programm resides in Bank 0<<
20 FOR A=1 To 100
30 |GOSUB,1,20
40 NEXT A

------------------BANK 1 ------------------

10 REM >> this programm resides in bank 1 <<
20 Print "The square  of ";A;" is ",A^A

*** Bank 1 active ***

Ready
```

****
*** LOAD ***
****

Format:

```
|LOAD,name
```

Function: This programm allows you to load a programme previously saved whith the |SAVE command. This command is used to save programs which occupy more than 1 bank.

Comment: Assume a programm called "TEST" is occupying three banks of memory. In order to save the programm, you must use the command |SAVE. Afterwards, looking at your disc you will see that your programm has been saved under four different file names.
(TEST.BAS, TEST.BK0, TEST.BK1, TEST.BK2)

In order to reload the above program, which occupied more than 1 bank, you must use the |LOAD command. The file TEST.BAS is a 'so-called' loader for the other three files. It insures that each is loaded to the proper bank. (BK0=Bank 0, etc...)
The files with the type BKX can also be loaded with the "normal" LOAD command. In this case you must guarantee that each program part is loaded into the right bank.

Example:

```
10 A$="TEST"
20 |LOAD,@A$
```

NEW

Format: |NEW/,1,m,m,o.../

Function: Erases selected/all banks of memory

Comment: This command is used to erase contents of a specified bank or all banks of memory.

IMPORTANT: Once you have used the command |GOSUB, and now choose to write a new programm, you must (!) use this command to erase all banks of current memory, before proceeding.

Example: |NEW,1,3

- erase banks 1 and 3. Other banks are not erased

****
* PEEK *
****

Format: |PEEK,n,addr,@var

Function: This command assigns the value of the memory address specified by addr, bank n, to the integer variable var.

Comment: With this command it is possible to examine the contents of memory bank n, even though you are presently in another bank.

Example: |PEEK,2,&4000,@A

****
* POKE *
****

Format: |POKE,n,addr,amount

Function: This command assigns the value amount to the address addr in bank n.

Comment: With this command it is possible to change contents of specified memory location(s) of bank n even though you are presently in anther bank.

Example: |POKE,2,&6800,&FF

****
* RAMCLOSE *
****

Format: |RAMCLOSE

Function: This command is used to close the relative data file which was opened with the |RAMOPEN command.

Comment: The data storage on the RAM Expansion card can be setup to use as a relative file. This is a file consisting of several so called Records of a specified length. A record can be selected through its record number.
Compared with normal sequential file management this method brings much more speed in file I/O.
The previously with |RAMOPEN opened file must be closed with |RAMCLOSE. If you forget this, the data storage remains as relative RAM file an the group of the |VIDEO commands are not accessible.

CAUTION: Do not forget to save your data from the relative RAM file to disc or cassette, before activating the |VIDEO commands. If you forget this all data are erased !
More commands are |RAMOPEN, |RAMFIELD, |RAMREAD, |RAMWRITE and |RECORDS

Example:

```
10 |RAMOPEN,128

90 |RAMCLOSE
```

|RAMFIELD

Format: |RAMFIELD,len1,len2...

Function: Separation of records into fields of differing lengths. This command is used in conjunction with the other |RAM commands

Comment: This command is used to structure a record into fields, and must appear immediately after the |RAMOPEN command in the program. A record having a length for example of 256Bytes can be devided for example in three fields with len1=30 len2=40 and len3=20.
The commands |RAMREAD and |RAMWRITE uses in this case three variables correaponding to the three fields of a record.
Of course the sum of all fieldlength cannot be longer than the length of one record.

Example:

```
10 |RAMOPEN,128
20 |RAMFIELD,32,32,64
30 .....
40 .....
50 |RAMCLOSE
```

|RAMOPEN

Format: |RAMOPEN,recln

Function: This command defines RAM as a Data-File storage area. The variable recln can be any value between 8 and 1048. Using the |RAMFIELD command the record can be devided into several fields of different or equal length.

Comment: This is the first command to be input whenever you desire to establish a Relative length Data-File in the Expansion RAM.
The next command to be input, must (!) be the |RAMFIELD command which further defines the record length into fields of specific length.
The total of all specified fields must not exceed the total record length originally declared
Remember... it is not possible to use the Data-File and VIDEO RAM features at the same time

Example:

```
10 |RAMOPEN,100
20 |RAMFIELD,10,5,25,45,15 ..(Total)=100)
30 |RAMWRITE...
40 ......
50 ......
60 |RAMCLOSE
```

RAMREAD

Format:

```
|RAMREAD,recnum,@var1$,@var2$...
```

Function: This command is used to read the contents of all or selected fields of a specified record (recnum) into the variables var1$, var2$,...

Comment: The record is selected by its number recnum. recnum is allowed between 0 an REC-1. REC is a System variable and is calculated with the command |RECORDS. |RECORDS must stand after the |RAMOPEN command. REC depends on the RAM capacity of your RAM-Expansion card and recing. It represents the maximum of available records.

IMPORTANT: var1, var2, .. must (!) be of the string ('$') type and must be defined as 'dummies' before first use (var1$="", var2$="").

Example:

```
 5 u$="":v$="":x$="":y$="":z$=""
10 |RAMOPEN,100
20 |RAMFIELD,5,45,25,15,10

40 |RAMREAD,19,@u$,@v$,@x$,@y$,@z$
50 ....
90 |RAMCLOSE
```

RAMWRITE

Format:

```
|RAMWRITE,recnum,@var1$,@var2$,...
```

Function: Writes the Data assigned to string variables var1$, var2$, .. into fields 1, 2, .. of record number (recnum).

Comment: The contents of the field variables var1$ var2$, .. is written into the corresponding fields of the record assigned by recnum. recnum is allowed between 0 an REC-1. REC is a System variable and is calculated with the command |RECORDS. |RECORDS must stand after the |RAMOPEN command. REC depends on the RAM capacity of your RAM-Expansion card and recing. It represents the maximum of available records.

IMPORTANT: var1, var2, ... must (!) be of the string ('$') type and must be defined as 'dummies' before first use (var1$="", var2$="")

Example: 100 |RAMWRITE,20,@A$,@BB,@XC$

RECORDS

Format: |RECORDS

Function: Used after |RAMOPEN command. This command calculates the total number of possible records, and puts the result into a System variable 'REC'

Comment: The value of 'REC' is dependent on the size (in KB) of your RAM-Exp. card, and the size of the record specified in your |RAMOPEN command.

Example:
```
10 |RAMOPEN,128
20 |RECORDS
30 Print"The maximum of records is ";REC
40 .....
50 .....
90 |RAMCLOSE
```

RETURN

Format: |RETURN

Function: This command is used to return to the calling program, after execution of a |GOSUB.

Comment: This command must (!) appear as the last command in a subroutine, which was called by the |GOSUB,n,line command.
The point of return is the line immediately following the |GOSUB command in the calling BASIC program.

IMPORTANT: The "normal" RETURN command (without a |) has no function in connection with the |GOSUB command.

Example: 45 |RETURN ... Program execution will return to calling program

****
* RUN *
****

Format: |RUN,<string/,n,line/

Function: Loads and Runs a Bank-Orientated-Basic program

Comment: A program SAVEd with the |SAVE command, is stored in several parts on the discette. The |LOAD command only loads the program, while the |RUN loads and starts it. Alternatively, you can specify which bank (n), and/or which BASIC line (line) the program execution is to start.
(See the |LOAD, and |SAVE command explanations)

Example: A$="PROGRAM"
|RUN,@A$... Loads and Runs the program "PROGRAM"

****
* SAVE *
****

Format: |SAVE,<name/,l,m,n,o,.../

Function: Saves a BASIC program which is occupying more than 1 Bank of memory.

Comment: With the help of this command, it is possible to save the contents of several Banks of memory in one process to Cassette or Discette.
For example we have the programm TEST which occupies three program memory banks. After using the |SAVE command you'll find on disc/cassette four files
TEST.BAS TEST.BK0 TEST.BK1 TEST.BK2
The file TEST.BAS is the so called "loader" for the other three files. It insures that each is loaded to the proper bank.
The options l,m,n,o are used to save only some special program memory banks.
(also see the |LOAD command explanation)

Example: A$="RAPIDROY"
|SAVE,@A$... Saves the program "RAPIDROY" which occupied more than 1 Bank of memory

****
* SCREEN.IN *
****

Format: |SCREEN.IN,1

Function: Loading an entire 16KB video RAM "Picture" out of VIDEO RAM

Comment: Before using this command you have to set up the Data storage to VIDEO RAM. Do this with the command |VIDEO.ON.
The |SCREEN.IN command reads in 0.3sec an entire 16KB Video RAM "Picture". i defines the "Picture" number and is allowed between 0 and SCR. SCR is a system variable which is calculated with the |SCREENS command. SCR depends on the size of you RAM-Expansion card.
In the case of the SP512 card SCR is 16.

Example:
```
10 |VIDEO.ON
20 FOR i=1 TO 6
30 |SCREEN.IN,i
40 Next i.... This routine calls out and
              displays 6 VIDEO RAM 'Pictures'
              at three pics' per second speed
```

SCREEN.OUT

Format: |SCREEN.OUT,i

Function: Stores an entire 16KB VIDEO 'Picture' to VIDEO RAM

Comment: Before using this command you have to set up the Data storage to VIDEO RAM. Do this with the command |VIDEO.ON.
This command stores an entire 16KB VIDEO "Picture" to the VIDEO "bank" assigned by i.
i defines the "Picture" number and is allowed between 0 and SCR. SCR is a system variable which is calculated with the |SCREENS command. SCR depends on the size of you RAM-Expansion card.
In the case of the SP512 card SCR is 16.

Example:
```
10 |VIDEO.ON
20 |SCREEN.OUT,2.... Stores contents of CRT to
                     VIDEO Bank 2
```

SCREENS

Format: |SCREENS

Function: This command is used after the |VIDEO.ON command. It calculates the maximum possible number of VIDEO RAM Banks, which is dependent on the size of your RAM-Exp. card, and puts the result into the system variable 'SCR'
(SCR+1 is the maximum of 16KB VIDEO RAMs).

Comment: In order to work effectively with the VIDEO and SCREEN commands your program must be able to determine the number of available VIDEO RAM Banks. The command |SCREENS causes the total to be calculated, and the result put into system variable 'SCR'. It is then a simple matter for you to write your SCREEN programs.

Example:
```
10 |VIDEO.ON
20 |SCREENS
30 Print SCR+1;" 16KB VIDEO RAMs are available"
```

SPOOL.ON

Format: |SPOOL.ON

Function: Turns the 32KB Printer-Buffer ON

Comment: All output destined for the printer, will be stored in the 32KB Printer-Buffer after inputting this command. The SPOOL function, frees the console for work continuation by 'taking charge' of all Hardcopy/Output functions. This makes it possible to Print and Type at the same time. The 32KB is taken away from the total available RAM, which can be verified by typing |ID, noting the display, typing |SPOOL.ON, and then typing |ID again.
IMPORTANT: Before inputting this command, be sure that you don't have any relevant data in memory. This measure will insure that nothing is lost when the SPOOL function is called.

Should you be in the RAMOPEN or VIDEO mode, the total number of records/Pictures which can be stored will also be affected.

Example: |SPOOL.ON

SPOOL.OFF

Format: |SPOOL.OFF

Function: Turns the 32KB Printer-Buffer OFF

Comment: This command is used to disable the Printer-Buffer. It also frees-up 32KB more memory for relative Data-File or VIDEO use.

Example: |SPOOL.OFF

VIDEO.ON

Format: |VIDEO.ON

Function: Causes the RAM-Expansion card to be divided up into a maximum of 17 (0...16), 16KB VIDEO RAM Banks, with each Bank being identified numerically. The total number of Banks can be derived by typing |SCREENS, and then adding 1 to the system variable `SCR'. All data which may have been present in the RAM card is lost.

IMPORTANT: The entire group of |VIDEO commands can only be used out of Bank 0 (Console RAM). The command |VIDEO.ON moves the built-in VIDEO RAM from location &C000 to &4000. HIMEM is likewise reduced to 16360, and must (!) not be reset to a higher value.

Example:
```
10 |VIDEO.ON
```

VIDEO.OFF

Format: |VIDEO.OFF

Function: Turns off the Auxiliary VIDEO RAM (All Banks)

Comment: This command returns the RAM Expansion card to its normal state and it may now be used for relative Data-File, or just a 32KB Printer-Buffer.

Example:
```
10 |VIDEO.ON
20 |SCREENS
30 ...
.
.
50 i=3
60 |SCREEN.IN,I
70
.
.
100 |VIDEO.OFF
```

THE ROM RESIDENT Z-80 MONITOR

The Built-In machine code Monitor, with which machinecode programs can be loaded, tested, corrected and saved, is a very powerful programming tool.

It can be called instantly, takes up very little memory space, and is faster than any other Z80 monitor on the current market. Written in pure machine code, the monitor has the capability of both Assembling and Disassembling, Tracing, Single-Stepping, Displaying and Altering individual memory locations and registers, and the manual setting of Breakpoints so that you can check on the actual run of your own programs.

The command for calling the monitor is |MON (| is achieved by pressing SHIFT and "@" keys), and MON signs-on with an asterisk (*), and a blinking cursor. To exit the Monitor, simply press 'ESC'.

One remarkable item worth mentioning, is the fact that the Monitor can be called at any time, even during another programs run, and upon exiting the Monitor, you will find yourself right back where you were before you called it, with the program still running!

The Commands of the Built-In Monitor

Appearing below in alphabetic order are the Monitor commands, and a detailed explanation of each. You must remember that all data is displayed in hexadecimal format, that is...an 8-bit value occupies two places, while a 16-bit value occupies four.

A address -- Line Assembler

With the Line Assembler, you may input Z80 commands in their readable Mnemonic format. These will then be immediately translated into the corresponding hex values, and deposited into the current address(es). The translation occurs upon pressing the ENTER key, and any error in the command or its' syntax will cause a '?' mark, a space, and the same address to be displayed again. All numeric data must (!) be preceeded by a '&' so it is not misinterpreted as an address, and TAGS and LABELS are not supported. (No variable addresses allowed)

Example of use:

```
A2400 <ENTER>     Using the Line Assembler at address 2400H
2400 LD A,&65     Load the A register with 65H
2402 LD H,A       Load the H register with the contents of A
2403 EX DE,HL     Exchange contents of DE and HL registers
2404 JR &2400     Jump (Relative) to address 2400H
2405 <ESC>        Leave the Line Assembler Mode//Return to Monitor
```

B -- Breakpoints View/Change

The monitor also allows you to set up to eight Breakpoints of your choosing.
Should the Monitor encounter one of these eight breakpoints during the run of your (any) program, it will stop execution, and display the ready prompt (*).

Typing B<ENTER> will cause all breakpoints to be displayed:

```
-B1-    -B2-    -B3-    -B4-    -B5-    -B6-    -B7-    -B8-
0000    0000    0000    0000    0000    0000    0000    0000
```

The cursor will then appear below B1, and the addresses of any one (all) breakpoints may be 'typed' in. Pressing enter causes the cursor to 'jump' to the next position 'B2' and so on until all eight have been set. Pressing 'ESC' at any time terminates the Breakpoint Set sequence, and returns control to the Monitor.

D addr1,addr2 -- Display in HEX and ASCII format

The D (Display) command allows you to look at any (all) memory locations. The display appears in the following format:

```
0100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

The first value in the line (100) is the start address of the line being displayed, the next 16 (!) values are the Hex equivalent of the actual contents of each successive memory location. Finally... the ASCII equivalent of the data being displayed is shown. It is important to remember that no value less than 32 nor greater than 127 will be displayed in their true form. Instead.. a period will be displayed for these "Non-ASCII" values.

Simply typing D addr<ENTER> will cause the address specified along with the next 100H locations to be displayed.

Typing D addr1,addr2<ENTER> will cause the address specified, and the next xxxH (addr2-addr1=xxxH) locations to be displayed.

Pressing any key will stop the display, pressing another will re-start it.

Terminate the Display mode by pressing 'ESC'.

F addr1,addr2,const -- Fill memory with a constant

Often, it is necessary to erase or fill an area of memory with a certain value. This need is satisfied by the F (Fill) command, which fills a specified area of memory with a user defined constant.

Example: F C000,FFFF,FF.. Fills all memory from C000 to FFFF with 'FF'

G addr -- Go command

The G (Go) command allows you to start your machine code programs. It allows either Start from present position, or Start from a specific address (addr).

Example: G120<ENTER>... Starts execution of your program from address 120H

I fname -- Loads a Binary program from Cassette/Diacette

With this command, it is possible to load Binary files from Cassette/Diacette into the Monitor and Examine, Change or Run them. The source is selected using the |CAS or |DISC commands from BASIC (Before the Monitor is called). After the load is complete, a Load address, Length, and Start Address will be displayed.

```
Example:        ITEST.BIN<ENTER>
                <2000,0380,2140>....  Load address=2000, Length=380
                *                     and Start=2140H
```

L addr1,addr2 -- List in disassembled form

The use of this command is similar to that of the 'D' command, with the exception that the display is in both Hex and Mnemonic (Z80 language) form.

```
Example:        L 0100,0106<ENTER>
                0100 3E64       LD    A,64H
                0102 ED5B3204   LD    BC,(0432H)...Note the Inversion
                                                   Here
                0106 E3         EX    (SP),HL
                *
```

Lists the memory contents between addr1 and addr2 in Mnemonic form.

M addr1,addr2,addr3 -- Move memory

The M (move memory) command is used to duplicate or move memory between two specified addresses (addr1 and addr2) to another area of memory (Starting at addr3). If used in conjunction with the fill command, it is possible to re-locate an entire program in memory and then 'Erase the Original'.

```
Example:        M 0000,3FFF,C000<ENTER>...  Makes copy of memory
                                            between 0000 and 3FFF
                                            and puts it in memory
                                            starting at address C000
                                            (CRT Video)
```

O fname,addr1,len,addr2 -- Output contents of memory to device

This command is used to 'save' the contents of memory starting at addr1 and having a length of len with a starting address of addr2, under the name fname to the device (Cassette or Diacette) specified.

```
Example:        OTEST.BIN<ENTER> .. Starts the SAVE-File sequence
                *                   which checks the filename for
                                    Syntax. If OK, you will then be
                                    asked for the Load addr, the
                                    Length, and the Start addr one
                                    after the other
```

P -- Turn ON/OFF the Printing Device

When a printing device is connected, the P (Printer) command can be used to selectively turn it on or off. This can become very handy if only portions of the displayed data are to be printed, or a permanent record of a session is desired.

R -- Display/Alter Register contents

This command allows you to View and alternately change the contents of all of the of the Z80s' registers. They are Displayed and changed in much the same way as the breakpoints. NOTE: only the first register set can be changed.
(See B explanations)

S addr -- Show contents of memory

The S (Show) command allows you to display the contents of single memory locations in Hexadecimal format, and optionally change them.

```
Example:        S4000<ENTER>
                4000 $ 24  34<ENTER>..Location now contains 34H
                4001 C 43  40<ENTER>..Location now contains 40H
                4002 : 58  44<ENTER>..Location now contains 44H
*
```

T addr,xxx -- Trace program execution/single-step

The T (Trace) command allows you to trace the actual run of your program on a step-by-step basis, starting at address addr, for xxx steps. This enables you to troubleshoot your program and/or identify timing/execution bugs. The contents of the Z80 registers are displayed after completion of each step, and execution of the 'run' can be terminated at any time by pressing ESC. Note that it is not possible to Single-Step ROM resident programs, or those which utilize the Z80's alternate register sets. Additionally you must (!) insure that your stack-pointer does not occupy an address under ROM, as this can lead to system crash.

Example:	T4000,0007<ENTER>...	Traces 7 Steps of program execution from addr 4000H. The contents of the Z80 registers will be displayed, and like the D command, it may be stopped or terminated at any time.

THE EXPANDED BASIC GRAPHIC COMMANDS

FAST

Format: |FAST

Function: Increases the VIDEO output speed in mode 2

Comment: When working in 80-column mode, you have the possibility of increasing the speed of Text output (To the CRT) by a factor of two. The only drawback, is the loss of "windowing" which cannot be fully utilized with |FAST. The command |SLOW will return the I/O system to normal, and restore "Windowability"

Example: As a demonstration of the power of the |FAST command, type in the following 'One-Liner':

```
10 Z=Z+1:?Z:Goto 10
```

Now, type 'RUN', observe the CRT, and press ENTER after you've gotten used to the speed, type |FAST and RUN it again. The drastic change in speed is very exciting....

FRAME

Format: |FRAME

Function: Allows for 'Flicker-Free' moving graphics

Comment: Because of the fact that the Data which is to be displayed on the CRT is not always syncronized with the physical makeup of the CRT itself, or the position of the electron stream, your moving graphics can sometimes be seen to 'Flicker and Twitch'

As an example, key in the following program:

```
10 MODE 2
20 FOR I=1 TO 80
30 LOCATE I,12:?"o":LOCATE I,12:?" "
40 NEXT
50 GOTO 20
```

When you RUN the program, the characters can be seen to Jerk erratically from one side of the CRT to the other. While some PC owners would be satisfied with any moving graphics, you are one

of the few, who demand more than others can offer. Stop the program and key in the following:

```
35 |FRAME
```

Now Run it again, and you'll see what your CPC 464 can really deliver. Absolutely flicker-free moving Graphics...

GCHAR

Format: |GCHAR,x,y,@var

Function: Reads a character from position x,y into the variable var.

Comment: Should you desire to find out which character is currently located in CRT position 12,14... You could do so with this command. Of course you must first prepare a place for this character by defining an integer variable to contain it. This must be done before the command is given, and has the following syntax:

```
10 DEFINT A:A=0
```

Thereafter, it is possible to execute the command |GCHAR,x,y,@A. After execution, the variable A will contain the ASCII value of the character at position x,y.

Please note that inadmissible coordinates or Graphic commands bring back a '0'.

GPAPER

Format: |GPAPER,value

Function: Selects and assigns a Graphic BACKGROUND Color

Comment: With this command it is possible to select and assign a new background color. See @MASK for additional information.

Example: |GPAPER,3... The new color for the Graphic BACKGROUND is 3

****
* GPEN *
****

Format: |GPEN,value

Function: Selects and assigns a new Graphic FOREGROUND color

Comment: This command, like the |GPAPER command is used to change graphic FOREGROUND colors. See |MASK for additional info.

Example: |GPEN,6.... The new color for the Graphic FOREGROUND is 6

****
* MASK *
****

Format: |MASK,value

Function: Defines a mask for Graphic output of Lines and Points

Comment: Points and lines on the CPC 464 are normally drawn in a single color. With the help of this command, it becomes possible to do multicolored line and point graphics. The 'PEN' for this multicolored output is a function of the number(s) assigned to the variable value. As an example, let's say you've input the following:

```
10 |GPAPER,2
20 |GPEN,12
30 |MASK,&55....55 has the binary value 01010101
40 Draw 200,200
```

When you RUN the program, the drawn line will be alternately Blue (2) and Yellow (12). This means that the value 55H (01010101) determines the physical makeup of the PEN. The 0's turn on the Ink for the BACKGROUND, and 1's the Ink for the FOREGROUND

****
* UNMASK *
****

Format: |UNMASK

Function: Removes the Graphic Mask

Comment: After input of this command, the CPC Draws lines and Points just like it used to. All |MASK definitions are cancelled

SLOW

Format: |SLOW

Function: Cancells the |FAST command.

Comment: This command causes the speed of the CRT output to return to normal.

APPENDIX

A. COPY-PATCH

The utility program COPY.COM, which was recieved as part of your vortex Floppy Disc System, will not run under your new 62K CP/M. There's no need for alarm, just a bit of patience....and the DDT.COM program. What we're going to do here, is talk you through a PATCH of the COPY.COM program.

First, read through the entire procedure to insure at least a minimal understanding of what is going to take place. We know that you are apprehensive of machine code, Hex values and the like...but if you take your time, and do everything slow and deliberate, you'll have no trouble at all.

****
Note: <CR> as used here...means to press ENTER
Note: O as used here is a ZERO and not a big "O"
****

When you are finished reading, grab a discette, which has copies of both DDT.COM and COPY.COM program, and we'll get started.....

(1) Insert the discette, which contains both programs, and load CP/M (|CPM<CR>)

(2) Type DDT COPY.COM<CR>...This will load DDT.COM, and it will in turn load the COPY.COM program

(3) If you did everything right... you should see the following on the CRT:

```
DDT VERS 2.2
NEXT  PC
0780  0100
-
```

(4) The short Dashed-Line is the "Ready" prompt for DDT.

(5) Now type: S 1CA<CR>...Show location 1CA

(6) DDT responds with:

```
01CA 2A  x...
```

The cursor will be located at the "x"

(7) Type in 21<CR>

(8) 01CA 2A 21
01CB 06 x... The next location (1CB) will appear

(9) Hit ENTER... No change to be made at this location

(10) 01CA 2A 21
01CB 06

```
     01CC 00  x... The next location (1CC) will appear

(11) Type in 9F<CR>

(12) 01CA 2A  21
     01CB 06
     01CC 00  9F
     01CD B7  x... The next location (1CD) will appear


(13) Hit ENTER again...No change at this location either

(14) 01CA 2A  21
     01CB 06
     01CC 00  9F
     01CD B7
     01CE ED  x... Hit a full-stop '.' and <CR>
```

The Patch is now all but finished! After you hit the ENTER <CR>, DDT will again display his 'Ready' prompt. Now all we have to do it to DUMP the contents of the locations which we changed, to insure that everything is in order.

(15) Type D1CA<CR>

026F. The dump will look like this:

```
01CA 21 06 9F B7 ED 52 !....R
01D0 D1 AF ED 52 3C 30 FB 3D FE 00 CA 15 04 32 B3 02  ...R<0.=.....2..
01E0 3A B4 02 FE 01 28 21 3A B2 02 3C 47 21 00 00 ED  :....(!:..<G!...
01F0 5B B0 02 19 10 FD 3A B3 02 5F 16 00 AF ED 52 3C  [.....:.._.....R<
0200 30 FB 3D D6 1E D4 1A 04 3A B4 02 FE 00 28 06 CD  0.=.....:....(..
0212 38 01 CD 42 01 CD 99 03 AF 32 AA 02 3E 30 32 F8  8..B.....2..>02.
0220 06 32 15 07 16 00 06 00 AF 32 AB 02 CD 7E 02 3A  .2.......2...~.:
0260 C6 30 32 F8 06 32 15 07 16 01 1E BA 11 5A 06 CD  .02..2.......Z..
- .... The DDT Ready prompt will appear after the dump finishes
```

Now look at the very first line of the dump. The characters 21 06 9F must appear there, when the PATCH has been entered correctly. If this is not the case, type CTRL-C (Press CTRL and C keys at the same time), and then go back to Item #2, and start all over again.

In the event it came out as it should have...Type CTRL-C to leave DDT, and return to CP/M

*Finally, to secure the contents of memory to the disc, type the following:

SAVE 7 COPY62.COM<CR>... This command causes 7 256byte pages, starting at location 100Hex to be saved to disc as a file named COPY62.COM.

This new COPY program (COPY62.COM) will run not only under your BIG 62K CP/M System, but under the old 44K CP/M as well.

B. PARA the vortex Disc Manager

The programm PARA works faultless only under a non-patched 38K CP/M (as described in the PARA user manual). 'Non-patched' means that you get this 38K CP/M out of your 44K CP/M (purchased which vortex disc station) by using the MOVCPM and SYSGEN programs.
Again: it muat be the CP/M that never came in contact with the program PATCH.COM !

C. GRAPHIC MASTER 2.0

If your GRAPHIC MASTER 2.0 doesn't run faultless after the installation of the RAM Expansion card, read the following very carefully.

1. RESET your system with CTRL+SHIFT+ESC
2. poke these addresses
   POKE &AC01,&AF <CR>
   POKE &AC02,&32 <CR>
   POKE &AC03,&45 <CR>
   POKE &AC04,&AE <CR>
3. load from your original GRAPHIC MASTER diacette by typing
   LOAD "GRAMA" <CR> the program GRAMA.BAS
4. erase now in line 30 the part with the MEMORY-command
5. save the program with SAVE "GRAMA",P <CR>
6. ready

D. SPTEST.COM

On the cassette in your RAM Expansion card package is beside the program PATCH.COM also the programm SPTEST.COM. This programm allows you to test your 'self-expanded' RAM card, especially to test if the RAMs you bought are 'good' or 'bad'.
Please note that this program is running endless, if no error occurs.
You transfer this program to disc in the same way as you did with the program PATCH.COM.

R E F E R E N C E - C A R D

for the new BASIC commands within the vertex BOS 1.0

ALL COMMANDS MARKED WITH A '*' ARE ONLY WORKING IF YOU START PREVIOUSLY THE BOS 1.0 OPERATING SYSTEM BY TYPING :BOS <CR>.

[,m,n,o]	number of a Program memory bank 0....8
addr	memory address
[,]	optional inputs
par1,..	parameters to be transfered by a CALL
chn	output channel number
line	line number of a BASIC Statement within a program
var	integer variable
var$	string variable
wert	integer number
len1...	field length
reclng	record length
recnum	record number
i	video RAM number 0....16

Command	Syntax	Notes
:BANK	,n	*
:BASIC		*
:BOS		*
:CALL	,n,addr[,par1,par2,..]	*
:COMMON	[,@var$,1,m,n,..]	*
:DEV	[,chn]	chn=0,8,9 *
:FAST		
:FRAME		
:GCHAR	,x,y,@var	
:GOSUB	[,n],line	*
:GOTO	[,n],line	*
:GPAPER	,wert	
:GPEN	,wert	
:ID		*
:LIST	[,l,m,n,o]	*
:LOAD	,@var$	*
:MASK	,wert	num=0..255
:MON		
:NEW	[,l,m,n,o]	*
:PEEK	,n,addr,@var	*
:POKE	,n,addr,wert	*
:RAMCLOSE		*
:RAMFIELD	,len1,len2...	*
:RAMOPEN	,reclng	*
:RAMREAD	,recnum,@var1$,@var2$,..	*
:RAMWRITE	,recnum,@var1$,@var2$,..	*
:RECORDS		calculates REC *
:RETURN		*
:RUN	[,@var$],n,line]	*
:SAVE	,@var$[,l,m,n,o]	*
:SCREEN.IN	,i	*
:SCREEN.OUT	,i	*
:SCREENS		calculates SCR *
:SLOW		
:SPOOL.ON		*
:SPOOL.OFF		*
:UNMASK		
:VIDEO.ON		*
:VIDEO.OFF		*

VORTEX SERVICE PASS

By completing and sending in this Service Pass, you become a fully Registered User/Owner of the vortex RAM-Expansion Card, and as such are entitled to recieve Software Improvements to it, at no further charge except those of postage.

SURNAME: _____

CHRISTIAN NAME: _____

FIRM/BUSINESS NAME: _____

ADDRESS: _____

CITY/PROVINCE: _____

TELEPHONE/TELEX NO: _____

(Tick appropriate box)

SP/64M	SP64	SP128	SP256	SP320	SP512
☐	☐	☐	☐	☐	☐

DATE OF PURCHASE: _____

WHERE PURCHASE WAS MADE: _____

COMMENTS:

Please mail this form to:
SCREENS Microcomputer Distribution Ltd
Main Avenue, Moor Park, Northwood,
Middlesex, England.