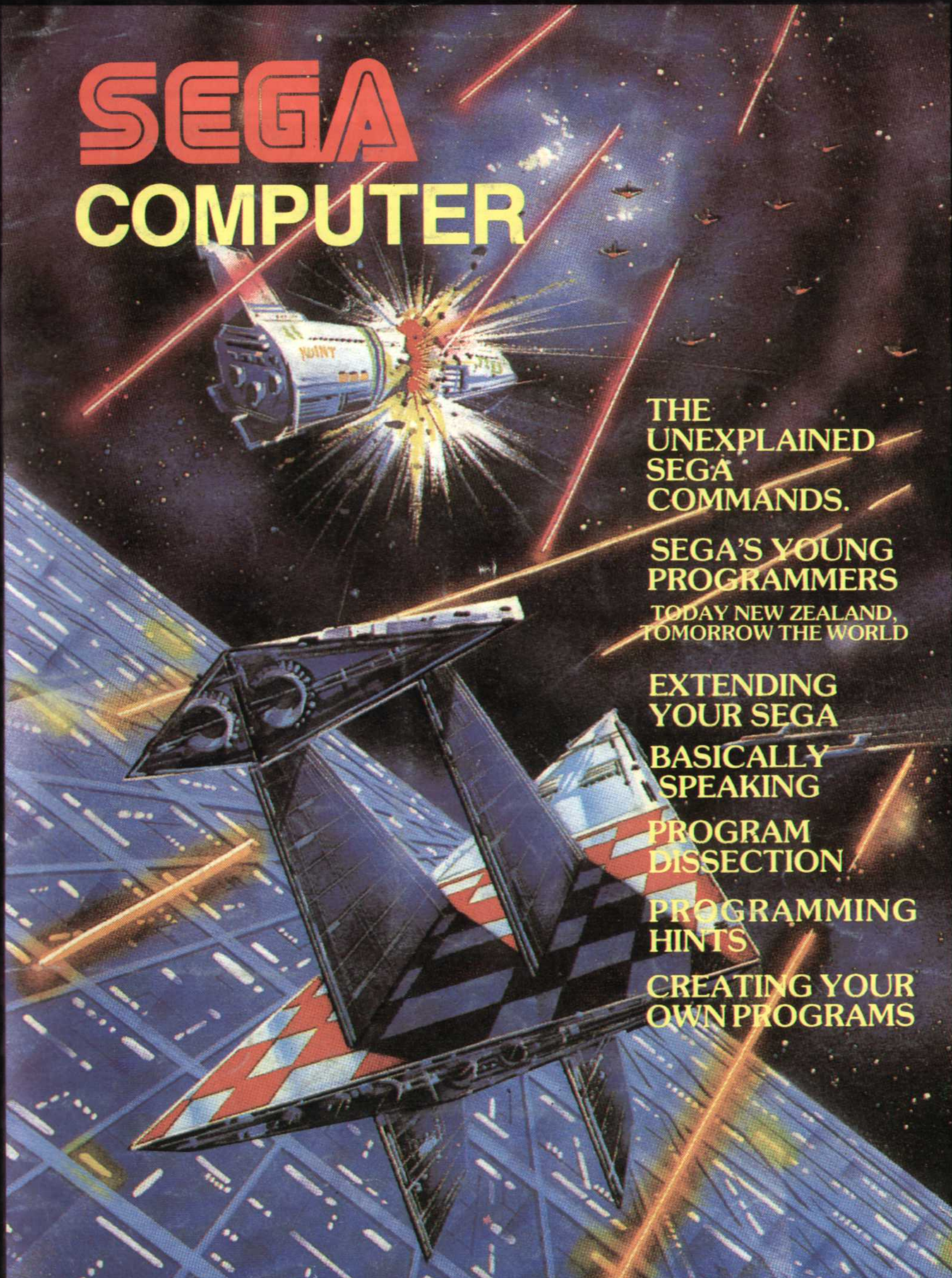
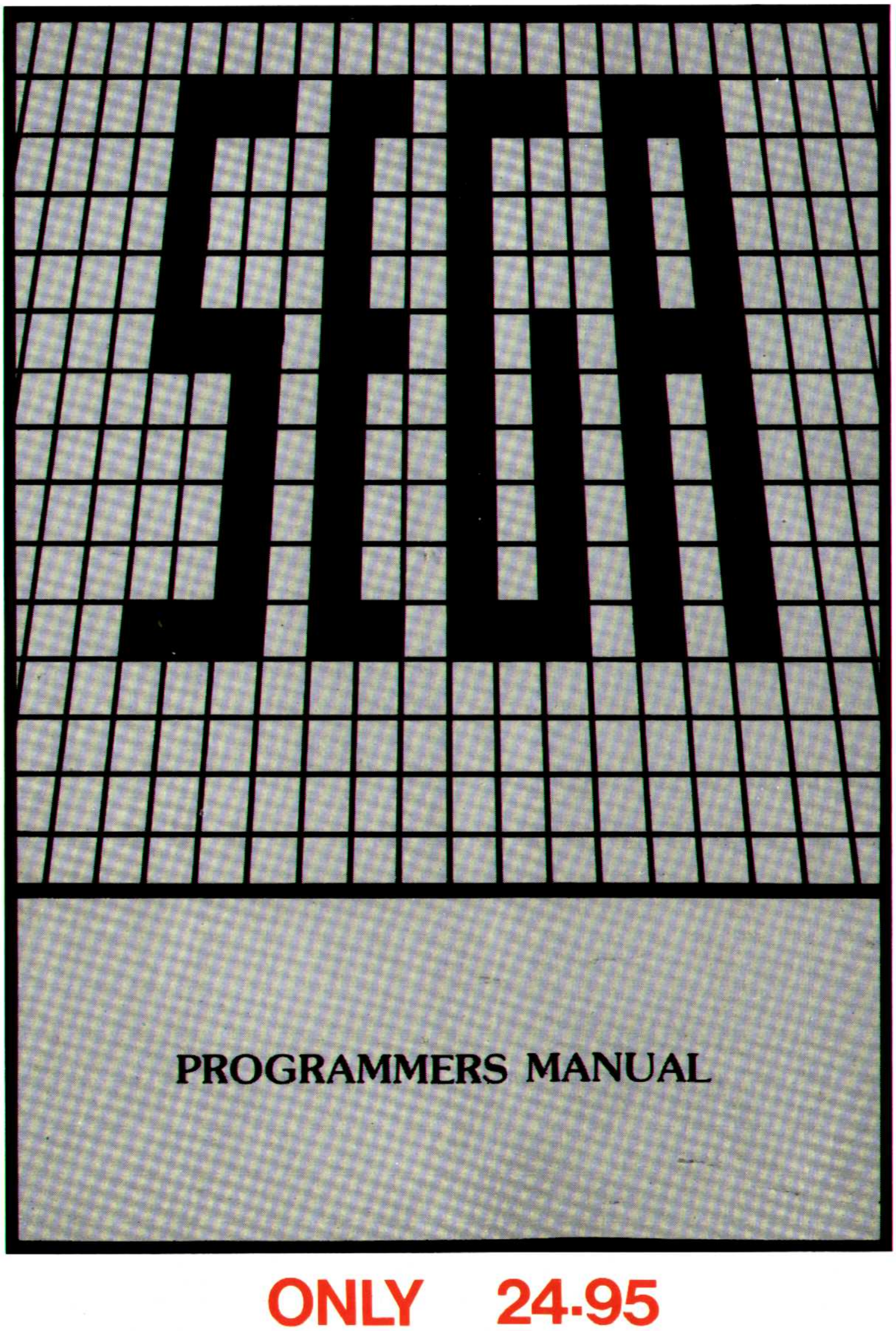
SEGA COMPUTER



NOW AVAILABLE!



ONLY 24·95
SEE PAGE 14

WELCOME TO THE SEGA USERS CLUB

As a Sega owner, you will probably have joined the Users' Club for one of three reasons.

A. You are already quite skilled with computers, and are looking for more advanced information to write better programs.

B. You have no idea how to operate a computer and are hoping for clear instructions and guidance to assist you in becoming "computer literate".

C. Your American Express Gold Card was refused so this was the next best thing.

Having made this wise decision, what can you expect to receive over the next 12 months? Well, for beginners, we shall be covering basic programming commands, with clear explanations of how your computer interprets them, and how they are used in programming.

Month by month, these will build your knowledge and ability to create your own programs and help you achieve the standard of capability which you set as your goal.

Advanced programmers will benefit from the pages of input enclosed each issue, and from letters, answering queries and problems, which we invite all owners to send in. As well as useful tips and programming aids, which you have discovered, and want to share with your fellow members, to encourage local meetings, each month we will publish the dates and venues of anyone who wants to hold a meeting in a particular area, on a page reserved solely for this purpose.

From time to time, we will make special offers through the magazine for members to take advantage of discounts off certain software titles, available through our Sega stockist network.

Each month, the new products available will be listed, with explanations of their functions and operation. Members are also invited to send in programs for evaluation, should you consider your work commercially viable.

This is your magazine, so we welcome your suggestions on what you want in it (no suggestions on where to put it if you please). Your support is needed to maintain the quality of the magazine, so come on all you budding superbrains — start sending in those programs and letters — all printed on your shining new Sega Plotter Printer, of course.

I look forward to hearing from you in the near future.

Yours faithfully,
GRANDSTAND LEISURE LTD

[illegible]

P. Kenyon,
COMPUTER DIVISION.

PUBLISHED BY: NOMAC **PUBLISHING LTD.**

Introduction

You are about to be introduced to the exciting new world of the personal computer. Until just a few years ago, the size, price, and complexity of computers put them beyond the reach of the individual purchaser. Today, Sega Home Computers bring you remarkable computing power in affordable compact units that can be easily set up in your home, office, or school.

Whether you have years of computer experience or have never worked with computers before, the innovative and flexible features of your Sega Computer offer you a wide variety of applications. Within minutes, you can begin using your computer to:

- Manage your personal resources
- Develop projects for home and business
- Bring new dimensions to education — for you and your children
- Provide engaging new types of entertainment for the entire family — and much more.

Powerful Sega Basic

Sega Basic is a simple but very powerful computer language. With Sega Basic Cartridge you can develop and use your own computer programs for applications ranging from colour graphics to statistical analysis and more. This language makes your Sega Computer a "true" computer —not a video game or electronic toy.

Convenient Cartridge System

The system of easy-to-use, snap-in Cartridges assures the continued versatility and usefulness of your computer. These rugged, all solid-state cartridges are completely preprogrammed for you. You just snap them in, and they "prompt" you through activities, applications, games, and entertainment. With a cartridge plugged into the computer console, you can start using your computer immediately. You can choose from a wide selection of cartridge titles. Ask your dealer to see all of them!

Tape and Diskette Programs

In addition to cartridges Sega offers a variety of convenient software on tape or diskette, ranging in complexity from simple games applications to high-level business and professional programs. Like cartridges, these applications are ready for you to use, without any programming on your part. Programs on cassette tape require the Sega Audio Lead to connect the computer and your cassette recorder, or Sega Data Recorder diskette programs require the Sega Disk Memory System. Ask your dealer to show you a list of the many tape and diskette packages available from Sega and other software developers.

Placement and Care

The Sega SC3000 computer generates and uses radio frequency (RF) energy. **IF NOT INSTALLED AND USED PROPERLY** (as outlined in the instructions provided by GRANDSTAND LEISURE (NZ) LTD), the computer may cause interference to radio and television reception (which you can determine by turning the equipment on and off).

If this equipment does cause interference to radio or television reception try to correct the problem by one or more of the following measures:

- Reorientate the receiving antenna (that is, the antenna for the radio or television that is "receiving" the interference), if possible.
- Change the position of the computer with respect to the radio or television equipment that is receiving interference.
- Move the computer away from the equipment that is receiving interference.
- Plug the computer into a different wall outlet so that the computer and equipment receiving interference are on different branch circuits.

If these measures do not eliminate the interference, please consult your local Sega dealer or any experienced radio/television technician for additional suggestions.

NB. Electronic equipment can be damaged by static electricity discharges. Static electricity build-ups can be caused by walking across a carpet. If you build up a static charge and then touch the computer, a cartridge, or any accessory device, you can permanently damage the internal circuits. We suggest you always touch a metal object (a door knob, a desk lamp, etc) before working with your computer, connecting accessory devices, or handling or inserting a cartridge.

Interference

First, find the right location for your computer system. Select a place where sunlight or bright light doesn't fall directly on the screen. Also, it's best to place the system on a hard-topped non-metallic surface, such as a table. Do not set the computer console on top of a television set.

Correct ventilation is necessary for the continued proper operation of your computer system. Be sure air can flow freely through all the ventilation slots on the bottoms, backs, and tops of the console and monitor (or TV set). Do not obstruct the ventilation or enclose the system in any way.

From time to time you may want to clean the surfaces of your computer. First, turn the computer OFF. Then gently wipe the surface using a damp, lint-free cloth. Do not use solvents or other cleansers to clean the computer console.

CARING FOR YOUR SOFTWARE

WHAT IS SOFTWARE: The term software refers to any program or programs which make the computer perform a specific function. For most of us, the software is stored on a cassette tape, or is a games cartridge. By obeying some simple rules, you can prolong the life of your software, and save time and money. The basic rules are,

1) **NEVER EXPOSE THE CASSETTE TO MAGNETIC FIELDS,** Don't place your cassette on top of the TV set, or right alongside it. Parts of your program might be erased. This also applies to video monitors.

2) **NEVER USE A PLASTIC CHAIR TO SIT ON WHEN USING A COMPUTER,** Plastic can generate static electricity, the major cause of computer/program failure.

3) **NEVER WEAR WOOLEN MATERIAL WHEN USING YOUR COMPUTER,** Again, wool generates static electricity, so be careful.

4) **NEVER TOUCH THE EDGE (WITH CONTACTS ON IT) OF ANY GAME CARTRIDGE,** Static electricity could damage the internal components of the games cartridge.

5) **NEVER PLACE YOUR GAMES CARTRIDGES ON TOP OF A TV SET,** when the power is turned off or on, this could damage the games cartridge, depending upon the type of set, and its age.

6) **REGULARLY CLEAN THE CASSETTE PLAYER HEADS WITH ALCOHOL,** Tape particles actually rub off when the tape passes the cassette player's head. This builds up over a period of time, so use a cotton bud soaked in alcohol to clean it off, and also clean the capstan pinch roller. It's made of rubber. This cleaning process should be done every ten hours of use, or sooner depending upon the environment in which the computer is being used.

7) **REGULARLY DEMAGNETISE THE CASSETTE HEADS.** This can be done by any reputable stereo shop, or you can buy one of those demagnetising cassettes and do this yourself. Do this about once a month. It will help to eliminate all those cases where programs that loaded last week, won't load at all this week.

8) **ONLY USE CASSETTE TAPE OF HIGH QUALITY.** This means don't try to use C60, C90, C120 tapes on your computer. The longer a tape is, the thinner it is, so longer tapes are more likely to sustain damage. The computer is extremely sensitive to tape damage, so use the thickest tapes you can buy. The C10, C12, C15, C20 computer grade tapes will be the best choice in the long term.

9) **KEEP A BACKUP COPY OF ALL PROGRAMS ON A SEPARATE TAPE.** If something does go wrong, then you won't have lost all your hard work. If and when you get a disk drive, this rule still applies.

10) **NEVER TURN THE COMPUTER OFF THE BACK ON AGAIN.** Wait about ten seconds first before turning it back on. Serious damage could occur if you don't. Just because the computer hasn't blown up so far is no guarantee that it won't very soon if you have been doing this.

By following the guidelines set out above, you will prolong the life of your computer and software, while maintaining relatively error free operation.

—EXTENDING YOUR SEGA—

B. Brown 1984

THE BASIC COMPUTER SYSTEM: The SEGA computer is made up out of three main sections, as shown below.

```
************                                ************         ***********
*          *--------------------------------* M        *---------* I       *
*    C     *   ADDRESS BUS                  *  e       *---------* n       *
*    P     *--------------------------------*   m      *---------*  p      *
*          *                                *    o     *         *   u     *
*    U     *--------------------------------*     r    *---------*    t    *
*    N     *   DATA BUS                     *      y   *         * O       *
*    I     *--------------------------------*          *---------*  u      *
*    T     *                                ************         *   t     *
*          *-------------------------------------|  |------------*    p    *
*          *   CONTROL BUS                                       *     u   *
*          *-----------------------------------------------------*      t  *
************                                                     ***********
```

CENTRAL PROCESSING UNIT (CPU): The CPU performs the following tasks,

1) Interprets the users command
2) Reads and decodes the information from the keyboard
3) Controls the sound generator
4) Sends the proper information to the Video display
5) Runs user programs etc.

It accomplishes this by communicating with all the devices which are connected to it, and transfers information between the devices as required. (This may involve the manipulation of the data internally within the CPU before it is transferred).

MEMORY: There are two types of memory used, Read Only and Random Access memory (ROM and RAM). The ROM contains the BASIC language (beginners all-purpose symbolic instruciton code), and the necessary programs which enable the CPU to communicate with all the other devices. The contents of the ROM are retained when the power is turned off. The ROM can only be READ by the CPU, and is a sort of text book which which the CPU gets the necessary instructions which tells it of what to do. RAM is used for temporary program storage, and its contents disappear when the power is turned off. This explains why you must transfer your program to cassette tape. RAM can be thought of as a blackboard. Information can be both written onto it and erased. The RAM contents can be altered by the POKE command, but ROM contents cannot, and should not or damage to the computer may occur.

INPUT/OUTPUT DEVICES: These devices allow the user to communicate with the CPU and allows feedback from the CPU to the user. The Sega has the following Input/Output devices,

Port Number in Hexadecimal.	
&HDC, &HDD, &HDE, &HDF	The Keyboard, Cassette and Printer
&HBE, &HBF	The Visual Display Processor
&H7F	The Sound Generator

COMMUNICATION BETWEEN DEVICES: Each device connected to the CPU is given a unique box number (ADDRESS). The CPU can communicate with the specific device by placing its box number (ADDRESS) on the ADDRESS BUS. A BUS is a common highway which allows communication between devices. Having placed the right address on the bus, (ie selected the correct box number), the CPU can then read from or write to the selected device. The CPU transfers information between devices in BINARY format. The smallest element in binary is a BIT. The ON state is represented as having one of two possible states, ON or OFF. The ON state is normally designated '1' whist the OFF state is designated a '0'. The CPU however, can work with eight bits at any given time. This group of eight bits is called a BYTE. A byte can be thought of as eight buckets, where each bucket could be full or empty. It thus follows that the maximum number of combinations possible with eight bits is 256. Each address (box) is capable of storing eight bits, thus any box can have as its contents a value of between 0 and 255. The CPU moves the bytes around via the DATA BUS. In this case the DATA BUS is bidirectional, ie information can travel from the cpu to a device or from a device to the cpu. Each device is connected to the address bus which is used by the cpu to tell the device that the cpu is talking to it. The address bus is sixteen bits wide, thus the cpu can access any one of 65536 possible locations (or boxes which hold 8 bits each). To inform the devices as to which way the information is travelling on the data bus, a CONTROL BUS is used. This control bus informs the device if it should expect to receive data (ie a write) or whether it should present data so that the cpu can ready it (ie a read).

The cpu has temporary storage boxes inside it called REGISTERES. When the cpu wishes to transfer information from one address to another, the cpu carries out the following sequences,

1) Places the correct address (box number) on the address bus
2) Reads the contents of the selected address via the data bus
3) Transfers the information to one of its registers
4) Places the destination address on the address bus
5) Transfers the contents of its register onto the data bus
6) Informs the device at that address to get the new contents for that address, which is appearing on the data bus.

INPUT/OUTPUT PORTS: The cpu can have up to 256 separate ports. These are selected by an eight bit value on the address bus, combined with a special signal on the control bus. This signal is activated when you use the command OUT or INP in basic. Each of these 256 separate ports can hold an eight bit value (ie 0-255). Not all of the ports are used however, and it is advisable that you do not POKE the ports unless you know what you are doing.

This covers the sequence of operations in a relatively simple manner, and has served to introduce you to the Basics of Computers at this time.

NUMBER BASES.

DECIMAL: Decimal numbers have a base of 10. This means that their positional values are as follows, eg, the number 1123 in decimal is actually,

```
                   3
 ----------1 * 10   = 1000
|
|                  2
|  --------1 * 10   =  100
| |
| |                1
| |   -----2 * 10   =   20
| | |
| | |              0
| | |  _ 3 * 10   =    3
| | | |

1 1 2 3
```

Note that each time a digit is shifted to the left, the value of the number increases by a factor of ten.

BINARY: Binary numbers are represented by a series of BITS, where each BIT can be a '0' or a '1'. It is usual to represent bits in groups, where there are eight bits to each group. A group of eight bits is called a BYTE. These eight bits are numbered bit0 to bit7, and bit7 is always on the left, with bit0 on the right. These eight bits can be either on or off, so a byte in binary could be represented as follows,

B7	B6	B5	B4	B3	B2	B1	B0
1	1	1	0	1	0	1	1

Bit seven is the bit which has the greatest value, while bit zero has the least value. Bit seven is thus called the MOST SIGNIFICANT BIT (MSB) while bit zero is called the LEAST SIGNIFICANT BIT (LSB). In terms of the decimal value of each bit, the following example should help,

Decimal Value	128	64	32	16	8	4	2	1
Binary digit	B7	B6	B5	B4	B3	B2	B1	B0

thus a byte of 11000000 will have a decimal value of 192, because bit 7 and bit 6 are both '1', so the decimal result is 128+64. Where a '1' occurs, the decimal value is added, while all '0's are ignored. Another example,

```
                         7
  ----------------- 1 * 2  = 128
|
|                        6
|  ---------------- 1 * 2  =  64
| |
| |                      5
| |  -------------- 0 * 2  =   0
| | |
| | |                    4
| | |  ------------ 1 * 2  =  16
| | | |
| | | |                  3
| | | |  ---------- 0 * 2  =   0
| | | | |
| | | | |                2
| | | | |  -------- 0 * 2  =   0
| | | | | |
| | | | | |              1
| | | | | |  ------ 0 * 2  =   0
| | | | | | |
| | | | | | |            0
| | | | | | |  -    1 * 2  =   1
| | | | | | | |
                           ---
1 1 0 1 0 0 0 1            = 209
                           ---
```

HEXADECIMAL NOTATION: Binary numbers of eight bits are sometimes tedious to write down, so a method was devised in which the binary numbers are represented in another form. This form is known as HEXADECIMAL (HEX). It has a number base of 16 digits (decimal has 10 and binary has two). The equivalent decimal, binary, and hex values are listed below,

Binary	Decimal	Hexadecimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

As shown, hex ranges from 0-F. When the hex number is larger, ie 16 in decimal then the hex number becomes 10. This is exactly the same as in decimal when you go from 9 to 10. Looking at a byte (a group of eight bits), the four least significant bits are called the LOWER NIBBLE, whilst the four most significant bits are called the UPPER NIBBLE. (A nibble is 4 bits).

Upper Nibble				Lower Nibble			
B7	B6	B5	B4	B3	B2	B1	B0
1	0	0	1	1	1	1	0

Binary value of each bit

To represent this in hex requires two hex digits, as each hex digit can only represent four bits. The upper and lower nibbles are converted to hex digits, with the resultant hex digits being written with the most significant one first. In the example above,

1001 in binary is '9' decimal so that's '9' in hexadecimal

1110 in binary is '14' decimal so that's 'E' in hexadecimal

so the corresponding hex digits which represent the byte 10011110 is '9E'. Hexadecimal digits are prefixed with &H in SEGA basic, and the hexadecimal value of any decimal number can be found by using HEX$.

```
              2
 ----- 10 * 16  = 2560
|
|             1
|  --- 15 * 16  =  240
| |
| |           0
| |  -  5 * 16  =    5
| | |
                  ----
A F 5           = 2805
                  ----
Hex Number
```

To verify that the hex number 'AF5' is equal to '2805' decimal, type,

PRINT &HAF5

To change the decimal number '2805' to hexadecimal, use,

PRINT HEX$(2805)

NOTE: The hexadecimal address's of &H8000-&HFFFF are represented internally within the Sega computer as negative numbers. Thus, the equivalent address's are listed below,

&H8000	-32768
&H8001	-32767
&HFFFE	-2
&HFFFF	-1

That concludes this session on number bases. In the next newsletter we will look at some SEGA commands that work on Binary numbers.

THE SEGA MEMORY MAP: The following table lists the memory map of the Sega computer. Also listed are the 1/0 (Input/Output) ports and their various functions.

```
                  &HFFFF  |-----------------| 65536 (-1)
MEMORY MAP:               | FREE SPACE &    |
                          | STRING STORAGE  |
                          |-----------------|
                  &HC7FF  |                 |
                          | SYSTEM RAM      |
                  &HC000  |-----------------|
                          |                 |
                          | BASIC PROGRAM   |
                  &H9800  |-----------------|
                          |                 |
                          | RESERVED RAM &  |
                          | BASIC POINTERS  |
                  &H8000  |-----------------| 32768 (-32768)
                  &H7FFF  |                 | 32767
                          | BASIC ROM       |
                          | OPERATING       |
                          | LANGUAGE        |
                          |                 |
                  &H0000  |-----------------| 0

                          ------------------
                          | Sound            | Poke Suitable
                    &H7F  | Generator Chip   |
                          |==================|
                    &HBE  | Visual Display   | Poke & Peek suitable
I/O MAP.            &HBF  | Processor Chip   | Status Register, Peek only
                          |==================|
                    &HDC  | Keyboard,        |
                    &HDD  | Cassette and     |
                    &HDE  | Printer          |
                          |==================|
                          | Key Matrix       |
                    &HDF  | Control          |
                          |------------------|
```

MAIN MEMORY MAP

&H0000 to &H7FFF Rom Routines

This area is exclusively for read out. Poke command is unusable.

Hexidecimal Address	**Nature of Routine**
10C0-17BF	VDP Character Table (8x8)
17C0-19FF	Basic Key Word Table
1CB1	Determination of the number of free bytes
2310	Places the next character into the DE register
2400	Writes the character in the A register to the video screen
2BD4 (2BD1)	Reads 80 bytes of data from the VRAM at address & H1800 into address & H9364: read 80 bytes from & H8B36 to video ram address & H1800: moves 80 bytes from & H9634 to & H8B36
2C2A (2BCE)	Reads the data from VRAM
2C32 (2BCB)	Writes the address in HL register to the VDP for VRAM reading
2C3D (2BC8)	Writes the data to VRAM
2C44 (2BC5)	Writes the address in HL register to VDP for VRAM writing
2C51 (2BC2)	Reads the VDP Status register
2C54 (2BBF)	Writes to the VDP register. Register in C, data in A
3604	Hexidecimal Conversion
3A03	Delay using the BC register
3A12	Writes a byte to the tape

Hexidecimal Address	**Nature of Routine**
3B33	Writes 8 bytes from address & H9413 to the VRAM
3D32	Accessing of screen 1,1
3D90	Accessing of screen 2,2
3DEE	Text and graphics screens
3FA0-411F	Keyboard characters arranged in a matrix form
4120-4258	Basic keyboard symbol table
4590	Resets the time $ to "00:00:00"
4756	Changes the cursor to the graphics mode
475E	Changes the cursor to the normal mode
4766	Lower case input
476E	Upper case input
4918	Inkey$
4A6F	Write the text pointed to in the HL register to current screen position
6800	Restart from 00H
6803	Restart from 38H
6806	NMI Entry
6809	Self-diagnostic routines RAM failure (single beep)
680D	Self-diagnostic routines ROM failure (double beep)
6811	Self-diagnostic routines VRAM failures (triple beep)
6AB5	Print FRE routine
6C37	Run routine
73B7	Error determination
73E8-7676	Error messages
779F	Verify routine

Hexidecimal Address	Nature of Routine
77F7	Skip program
7822	Found program
785D	Verifying end
788F	Verifying error
78D5	Load routine
78FD-790E	Compare file names
792B	Skip program
7956	Found program
7982	Load program
79AA	Loading end
79E9	Tape read error
7A40	Save routine
7A59-7A85	Save file name
7A94	Save number of bytes
7AB9	Save sync bytes
7AD2	Save program
7AED	Saving end
7B07	Write HL register to the tape
7B13	Pad file name with blanks

Ram Routines

8160/8161	Start of the basic program
8162/8163	End of the basic program
8164/8165	String storage pointer
8166/8167	Top of string storage
8168/8169	Top of memory pointer
82A2	Program found flag, 0= found program
82A3	File name being loaded from the cassette (16 bytes)
83A3	File name being saved on the cassette (16 bytes)
8B30	Basic stack area

Hexidecimal Address	Nature of Routine
8B36	&H80 bytes. Writes to VRAM &H1800 onwards
9336	Screen control
9339	Colour of the text screen
933A	Colour of the graphics screen
936A	(&H80 bytes VRAM stores &H1800= here) stores PH80 bytes of VRAM here which is then copied to VRAM &H1800
9411	Top of cursor range
9412	Bottom of cursor range
9413	8 bytes for storage of PATTERN command
9420	&H28 bytes for storage of VRAM date
9460-9480	INKEY$ Storage area
9484	Cursor control, 0=normal, 2=graphics
9485	Keyboard control 1=lower case, 0=upper case
9486	Key beep, 0=beep, 1=beep off
9489	Cursor position (X value)
948A	Cursor position (Y value)
948E	Time$ seconds
948F	Time$ minutes
9490	Time$ hours
9744/9745	Address of DATA byte

( ) Refers to the jump table address which calls the routine.

Please note the above map is as yet incomplete. Further addresses for new routines will be updated in each issue.

CREATING YOUR OWN PROGRAMS

When writing your programs you must learn the way in which a computer understands and interprets the various commands in Basic. Basic is the computer language which Sega understands, and is similar in many ways to the basic used by other computers. However, the differences with computers own operating systems mean that programs written for other machines require certain alterations before they can be run or understood by a different computer.

In certain short programs these differences may only be very slight, with more lengthy programs, particularly those which refer to Peek or Poke commands, the differences will mean altering the program which will be extremely hard.

When following the instructions which follow to help you learn to program, the Glossary at the back of the magazine will be useful to explain certain programming terms, and it should be used whenever you become confused or need clarification.

To Begin: all programs must run in a logical sequence. The way the computer determines what needs to be done in what order is by the programmer giving each line of the program a line number.

Normally these are spaced with an increment of 10 units. Thies does not use up any extra memory but it does allow for insertions of new lines of instruction to the program.

Sega has a command which is designed to save time by generating line numbers for you automatically. It is called "Auto". Once entered it will produce line numbers starting at 10, with an increment of 10; producing the next line number as soon as you press CR to enter the program line into the computers memory. To stop the computer producing new line numbers simply push the break key.

By typing a value larger than ten after the word Auto, the computer will begin line numbering from that amount onward. You may also change the increment.

Try the following examples.

If you specify only an **Initial Line**, then 10 is used as the increment.

```
NEW
AUTO 1000
1000 C$="Hi"
1010 PRINT C$
1020 END
1030 ---BREAK---
```

If you specify just an **Increment,** then 10 is used as the **Initial Line.** NOTE the comma before the 5 in the example. Remember, if you wish to specify only an **Increment**, the comma must be typed before the **Increment.**

```
NEW
AUTO ,5
10 Z=99.7
15 PRINT Z
20 END
25 ---BREAK---
```

CAUTION: When you are in the automatic line numbering mode, if a line number generated is already a line in the program, then the existing program line will be erased by the new line entered.

```
10 A=37.1
20 B=49.6
30 C=10.1
AUTO 25,5
25 PRINT A;B
30 END
35 ---BREAK---
```

If while in the automatic numbering mode, you should change the generated line number. This new number now becomes the **Initial Line.**

```
10 C$="GOOD"
20 D$="MORNING"
30 ---BACK SPACE WITH 3
     TIMES AND TYPE 25---

25 PRINT 4,5 & D$
35 END
45 ---BREAK---
```

PRINT

Print is a command. It is used to make the computer display something on the screen.

TYPE

Print 2 + 3 and press cursor return (CR).

The computer will display 5 on the screen.

Type

Print 7 + 4
Print 27 + 14
Print 2 x 7
Print 18 ÷2

An asterisk * tells the computer to do multiplication, while a forward slash / signifies division. Did you remember to press the CR key?

What we are doing is asking the computer to calculate mathematical problems and rather than using a command such as calculate we use a very versatile and commonly used command called P-R-I-N-T.

Type

Calculate 2 + 3 CR
Display 2 + 3 CR

Although the English looks correct the computer does not know the meaning of the words calculate and display and so thinks you have made a mistake. The computer will only accept words it knows, this is called the BASIC language.

The following exercise shows the effect of inserting spaces into the print command.

Type

Print 2x4
Print 2 x 4
Print 2   x   4
Pri nt 2 x 4

All these are valid commands except for the last one. Spaces can be inserted between words and numbers but it is wrong to insert spaces between the letters making up a word.

You should always try and put a space after a command word.

Print 2 / 3 is clear than
Print2/3

Up to now, whenever you have typed a command at the keyboard the computer has executed it immediately. If you want the computer to do the same thing again you must re-enter the command. This can get boring. A PROGRAM is a sequence of commands. Each line of a program consists of a line-number followed by a command. For example:

```
10 LET AGE=10        CR
20 PRINT AGE         CR
30 LET PAY=5*AGE     CR
40 PRINT PAY         CR
50 END               CR
```

This program is stored in the computer's memory. You can tell the computer to execute the commands of the program by typing RUN. In response, the computer will fetch from memory the command contained in Line 1 of the program and execute it. Then the computer will fetch the command contained in Line 2, and execute that. This continues until it reaches the END of the program. So you can see, the line numbers tell the computer the order in which the commands are to be executed.

Notice in this example that we can not only type numeric expressions but also alphabetical values.

When the program is run the computer displays two numbers on the screen. These numbers are 10 and 50. From our knowledge of the program, we know that 10 means '10 years of age', and 50 means '50 cents per week'.

So far we have used the PRINT command in two ways:

Print 2 + 3: The computer works out 2 plus 3, and display the answer on the screen.

Print Age: The computer displays on the screen the number stored in the memory box called age.

TYPE:

Print"HELLO"

The computer displays HELLO on the screen. We can get the computer to display any message on the screen by using a PRINT command and the message between two quotation marks.

FOR EXAMPLE

```
5 PRINT "FOR EVERY YEAR IN AGE I GET 5
 CENTS PER WEEK THEREFOR IF I AM 10 YE
ARS OLD I GET 50 CENTS PER WEEK"
10 LET AGE=10
15 PRINT "AGE IN YEARS"
20 PRINT AGE
30 LET PAY=5*AGE
35 PRINT "CENTS PER WEEK"
40 PRINT PAY
50 END
```

Try running the program now. If you like you can change your age or your pay.

Finally we will look at the effect of puncuation in our program. As discussed earlier it is easier to understand a program which is set out clearly and neatly with spacing, puncuation, etc. We must be careful when using puncuation that it is used in the right part of the program.

THE COLON

:

A colon can be used at the end of a program line which will enable us to put more than 1 command on 1 line.

THE SEMI-COLON

;

A semi-colon like a colon is used at the end of a program line to tell the computer to execute the next command, but rather than placing it on a new line it will place it directly after the last command position.

THE COMMA

,

A comma is used to position a value from a command line directly after a print statement so that the numbers or letters directly relate to information in front of it. You might also notice that the comma also positions everything down the lefthand column and every alternate number, letter down the centre.

For Example:

```
5 PRINT "FOR EVERY YEAR IN AGE I GET 5
 CENTS PER WEEK THEREFOR IF I AME 10 Y
EARS OLD I GET 50 CENTS PER WEEK"
10 LET AGE=10
15 PRINT "AGE IN YEARS", :PRINT AGE
20 LET PAY=5*AGE
25 PRINT "CENTS";"PER";"WEEK", :PRINT P
AY
30 END
```

REM

The REMark statement allows you to explain and document your program by inserting comments in the program itself. When the computer encounters a REMark statement while running your program, it takes no action but proceeds to the next statement.

```
10 A=762
20 B=425
30 REM NOW PRINT THE SUM OF A + B
40 PRINT A+B
50 END
60 ---BREAK---

RUN
1187
```

You may use any printable character in a REMark statement. The length of the REMark statement is limited by the length of the input line. If you do not wish to break the word in the middle, press the space bar repeatedly until the cursor returns to the left side of the screen, and then you may begin typing again.

```
10 REM COUNTING FROM 1 TO 10
20 FOR X=1 TO 10
30 PRINT X;
40 NEXT X
50 END
60 ---BREAK---

RUN
 1 2 3 4 5 6 7 8 9 10
```

LET

The LET statement allows you to assign values to variables in your program. The computer evaluates the **expression** to the right of the equals sign and puts its value into the **variable** specified to the left of the equals sign.

```
   AUTO
   10 LET M=1000
   20 LET C=186000
   30 E=M*C^2
   40 PRINT E
   50 END
   60 ---BREAK--

   RUN
     3.4595999989E+13
```

The **variable** and the **expression** must correspond in type: numeric expressions must be assigned to numeric variables; string expressions must be assigned to string variables. The rules governing overflow and underflow for the evaluation of a numeric expression are used in the LET statement. If the length of an evaluated string expression exceeds 255 characters, the string is truncated on the right, and the program continues. No warning is given.

```
AUTO
10 LET X$="HELLOW,"
20 LET NAME$="GENIUS"
30 PRINT X$;NAME$
40 END
50 ---BREAK---

RUN
HELLOW,GENIUS
```

You may use relational opertaors* in numeric and string expressions. The result of a relational operator is -1 if the relationship is true and is 0 if the relationship is false.

* Please see the glossary at the back of this machine

```
   AUTO
   10 LET A=20
   20 B=10
   30 LET C=A>B
   40 PRINT A;B;C
   50 C=A<B
   60 PRINT A;B;C
   70 END
   80 ---BREAK---
```

```
RUN
 20 10-1
 20 10 0
Ready
```

IF THEN

This is the statement which gives basic much of its "intelligence", and give your programs the chance to evaluate conditions and take different actions depending on the outcome.

The word **IF** should be followed by an expression which can include comparisons, numbers, strings, variables, and logical operators. The word "then" must follow on the same line with either another line number, or one or more basic statements.

The expression will usually be an equation which will either prove to be true or false. When the expression is false everything after the word then is ignored and the program continues to the next line in sequence. A true result makes the program branch to the line number following the word then, or execute whatever other basic statements are found on that line.

```
10 CLS :REM CLEARS THE SCREEN
20 INPUT "PICK A NUMBER 1-6";N
30 A=INT(RND(1)*6)
40 IF A=N THEN 100
50 PRINT "WRONG ANSWER":FOR T=1 TO 100
0:NEXT
60 GOTO 10
100 PRINT "CORRECT":END
```

Here the value of the number you choose is stored as N by the computer. At line thirty the computer randomly chooses a number between 1 and 6 then in line 40 compares it to your choice (N). If the two values are not equal (False) line 50 is executed which prints wrong answer for a period of 500 computer counts before going back to the beginning and starting again. Should you guess right the program ends. This program simulates a dice roll.

INPUT

This statement tells the computer to expect information from the operator, and will act accordingly once it is received. When the computer encounters an input statement in a program, it produces a question mark on screen and waits for the information.

If the word INPUT is followed by text in quotatin marks, that text is printed on screen in front of the question mark.

After the text comes one or more Variables separated by commas. This Variable is where the computer stores the information it is about to receive.

The Variable can be any legal variable name, and you can have several different Variable names each for a different required input.

For example:

```
10 CLS
20 INPUT A
30 INPUT "NOW ENTER TWO NUMBERS";B,C
40 D=A*B+C
50 PRINT D
60 END
```

Here in line 20 all Sega will do is produce a question mark as a prompt and wait for a numeric input, which it stores as Variable A in line 30, by putting words in quotation marks. Sega can ask for a specific information, and by giving more than one variable name, separated by a comma, you can be required to enter more than one amount. Each ammount must be entered separately with CR key.

Line 40 asks Sega to take the amounts stored as Variables perform a mathematical calculation on them, and store the result as Variable D. Line 50 asks Sega to give us the figure stored as D by printing it on the screen.

Now change Line 40 as follows:

```
        40 D=A*(B+C)
```

Entering in the same numbers will now cause a different solution. By putting a part of the calculation in brackets, you have told Sega to perform that part of the calculation first, before applying it to the rest of the equation.

To use words, as well as numbers you must have the information stored as a strings. For example.

```
10 CLS
20 INPUT "TYPE IN A WORD";A$
30 PRINT "FIRST WORD IS --- ";A$
40 INPUT "TYPE ANOTHER WORD";B$
50 PRINT "SECOND WORD IS --- ";B$
60 PRINT A$+B$
70 END
```

Input is used in all programs which require the same calculations to be performed on differing amounts, therefore, calculating interest rates, insurance premiums, area conversion calculations etc. can all be performed quickly and easily with on screen prompts to ensure you enter the correct information in sequence.

Entering in words and joining parts of sentences together is useful for adventure type games programs, or educational grammar exercise programs.

Here in line 20 the word or sentence is stored as a string variable A. The word is then retrieved and displayed by the print statement in line 30, after whatever is put in the quotation marks. Line 40 stores the second word as string variable B and line 50 displays that word. Line 60 asks the computer to get the information stored in A$ add it on to B$ and display them together.

GOSUB

This is a specialised version of the GO TO statement. GO TO is the statement which tells the program which line it should jump to during a program, and is not needed with IF-THEN. Using GOSUB enables you to send a program to another part of the program to perform a routine, and when the "Return" statement is reached in the routine the program jumps back to the statement inedately following the GOSUB statement from which it came.

The major use of GOSUB is when there is a small section of a program that is used often by other sections of the same program. By using SUBROUTINES rather than continually repeating the same lines, you save a lot of time and programming space.

```
10 PRINT "WHY KEEP USING"
20 FOR T=1 TO 500:NEXT
30 PRINT "THE SAME LINE"
40 FOR T=1 TO 500:NEXT
50 PRINT "TRY USING GOSUB"
60 FOR T=1 TO 500:NEXT
70 PRINT "FOR THE TIME DELAY"
80 FOR T=1 TO 500:NEXT
```

The program can be done far more efficiently using GOSUB as follows:

```
10 PRINT "THIS PROGRAM DOES"
20 GOSUB 100
30 PRINT "EXACTLY THE SAME "
40 GOSUB 100
50 PRINT "TAKING LESS TIME"
60 GOSUB 100
70 PRINT "USING LESS MEMORY"
80 END
100 FOR T=1 TO 500:NEXT
110 RETURN
```

Each time the program executes a GOSUB the line numbers and the position in the program line are saved in an area of memory called "Stack". The amount of memory available limits the number of GOSUBS available in any one "Nest" to eight.

With our time delay SUBROUTINE you can only have eight GOSUB statements to the one return, if you wish to use that routine more often it must be entered again on a new line, with another Return Statement.

Having more than 8 GOSUBS to Return will result in a GOSUB Nesting error.

THE UNEXPLAINED SEGA COMMANDS. PART ONE

There are some commands that the sega users manual leaves unexplained. The above average user may understand these commands, and be able to use them in their programs, but the majority of people are those who have bought a computer for the first time. In this section of the newsletter we shall look at some of these mysterious commands, explain them, and show how to use them in a simple program. The commands dealt with in this newsletter are,

MOD
POKE
PEEK
VPOKE
VPEEK
DEFN(X)

In a later newsletter we will look at AND, OR, XOR, NOT and CALL.

The MOD Command: The MOD command returns the remainder value which is left over when one number is divided by another. For example, if we divide 240 by 10, the remainder is 0. If we divide 240 by 7, the remainder is 2, ie

7) 240 = Quotient of 34 (7 * 34 = 238)
Remainder of 2 (240 - 238 = 2)

This command is useful where you may want to change the number base of the number in question, that is, when converting a decimal number to a binary value. The MOD command could also be used as the basis for a simple arithmetic program for young children. The following program illustrates such a program.

```
10 SCREEN 1,1:CLS
20 PRINT "SIMPLE ARITHMATIC."
30 PRINT :PRINT
40 PRINT "GIVE ME THE ANSWERS TO THE"
50 PRINT "FOLLOWING QUESTIONS"
60 PRINT
70 PRINT " 22 DIVIDED BY 7,"
80 INPUT "THE QUOTIENT IS ";A
90 INPUT "THE REMAINDER IS ";B
100 IF A<>3THEN PRINT
    "QUOTIENT WAS 3":GOTO 120
110 PRINT "QUOTIENT IS CORRECT":BEEP
120 IF B<>22 MOD 7THEN PRINT
    "REMAINDER WAS 1":GOTO 140
130 PRINT "REMAINDER IS CORRECT":BEEP
140 END
```

The POKE Command: Pages 140, 141 of the Sega Basic manual briefly mentions the POKE command. Please read the section on computer basic's elsewhere in this newsletter before continuing. The Sega computer can address 65536 separate locations, numbered 0 to 65535, or in hexidecimal, as 0000 to FFFF. Each of these locations is called an address, and can hold a number between 0 and 255. The format of the POKE command is as follows,

POKE address, value

where address is a number between 0000 and FFFF, and value is a number between 0 and 255. The following conditions for the Sega must be obeyed,

DO NOT POKE ADDRESS'S WHICH ARE LESS THAN '32767' OR '8000'.
DO NOT POKE ADDRESS'S WITH VALUES GREATER THAN 255.

The POKE command inserts the value into the specified address, thus the previous contents of that address are lost. You must be very careful as to what address's you actually poke, in general, use poke only with address values in the range 'A000' to 'FFFF'. Never try to poke the ROM which resides in the range '0000' to '7FFF', as you may cause serious damage to your computer.

The PEEK command: The PEEK command gets the value stored at the specified memory address. Its format is as follows,

PEEK (address) (where address is any number between 0000 and FFFF)

The following program displays the contents of the first ten address locations of the Basic ROM (Read Only Memory).

```
10 SCREEN 1,1:CLS
20 FORX=&H0000 TO &H000A
30 A=PEEK(X)
50 B$=HEX$(A)
60 PRINT "LOCATION ";X;
70 PRINT " DEC = ";A;
80 PRINT " HEX = ";B$
90 NEXT:STOP
```

The program lists the memory location, followed by the decimal and hexadecimal values, for the firstl ten address's of the ROM. PEEK is only suitable for the creation of special programs that manipulate memory, ie for programs such as Basic Merge or Machine Code Utility programs.

THE VIDEO RAM.

The Video Ram (VRAM) is only accessible to the computer via the VDP which uses port &HBE. The computer can read or write to the VRAM by using the commands INP and OUT. There are areas in VRAM which are reserved for certain portions of the screen (refer page 148 of the Basic manual). We shall concentrate on the text screen, as it is the easiest to follow. Each character displayed on the text screen is stored in a location in VRAM, the next character is stored in the next location and so on. Looking at the text screen, the first character is stored at &H3C00. The second character will thus be stored at location &H3C01, and so on for the 960 characters that the text screen can display (24 * 40). The user can use the sega commands VPOKE to place a character on the screen, or VPEEK to read the value of the character at any position on the screen.

The VPOKE Command: The VPOKE command places the specified character value onto the screen at the specified location. The format of the VPOKE command is,

VPOKE address, value

where address for the text screen is in the range '3C00' to '3FBF', and value is in the range '20' to 'FF' (32-255).

Demonstration. Let's move a character across the top line of the text screen.

```
10 SCREEN 1,1:CLS
20 FOR X =&H3C00 TO &H3C00+40
30 VPOKE X,64
40 FOR Y=1 TO 20:NEXT
50 VPOKE X,32
60 NEXT
```

Line 20, 1st character is 3C00, 40 characters in the line
Line 30, Poke the character to the screen
Line 40, A small delay
Line 50, Erase the character

If you need to poke a character or graphic to a specific location, ie X columns across, and Y rows down, then use the formula on page 144 of the Basic manual.

Further examples: You require to poke the 'little ship' to the cursor position 25,10. Look up page 155 of the manual, and find the code to poke. (it's 250).

CURSOR 25,10

Now let's work out the address required. Page 144 tells us to use the following formula,

Address = y * 40 + x + &H3C00

In this case y is 10 and x is 25. However, the note appended to this formula states that the character position actually deviates by two positions, so the real formula is,

Address = (y * 40 + x + &H3C00) + 2

so to poke the little ship we type,

VPOKE (10*40+25+&H3C02), 250

and the little ship appears exactly where it's meant to. The same procedure of adding two to the address must be used when VPEEK is carried out on the text screen. The reason for the extra two is that the text screen is actually 40 characters wide, but only 38 of them are used by the commands CURSOR and PRINT, but VPOKE and VPEEK actually use the full 40 characters available.

The VPEEK Command: VPEEK returns the character value which exists at the specified location. The format of VPEEK is,

VPEEK (address)

where address for the text screen is in the range '0000' — '3FBF'. A typical program which uses VPEEK follows, where a test is made to see if the character being moved has collided with anything. If there is no character at the specified location, the value returned by VPEEK is equal to 32.

```
10 SCREEN 1,1:CLS
20 VPOKE &H3C10,65
30 FORX=&H3C00 TO &H3C00+40
40 IF VPEEK(X)<>32 THEN 100
50 VPOKE X,250
60 FOR Y=1 TO 30:NEXT
70 VPOKE X,32
80 NEXT
90 END
100 CURSOR 2,10:PRINT "BOOM"
110 BEEP1:STOP
```

Note that the use of VPEEK and VPOKE should be used to speed up slow programs.

The DEF FNS(X) Command: This command is useful for generating special functions. The following illustrates how useful this is,

```
10 SCREEN 1,1:CLS:B=10
20 DEF FNS(A)=INT(RND(1)*B+1)
30 FOP X=1 TO 10
40 Z=FNS(B)
50 PRINT "RANDOM NUMBER ";X;" = ";Z
60 NEXT
```

The program generates 10 random numbers in the range 0 to 10. By changing the value 10 in line 30 it is possible to generate numbers in any range. This is handy for games which need to calculate a lot of random numbers, because it takes up less space and you don't have to keep repeating all those int(rnd()) statements. By changing the value of B, or the value in the FNS brackets, you can generate a random integer in any desired range. The value of B in line 20 is derived from inside the brackets of the FNS statement in line 40, which is set to 10 in line 10.

HARDWARE REVIEW

DISK DRIVE:SF6000
RELEASE: September

Sega's new Disk Drive will no doubt be an enormous advantage for all Sega owners who have longed the speedy access of information which comes with disk storage.

In Sega's case that access speed is even quicker than you might expect from older disk systems, due to the latest 3" disk which it uses. This size has rapidly become the industry standard size, due to its superior durability, being totally encased in plastic with a protective metal covered window which slides back on insertion to reveal the magnetic double sided floppy disk, and also due to its large storage capacity.

The size also has a bearing on the access speed, because it is smaller, it revolves more quickly, allowing the information to be found and read that much quicker.

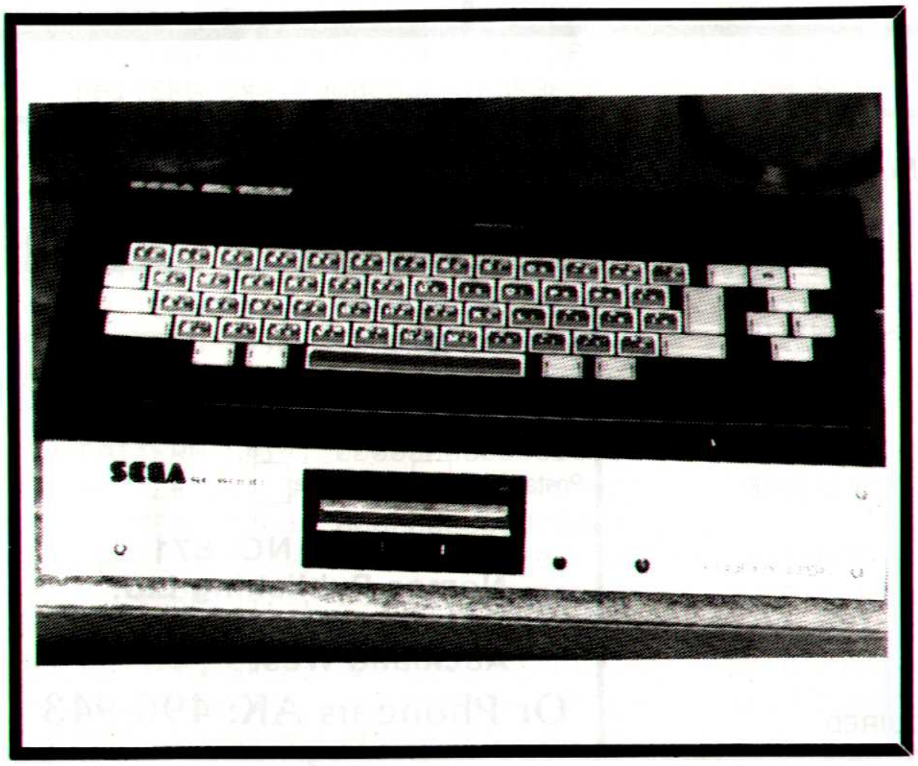
Software availability on disk, initially will comprise of Logo, an excellent educational language, small business packages such as electronic spreadsheet, Database, Word Processing, Mailing List, with invoice management, and a wider variety of business applications following soon after. Sega are also introducing a wide selection of disk games although no titles have yet been released.

The two additional input/output ports greatly broaden the range of peripherals which Sega can control. The RS323C would be required for such items as Modem (telephone couplers) or electronic typewriters, whereas the centronics parallel compatible port would make connecting a wide variety of business printers a quite simple operation.

Basic is loaded into the computer from disk, when the drive is attached through the existing cartridge port. The range of commands is also greatly increased (see below), making such tasks as file handling and creation extremely easy to perform.

CONCLUSION

For the price tag the Sega Disk Drive offers a level of sophistication which makes older, more expensive systems look pale by comparison. It will be an invaluable aid to everyone who wishes to get the most from operating and programming the Sega without the usual delays cuased by cassette loading speeds (what takes 1 minute to load from tape would take around a second from disk).



One word of advice to anyone who is waiting with baited breath, there will be only one shipment from Japan prior to Christmas, so see your stockist early to ensure you are not disappointed.

ADDITIONAL PROGRAMS AND COMMANDS

CSAVE	BOOT
COMSAVE	UTILITY
CLOAD	KILL
COMLOAD	OPEN
MERGE	CLOSE
FILES	PRINT #
LFILES	INPUT#
NAME	COMSET
SET	EOF
INPUT$	

SPECIFICATION

Sega Floppy Disk Unit (Preliminary)

Floppy Disk	Double Side 3 inches 40 tracks
Capacity (Max)	500 K Byte unformatted per disk 163840 Bits (Formatted/ Single side) 327689 Bits (Formatted/ Double side)
Encording Method	MFM
Total Number of Track	40 16 Sectors Per Track
Transfer Rate	8946 FRPI
Access Time	Track to Track 12 MSEC Settling 15 MSEC
Power Supply	220V, 240V AC50/60HZ
Power Consumption	-
Physical Dimension	350(W) x 230 (D) x 55 (H) MM
Weight	
Main memory Ram	64K Bytes (Approximately 22K useable)
I/O Port	
Printer Port	8 Bit Parallel (Centronics Compatible)
RS-232C	CCITT V24, EIA

TRY THIS

DOUBLE SIZE CHARACTERS

```
10 SCREEN2,2:CLS
20 PRINTCHR$(16):PRINT"HELLO"
30 PRINTCHR$(17):PRINT"GOODBYE"
40 GOTO40
```

CURSOR REDEFINITION

```
10 PATTERN C#144,"FC8484848484FC00"
```

LINE — (X, Y) COLOUR

```
10 SCREEN2,2:CLS
20 COLOR5,15
30 FORI=0TO20
40 LINE-(RND(1)*255,RND(1)*191)
50 FORDE=0TO100:NEXTDE
60 NEXTI
70 GOTO70
```

How to make your games and other programs run more rapidly.

1 Remove all spaces between Basic Keywords, variables and numeric characters.

2 Remove all Rem & let statements.

3 Have all sprite movement loops and key input subroutines at the very start of the program.

4 Keep variable names short eg use CT=1 instead of count = 1.

5 Place all instructions, title screens and background graphics at the rear of the program.

6 Use multistatement lines as often as possible.

7 When the program is finished then renumber starting at line 1 with increment of 1 eg renum 1, origin starting line number, 1.

Each issue, we hope to bring you news of meetings which are being held to establish independent local area Sega owners groups.

To establish contacts between potential members we are more than happy to print names, addresses, dates and venues of meetings which are happening up and down the country.

To set everyone off on the right track here are the first addresses:

Tokoroa Sega Users Group
C/o 1 Pio Pio Place
TOKOROA
CONTACT: Geoff 67105

SEGA MAGAZINE

Grandstand are looking for keen personnel capable of assisting with presentations such as computer exhibitions, and with local training seminars in local areas.

Training for both situations will be available. Applications should be addressed to P. Kenyon Box 2353, Auckland.

SPECIAL OFFER

SEGA PROGRAMMERS MANUAL

By B. BROWN.

This manual is written with the more serious programmer in mind, and contains a wealth of information concerning sound graphics, and machine code information, sure to be of interest to the Sega enthuasiast.

Normal retail price will be $29.95. To obtain your copy for $24.95 cut out enclosed coupon and send to.

NAME:.......................................

ADDRESS:.......................................

....

.... PH ....

NO STAMP REQUIRED

I enclose payment: ✓

☐ Visa ☐ Bankcard

| | | | | | | | | | | | | | | | |

Cardholder signature.......................................

Date card expires ....

Cheque ☐ Cheque No.......................................

Postal Order ☐ Postal note ☐

FREE POST NO. 671
Nomac Publishing Ltd,
P.O. Box 39-278
Auckland West

Or Phone us AK: 496-943

SEA BATTLE

Sea Battle is computerised battleships at its best.

You key in the co-ordinates then fire, but be careful where you aim as you only have 10 shots and the enemy ships move as well.

The computer produces a 3 dimensional picture of the battle area and grid showing your hits and misses. When you fire the computer shoots into the 3 dimensional battle area.

```
10 DIM AD(4,9),SH(14)
100 SCREEN 2,2:HI=0
110 COLOR 15,1,,1:CLS
120 GOSUB 3010
130 CURSOR 190,100:PRINT"HI-SCORE"
140 CURSOR 230,110:PRINTHI
150 CURSOR 190,130:PRINT"   SCORE"
160 CURSOR 230,140:PRINTSC
170 REM
180 FOR X=152 TO 232 STEP 8
190 LINE(X,0)-(X,80),11:NEXT X
200 FOR Y=0 TO 80 STEP 8
210 LINE(152,Y)-(232,Y),11:NEXTY
220 FOR X=152 TO 224 STEP 8
230 CURSORX+2,85:PRINTCHR$((X-152)/8+6
5):NEXT X
240 FOR Y=0 TO 72 STEP 8
250 CURSOR235,Y:PRINT9-(Y/8):NEXTY
260 REM
270 LINE(50,90)-(150,90),1
280 LINE-(200,190),1
290 LINE-(0,190),1
300 LINE-(50,90),1
310 PAINT(100,189),5
320 COLOR 15,1
330 REM
340 PATTERN S#0,"0F1F3F7FFFFFFFFF"
350 PATTERN S#1,"FFFFFFFF7F3F1F0F"
360 PATTERN S#2,"F0F8FCFEFFFFFFFF"
370 PATTERN S#3,"FFFFFFFFFEFCF8F0"
380 PATTERN S#4,"0000030F1F3F3F7F"
390 PATTERN S#5,"7F7F7F3F3F1F0F03"
400 PATTERN S#6,"0000C0F0F8FCFCFE"
410 PATTERN S#7,"FEFEFEFCFCF8F0C0"
420 PATTERN S#8,"0000000001070F1F"
430 PATTERN S#9,"1F3F3F1F1F0F0701"
440 PATTERN S#10,"0000000080E0F0F8"
450 PATTERN S#11,"F8FCFCF8F8F0E080"
460 PATTERN S#12,"0000000000000000"
470 PATTERN S#13,"0000010307070301"
480 PATTERN S#14,"0000000000000000"
490 PATTERN S#15,"000080C0E0E0C080"
500 PATTERN S#16,"0000000000000000"
510 PATTERN S#17,"0000000000000101"
520 PATTERN S#18,"0000000000000000"
530 PATTERN S#19,"0000000000008080"
540 PATTERN S#20,"400630125859FB7B"
550 PATTERN S#21,"FF7DF5773F1F0F07"
560 PATTERN S#22,"0170062C65277EFC"
570 PATTERN S#23,"F0F7DEBCFEF0E0C0"
580 PATTERN S#24,"0C1EFF7F00000000"
590 PATTERN S#25,"0000000000000000"
600 PATTERN S#26,"0000000000000000"
610 PATTERN S#27,"0000000000000000"
800 MAG3:GOSUB 4000
1000 REM
1010 GOSUB 2000:REM Key Input
1020 GOSUB 4100:GOSUB 4000
1030 TA=TA-1:GOSUB 3500
1040 X=84:RESTORE 1500
1050 K=KX:IFKX<5 THEN K=9-K
1060 DX=SGN(KX-4.5)*AD(K-5,KY)/30
1070 DY=DY(KY)-131:FOR TH=0TO29:READY
1080 SOUND1,1300-ABS(TH-15)*40,15
1090 X=X+DX:Y=Y+DY:IF Y<0THEN Y=0
1100 SPRITE 0,(X,Y),4*INT(TH/6),15
1110 NEXT TH:SOUND0:GOSUB 4210
1120 E=0:FORI=0TO4
1130 IF SH(I)<0 THEN E=E+1
1140 NEXT I:IF E=5 THENGOSUB 3190:S1=S
1*2:GOSUB 4000:TA=TA+5
1150 IF TA=0 THEN 4500
1160 GOTO 1010
1500 DATA 113,103,93,83,74
1510 DATA 65,57,49,42,36
1520 DATA 31,26,23,20,18
1530 DATA 18,18,19,21,25
1540 DATA 29,34,40,46,54
1550 DATA 62,70,79,89,98
1800 SPRITE 0,(X,Y),20,C:BEEP
1810 FORI=0TO200:NEXTI
1820 SPRITE 0,(X,191),20,0:RETURN
2000 REM DATA INPUT
2010 BLINE (30,12)-(140,41),,BF
2020 CURSOR 30,12:PRINT"X CO-ORD(A~J)
? ";
2030 GOSUB 2180
2040 K=ASC(A$):IFK>64ANDK<75THEN2060
2050 GOTO 2030
2060 KX=K-65:PRINTA$
2070 CURSOR 30,20:PRINT"Y CO-ORD(0~9)
?";
2080 GOSUB 2180
2090 K=ASC(A$):IFK>47ANDK<58THEN2110
2100 GOTO 2080
2110 KY=K-48:PRINTKY
2120 CURSOR 30,33:PRINT"FIRE? (YorN)";
```

```
2130 GOSUB 2180
2140 PRINTA$:IF A$="N"THEN 2160
2150 RETURN
2160 BLINE (30,12)-(140,41),,BF
2170 GOTO 2000
2180 A$=INKEY$:IF A$="" THEN 2180
2190 RETURN
3000 REM
3010 RESTORE 3050
3020 FORI=0 TO 9:READ DA
3030 DY(I)=DA
3040 NEXT I
3050 DATA 180,155,142,133,124,118
3060 DATA 111,105,98,94
3070 RESTORE 3120
3080 FORI=0TO4:FORJ=9TO0STEP-1
3090 READ AD(I,J)
3100 NEXT J,I
3110 TA=10:SC=0:S1=20
3120 DATA 5,5,5,6,6,6,7,7,8,8
3130 DATA 15,16,17,17,18,19,20,22,23,2
5
3140 DATA 25,26,28,29,30,32,34,36,39,4
2
3150 DATA 35,37,39,40,43,45,48,51,55,5
9
3160 DATA 45,47,49,52,54,58,61,65,70,7
5
3170 PATTERN C#92,"0018183C3C3C3C7E"
3180 D(0)=0:D(1)=10:D(2)=-10:D(3)=1:D(
4)=-1
3190 FOR I=0TO4
3200 K=10*RND(1)+INT(10*RND(1))*10
3210 SH(I)=INT(K)
3220 NEXT I
3230 FOR I=0TO3
3240 FOR J=I+1TO4
3250 IF SH(I)=SH(J) THEN 3260
3260 NEXT J,I
3270 RETURN
3500 REM
3510 BLINE(20,55)-(130,63),,BF
3520 IF TA=0 THEN RETURN
3530 CURSOR 30,55
3540 FOR I=1TOTA
3550 PRINT "\";:NEXT I
3560 RETURN
4000 FOR I=0TO4:IF SH(I)<0 THEN 4050
4010 SX=SH(I)MOD10:K=SX
4015 IF SX<5 THEN K=9-K
4020 SY=INT(SH(I)/10)
4030 X=SGN(SX-4.5)*AD(K-5,SY)+92
4040 SPRITE I+1,(X,DY(SY)-10),24,14
4050 NEXT I
4060 RETURN
4100 FOR I=0TO4:IF SH(I)<0 THEN 4160
4110 R=INT(5*RND(1)):K=D(R)+SH(I)
4120 IF K<10ORK>99 THEN 4110
4130 IF SH(I)MOD10=0ANDR=4THEN 4110
4140 IF SH(I)MOD10=9ANDR=3THEN 4110
4150 SH(I)=K
4160 NEXT I:RETURN
4200 REM
4210 C=15:A=KY*10+KX
4220 X1=KX*8+152:Y1=72-KY*8
4230 LINE(X1,Y1)-(X1+7,Y1+7),7,BF
4240 FORI=0TO4:IF  A=SH(I) THEN 4310
4250 NEXT I:GOSUB 1800
4260 COLOR15,1:RETURN
4300 REM
4310 C=8:SC=SC+S1
4320 BLINE(190,140)-(255,150),,BF
4330 LINE(X1,Y1)-(X1+7,Y1+7),8,BF
4340 SH(I)=-100:SPRITE I+1,(0,0),24,0
4350 CURSOR 220,140:PRINTSC
4360 COLOR15,1:GOTO 4250
4500 BLINE (50,100)-(150,150),,BF
4510 CURSOR 60,110:PRINT"TRY AGAIN ?"
4520 CURSOR 70,130:PRINT"Y or N"
4530 FORI=1TO5:SPRITEI,(0,191),24,0
4540 NEXT I
4550 GOSUB 2180
4560 IFSC>HI THEN HI=SC
4570 IF A$="Y" THEN 110
4580 END
```

BRICKS

This program is a version of one of the first Video games. You use a paddle at the bottom of the screen to bounch a ball up into a wall of bricks. The object being to remove all the bricks. This program operates on the text screen. Notice the use VPEEK to search blocks off the screen to see if they are empty or full.

```
10 SCREEN 1,1
15 COLOR 10,1
20 CLS:T=5:SC=0
30 CURSOR 0,0:PRINT"** BRICK OUT GAME
 **  HIGH SCORE:";:PRINT HS
40 CURSOR 21,1:PRINT"SCORE:";:GOSUB 86
0
50 FOR I=2 TO 21:CURSOR 0,I:PRINT"X"
60 CURSOR 35,I:PRINT"X":NEXT I
70 CURSOR 1,2:FOR I=1 TO 34:PRINT"X";:
```

```
NEXT I
80 FOR Y=4 TO 10 STEP 2
90 CURSOR 2,Y:FOR I=1 TO 11:PRINT"██ "
;:NEXT I,Y
100 A=16:CURSOR A,22:PRINT"▔▔"
110 XX=INT(RND(1)*13)+10:YY=11:D=INT(R
ND(1)*2)+2
120 BEEP 1:FOR I=1 TO 150:NEXT I:BEEP
0
130 FOR I=1 TO 300:NEXT I
140 A$=INKEY$
150 IF A$=CHR$(28) THEN 190
160 IF A$<>CHR$(29) THEN 210
170 IF A<1 THEN 210
180 A=A-2:CURSOR A,22:PRINT"▔▔  ":GOTO
 210
190 IF A>31 THEN 210
200 A=A+2:CURSOR A-2,22:PRINT"  ▔▔"
210 ON D GOTO 220,270,330,390
220 PP=VPEEK((YY-1)*40+XX+1+2+&H3C00):
GOSUB 810
230 ON P GOTO 240,480,600
240 GOSUB 780
250 XX=XX+1:YY=YY-1:GOSUB 790
260 GOTO 140
270 PP=VPEEK((YY+1)*40+XX+1+2+&H3C00):
GOSUB 810
280 ON P GOTO 290,500,620
290 GOSUB 780
300 XX=XX+1:YY=YY+1:GOSUB 790
310 IF YY=22 THEN 440
320 GOTO 140
330 PP=VPEEK((YY+1)*40+XX-1+2+&H3C00):
GOSUB 810
340 ON P GOTO 350,520,640
350 GOSUB 780
360 XX=XX-1:YY=YY+1:GOSUB 790
370 IF YY=22 THEN 440
380 GOTO 140
390 PP=VPEEK((YY-1)*40+XX-1+2+&H3C00):
GOSUB 810
400 ON P GOTO 410,540,660
410 GOSUB 780
420 XX=XX-1:YY=YY-1:GOSUB 790
430 GOTO 140
440 GOSUB 780:BEEP 2
450 FOR I=1 TO 300:NEXT I
460 T=T-1:IF T=0 THEN 690
470 GOTO 110
480 X=XX:Y=YY-1:GOSUB 760
490 X=XX+1:D=2:GOTO 560
500 X=XX:Y=YY+1:GOSUB 760
510 X=XX+1:D=1:GOTO 560
520 X=XX-2:Y=YY+1:GOSUB 760
530 X=XX-1:D=4:GOTO 560
540 X=XX-2:Y=YY-1:GOSUB 760
550 X=XX-1:D=3
560 GOSUB 790
570 SC=SC+1:GOSUB 800
580 IF SC=44 THEN 690
590 GOTO 140
600 IF XX=34 THEN D=4:GOTO 680
610 D=2:GOTO 680
620 IF XX=34 THEN D=3:GOTO 680
630 D=1:GOTO 680
640 IF XX=1 THEN D=2:GOTO 680
650 D=4:GOTO 680
660 IF XX=1 THEN D=1:GOTO 680
670 D=3
680 BEEP:GOTO 140
690 FOR I=1 TO 200:BEEP 1:BEEP 0:NEXT
I
700 CURSOR 1,22:PRINTSPC(30)
710 CURSOR 5,22:INPUT "  TRY AGAIN  [Y
/N]? ";B$
720 IF B$="N" THEN END
730 IF B$<>"Y" THEN BEEP 2:GOTO 700
740 IF SC>HS THEN HS=SC
750 GOTO 20
760 CURSOR X,Y:PRINT SPC(3):GOSUB 780
770 BEEP:RETURN
780 CURSOR XX,YY:PRINT" ":RETURN
790 CURSOR XX,YY:PRINT"●":RETURN
800 CURSOR 27,1:PRINT SC:RETURN
810 P$=CHR$(PP)
820 IF P$=" " THEN P=1:GOTO 850
830 IF P$="■" THEN P=2:GOTO 850
840 P=3
850 RETURN
860 CURSOR 27,1:PRINT SC:RETURN
```

SAILING SHIP

This program produces a 3 dimensional picture of a fully rigged sailing ship on the ocean with clouds in the background. It also produces a diagramatic overview of the ship and how it sails are set. This effect has been achieved by first plotting the ship on graph paper and then transferring this information into a basic program using line circle and paint statements.

```
10 SCREEN 2,2:COLOR 1,7,(0,0)-(1,1),7:
CLS
20 POSITION (20,0),0,0
30 LINE (89,36)-(90,48),1,BF
40 LINE (88,69)-(90,78),1,BF
50 LINE (87,96)-(90,110),1,BF
60 LINE (142,42)-(143,48),1,BF
70 LINE (141,69)-(143,78),1,BF
80 LINE (140,99)-(144,110),1,BF
90 RESTORE 600
```

```
100 FOR I=0 TO 65
110 READ X,Y,X1,Y1
120 LINE (X,Y)-(X1,Y1)
130 NEXT I
140 LINE (2,153)-(51,153),15
150 LINE (145,153)-(195,153)
160 LINE (183,141)-(183,165)
170 LINE (135,135)-(183,141)
180 LINE (135,171)-(183,165)
190 LINE (110,132)-(137,135)
200 LINE (110,174)-(137,171)
210 LINE (72,127)-(93,180)
220 LINE (75,127)-(96,180)
230 LINE (72,127)-(75,127)
240 LINE (93,180)-(96,180)
250 LINE (126,127)-(147,180)
260 LINE (129,127)-(150,180)
270 LINE (126,127)-(129,127)
280 LINE (147,180)-(150,180)
290 COLOR 1,11,(0,125)-(220,191),7
300 PAINT (55,112)
310 CIRCLE (88,72),25,1,1.5,.65,.85
320 CIRCLE (82,85),25,1,.7,.67,.94
330 CIRCLE (109,84),7,1,1,.5,.65
340 CIRCLE (84,110),19,1,.8,.56,.9
350 CIRCLE (140,72),30,1,1,.67,.86
360 CIRCLE (170,53),13,1,1,.45,.6
370 CIRCLE (140,85),20,1,.9,.6,.92
380 CIRCLE(177,89),15,1,2,.38,.56
390 CIRCLE (140,125),30,1,.9,.65,.9
400 CIRCLE (70,90),40,1,.8,.5,.6
410 CIRCLE (125,15),26,1,1,.25,.45
420 CIRCLE (125,20),30,1,.9,.25,.4
430 CIRCLE (86,153),5,15,1,.75,.25
440 CIRCLE (140,153),5,15,1,.75,.25
450 CIRCLE (86,153),2,15,1,0,1,BF
460 CIRCLE(140,153),2,15,,,,BF
470 CIRCLE (100,179),60,15,.8,.6,.78
480 CIRCLE (100,179),60,15,.8,.6,.78
490 CIRCLE (100,127),60,15,.8,.22,.4
500 CIRCLE (10,65),15,15,1.5,.5,.75
510 CIRCLE (22,50),14,15,1.3,.5,1
520 CIRCLE (32,68),15,15,1.2,.65,.88
530 CIRCLE (32,68),15,15,1.2,.92,.95
540 CIRCLE (185,55),20,15,1,.5,.8
550 CIRCLE (205,35),15,15,.5,.5,.1
560 CIRCLE (225,55),20,15,1,.7,1
570 COLOR 1,4,(0,115)-(220,125)
580 CURSOR30,0
590 GOTO 1260
600 DATA 57,120,183,120
610 DATA 180,114,183,120
620 DATA 80,113,184,113
630 DATA 184,113,184,110
640 DATA 184,110,80,110
```

```
650 DATA 80,110,45,106
660 DATA 45,106,45,109
670 DATA 45,109,80,113
680 DATA 51,110,57,120
690 DATA 50,107,34,102
700 DATA 34,102,34,99
710 DATA 34,99,3,92
720 DATA 3,92,3,94
730 DATA 3,94,50,105
740 DATA 50,105,50,107
750 DATA 90,12,90,21
760 DATA 143,11,143,21
770 DATA 77,21,100,21
780 DATA 72,48,102,48
790 DATA 63,78,105,78
800 DATA 128,21,158,21
810 DATA 126,48,157,48
820 DATA 122,78,162,78
830 DATA 77,21,72,48
840 DATA 100,21,102,48
850 DATA 72,48,69,72
860 DATA 69,72,66,78
870 DATA 102,48,105,78
880 DATA 63,78,66,108
890 DATA 102,85,102,102
900 DATA 102,102,106,110
910 DATA 128,21,126,45
920 DATA 158,21,159,48
930 DATA 126,48,122,78
940 DATA 162,78,158,57
950 DATA 122,78,120,107
960 DATA 144,78,151,74
970 DATA 161,67,168,62
980 DATA 168,62,194,105
990 DATA 194,105,165,105
1000 DATA 155,105,144,105
1010 DATA 90,12,7,92
1020 DATA 70,52,7,92
1030 DATA 90,42,33,83
1040 DATA 90,42,36,98
1050 DATA 23,93,18,95
1060 DATA 8,92,29,90
1070 DATA 32,90,41,90
1080 DATA 41,90,54,68
1090 DATA 36,99,57,95
1100 DATA 57,95,65,68
1110 DATA 87,75,55,107
1120 DATA 102,48,140,98
1130 DATA 102,48,140,105
1140 DATA 105,78,135,110
1150 DATA 103,85,120,98
1160 DATA 103,89,121,110
1170 DATA 103,93,117,110
1180 DATA 143,15,90,45
1190 DATA 142,45,105,65
```

```
1200 DATA 142,45,105,88
1210 DATA 92,100,88,105
1220 DATA 142,75,103,96
1230 DATA 96,100,88,105
1240 DATA 162,78,174,110
1250 DATA 162,78,170,110
1260 FOR N=0 TO 1000:NEXT N
1270 GOTO 10
```

BRICKS

In this program the objecive is to make your way through the rising "Rocks" without touching or moving into any of them.

Notice Line 90. VPEEK has been used to scan the square about to be moved into to see if it is a "Rock" (AscII code 236 see pg 154-155 of the III Basic Manual).

```
10 GOTO 130
20 CLS
30 C=INT(RND(1)*28):D=INT(RND(1)*10)
40 E$=INKEY$
50 IF E$=CHR$(28) THEN X=X+1
60 IF E$=CHR$(29) THEN X=X-1
70 IF X<0 THEN X=0
80 IF X>36 THEN X=36
90 IF VPEEK(&H3C02+200+X)=236 THEN 150
100 CURSOR X,Y:PRINT B$
110 CURSOR 0,23:PRINT TAB(C);A$;SPC(D)
;A$
120 SC=SC+1:GOTO 30
130 X=10:Y=5:B$=CHR$(43):A$=CHR$(236):
LIVES=3:SC=0
140 GOTO 20
150 CURSOR X,Y:PRINT CHR$(229):LIVES=L
IVES-1
160 IF LIVES=0 THEN 190
170 PRINT "BANG!!!      YOUR SCORE IS ";S
C
180 FOR DE=1 TO 500:NEXT DE:GOTO 20
190 CLS:PRINT "GAME OVER      YOUR SCOR
E IS ";SC
```

READERS' LETTERS

Dear Editor

Perhaps you could advise me on one matter: I have somewhere around 3000 books which I wish to catalogue by author, name and type. What sort of programme would I need to accomplish this.

A G Van Rysewyk

EDITORS REPLY

You would require a disk drive system as this allows you to cross reference information and as the access time is very much quicker than on cassette. (Please see the editorial on the disk drive in this magazine).

Dear Editor

I think I have a possible computer fault.

My graphic screen goes off the left hand side of the T.V. screen.

Psets below 6 cannot be seen in the X axis. A correction factor of 1.3 added to circles gives a true circle.

The T.V. test pattern gives a perfect circle, square output.

T.V. is National Quintrix.

L H Brooshooft

EDITORS REPLY

Your computer is not faulty, it is the adjustment of your television set which causes the first 6 pixels to be "lost". The only way to rectify this is to have your T.V.'s horizontal position adjusted.

We find that to get a perfect circle that you require a height to width ratio between 1.2 and 1.3 depending on your T.V. however, what is thinks is however as a dot or pixel on a T.V. set is not a true square but an oblong. The circle is squashed.

Dear Editor

I would like to know how to detect sprite collisions without having to check all the co-ordinates of every sprite. Is there any easier way to detect collisions.

A P McDonald

EDITORS REPLY

It is possible to detect a collision between any spritesl using these lines of programming.

20 IF (INP(&HBF) AND 32) = 32 THEN GOSUB 50
50 REM SPRITE COLLISION ROUTINE.

Unfortunately line 20 cannot be made to scan for any two particular sprites.

Dear Editor

I own a Sega and am most pleased with the machine and what it can do. I would like to be able to run it off a monitor (I see there is a Video plug at the back of the computer). Could you tell me the connections of this plug so that I can make a cord up.

S Cassidy

EDITORS REPLY

The list below, of the used port connections should enable you to use a video monitor with your Sega.

PIN NUMBER	FUNCTION
1	AUDIO
2	GND
3	VIDEO
4	GND
5	GND

Dear Editor

Could you please help me with the following question:

How do you access the 225 colours mentioned in your advertising.

R Sloane

EDITORS REPLY

To obtain the 225 hues of colour please try the program listed below. The print statements on lines 120 & 140 contain ASCII code 144 (page 154-155 of the level III manual). That is the shaded block found on the U key. You must insert this character repeated between the quotation marks or the program will not work.

Dear Editor

I would like to know how to get the Sega to scroll.

B Curry

EDITORS REPLY

On the text screen printed characters scroll automatically off the top of the screen when it is full.

The program below scroll a single line **down** the screen.

It is not possible to scroll a word on the graphics screen using a print statement (Machine Code is required). However it is possible to define words or letters as sprites and move them up, down and around the screen.

CASSETTE SOFTWARE

EDUCATION

Learning Alphabet

Picture book type graphic teach children the alphabet in a fun way, designed to hold the attention longer with bright colours, movement and sound.

Learn to Count

Number recognition and elementary counting to prepare young minds for basic mathematics.

Shape & Colour Quiz

Random objects in varying colours and numbers must be recognised and counted. Ideal for shape and colour recognition and for learning to add.

Addition Subtraction Multiplication

Three separate programs with one common theme. A classroom teacher who puts forward problems as they would be presented in the classroom. The computer will give help when asked in solving the problem, and will help when incorrect answers are given. Five difficulty levels take children from age 5 to 12 through maths which is fun to do on a computer, and is bound to assist in this subject in school.

SPELLING

This program requires you to enter in a series of words which the computer will flash on the screen briefly at random, to be copied by the child learning to spell. This enables the parent or teachers to tailor the level of difficulty to suit all ages, and ability.

ROCKET MATHS

Select program range, either ADDITION, SUBTRACTION, DIVISION, MULTIPLICATION. Difficulty level 10 to 20, then solve the problem on the screen while guiding your rocket ship across the screen into one of the six ports showing a solution, only one of which is correct. It's a race against time to solve six problems as quick as possible to record your name on the top 10 high score board (ages 7-14).

Watch Me Draw

Control a paintbrush on the screen by using the Joystick, dip into the different colour pots to change colours. Hours of fun doodling and drawing on your T.V. screen.

Typing Tutor

See applications software.

CASSETTE SOFTWARE (ENTERTAINMENT)

Mars Adventurer

An adventure game where your task is to escape from being marooned on the planet Mars.

Enterprise Escape

An adventure game racing against time to escape from a stricken starship.

Mars Mobile

Arcade type action game to manoeuvre across a planet's surface shooting down attackers from above.

Towers of Hanoi

A mind challenger used by ancient monks in Hanoi to move rings of varying size from one pole to another in a set sequence in the least number of moves. Challenge the computer to a battle of wits. With nine levels of difficulty.

Hangman

The classic hangman word challenge featuring three difficulty levels and computer graphics. Lots of fun and educational value.

CASSETTE SOFTWARE (APPLICATIONS)

File System

Allows you to enter between 300-800 pages of information about customers, or personal record keeping. This can be stored on tape and read adjusted, or transferred to paper via the printer.

Files can also be searched by the computer to give details of all files which contain certain pieces of information, ignoring those files which are not required.

Typing Tutor

Gives word and sentence tests to increase your typing skills. It monitors your words per minute performance and automatically displays the information as a bar chart every tenth test.

Cheque Book Reconciliation

Helps you check your monthly bank statement to ensure you pay only for what you have purchased. An invaluable home financial aid.

Loan & Mortage Calculator

A financial package to assist you in making the right decisions when taking out a loan for homes, cars, etc. Perform 'What If" calculations to ensure you pay the least amount of interest possible.

Graph & Chart Presentation

This enables the user to display figures and information quickly and easily in graphic form as either bar, line or pie charts. A frequently used easy to read display for presenting data.

Music Demo

Illustrates the capabilities of the excellent music cartridge, (which you require to run the cassette) and contains 10 pieces of music.

Disassembler

Lists sections of machine code memory on screen as mneumonics, an advanced programming aid.

NEW TITLES

June Releases

Golf (Cartridge)

Possibly the most realistic game of golf you're likely to play without the risk of getting wet. A must for any golf enthusiast.

Exerion (Cartridge)

A superb fighter plane acton game. 3D graphics, full screen aerial fighter control, huge arcade success with near identical computer conversion.

Teach Yourself Basic Game Programming (Cassette)

Learn how to program while creating your first game. A tremendous introduction to program writing with complete manual (support for beginners of all ages).

Account Receivable — Accounts Payable

Two separate programs to keep track of your finances at work on in the home. Totals 20 transactions each batch, stores information back onto tape for storage and retrieval.

Mailing List

Record information of friends or customers addresses, prints out a mailing list at the touch of a key with printer attached.

City-Lander

Your mission is to land your spacecraft in the under-ground city on landing pads to score points. On scoring 100 points you move to screen 2 to manoeuvre through changing passages and land on the mobile landing pad.

PROGRAM DISSECTION

```
10 SCREEN 2,2: CLS
20 FOR X = 0 TO 255 STEP 4
30 LINE (X,0) - (255=Y,191),1: NEXTX
40 FOR Y = 191 TO 0 STEP -3
50 LINE (255,191-Y) - (0,Y),1: NEXTY
60 GOTO 60
```

Line 10: This sends the computer to its graphic screen (screen 2,2) and clears it (CLS).

Lines 20 & 30: This is a for next loop that steps the variable X from 0 to 255 adding 4 to X each time the command next X is encountered. When X is greater than or equal to 255 the computer exits the loop and goes to the following command or statement (in this case Line 40).

The line statement draws a line each time the loop is repeated, "thus it draws a series of lines from the top to the bottom of the screen. The lines first co-ordinate (at the top of the screen) moves from left to right as X is increased by the loop while the second co-ordinate (at the bottom of the screen moves from right to left (this is because X is subtracted from 255, so as X increases the second co-ordinate moves along the bottom of the screen from right to left).

Lines 40 & 50: Essentially these perform the same function as Lines 20 & 30 except it steps the line up and down the right and left hand side of the screen each time the loop is executed.

Line 60: The GOTO command sends the computer to Line 60 (in other words to itself, so it is now in a never ending loop until the program is stop with the break key).

NOW ADD TO THE PROGRAM

```
60 C=C+1:IF C=15 THEN C=1
70 COLOR,C,(0,0)-(255,191),C
80 GOTO 60
```

Line 60: This increases the variable C by 1 and tests to see if C equals 15. If C does equal 15 then C becomes equal to 1 but if C does not equal 15 then the computer moves on to Line 70 without changing C to equal 1.

Line 70: This colours the background behind the pattern of lines drawn in Lines 20-50. As there are co-ordinates specified the computer colours the whole of the screen in colour C, as well as the top and bottom of the screen in colour C.

Line 80: Returns to Line 60 so the computer is in a never ending loop and can only be stopped with the BREAK key.

SEGA'S YOUNG PROGRAMMERS

TODAY NEW ZEALAND, TOMORROW THE WORLD

Over the past few months the list of program writers who have produced software which has been put into production, has grown steadily, with the software itself, receiving great acclaim from all who have received it, and as the list grows, the age of the programmers drops.

One of our latest inclusions is also our youngest. At 13 years old, John Perry recently received recognition of his "City Lander" game by the T.V.N.Z. Top Half team, when he appeared on the program on 18th June. He received his computer, as did many people, as a present for Christmas, and very quickly learned how to operate, and within a month was producing basic games, which he continually improved upon, to the point where "City Lander" became challenging and fast enough to keep anyone playing, enthralled for hours.

Others such as David Harvey, enroll assistance from classmates who have a particular flair for design or the technical aspects of assembling the game structure, before he sets about putting the ideas into computer language. The results are quite superb. His previous experience with computers was with the TI 99/4A, which he also produced programs for, but found the market too cut throat to be worthwhile. David's software will be produced with the team's "Scorpion Logo" present on all titles, and his first release will be "Cube It".

The fact that we have people like Michael Howard and Andrew Snowball, who are capable of producing titles for us, almost by request, as well as their own capabilities and imaginative production, means that the programs which are available in New Zealand, are relevant to the New Zealand way of life. They are programs which can teach children's school subjects in the ways which they are accustomed to in the New Zealand classrooms: The feedback for programs which are in demand locally, can be produced quickly to satisfy demand.

Not only are we talking about New Zealand, we are also already making inroads into Australia. Some of our programs have already been taken on by Sega's Australian distributor, and have proven very popular.

In the near future, Sega expect to launch the SC 3000 into the British market, which will also be thirsty for programs which could be supplied from the minds of the New Zealand elite corps of Sega specialists.

Once again, New Zealand men and women have the opportunity to make the rest of the world sit up and take notice of a small country with big talent for innovation.

Sega's state of the art technology and worldwide new release means mixed blessing for any company who undertake the launch of such a computer in any world market.

The positive benefits are the advances made by technology which mean that your machine is usually far more advanced, has more power, and is less expensive than older existing models.

Against you, is the stack of software which is available for more established computers, the literature which is built up over the years, and the fact that some people buy a name they recognise, rather than checking out what is really being offered.

In New Zealand, we have been particularly lucky in the way that people have got behind the Sega and made it such a huge success, for two main reasons:-

1. Word of mouth is a very powerful medium in the towns and schools. News travels fast in the communities and when you have a real winner, everyone gets to know fast.
2. New Zealand seems to have a great wealth of young people who are capable of producing world class software which is filling the gap created by the computer not yet being available in America and the United Kingdom.

SEGA CARTRIDGE SOFTWARE LIST

N-SUB

You are the captain of a powerful submarine under attack from all sides. Destroy enemy submarines before they attack you, attack battleships and cruisers above as they drop depth charges and deadly torpedoes. 1-2 players.

YAMATO

Cruise across the mighty ocean on the gun deck of a fearsome battleship, trap enemy vessels in the sights of your deadly cannon, guide sea to air missiles to knock out enemy aircraft, but beware torpedoes and heat seeking rockets, move quickly to survive. 1-2 players.

SINBAD MYSTERY

Move Sinbad through a complex 3-D maze, avoid capture, solve the mystery, who can tell what lies in store for you.

CHAMPION TENNIS

Pit your skill against the computer or a friend in a tennis game of superb realism. Control each shot as he or she plays forehand or backhand, volleys, lobs, dropshots, moves around the court in full 3-D excitement.

CHAMPION BASEBALL

3-D realism as you play to win against friends or the computer. Swerve that pitch, swing the bat, chase that fly ball. Home run!!! Great shot.

SAFARI HUNT

Move the hunter stealthily through the paths in the bush. Stalk your prey using the instinct of a big game hunter, take careful aim to bag your prize, but beware the jungle is a dangerous place of many surprises.

MONACO G.P.

Grand Prix action. As your car races through a winding circuit, you must pit your wits against mind-boggling obstacles, dangerous opponents, ice, tunnels, chicanes, and split roads.

Be ready to jump over damaged bridges and avoid ambulances rushing to the aid of less skilful drivers.

MUSIC

This amazing cartridge allows you to access the computer's ability to play music, with the computer's keyboard, taking on the role of a piano or organ keyboard. An overlay card is provided to simplify the operation, and as you play, the music is printed automatically on your T.V. screen. This enables you to not only learn to play music, but to read and write and arrange music, at the same time.

POP FLAMER

A fast and furious multi-screen maze game featuring a flame throwing mouse. Avoid capture by incinerating the chasing monsters, avoid the paralysing rays. Drink from the power well and turn into Super-Mouse to speed you to victory.

PACAR

A fast action 3-D maze game which puts you in control of a racing car pursued by an ever increasing number of chase cars. Speed through the changing tunnels to avoid collision. Score bonus points for erasing power dots.

BORDER LINE

A multi-screen action game. An armoured jeep moves through dangerous alleys, shooting down flying attackers, then cuts a trail through the bush to confuse enemy tanks. It searches to destroy you before you can knock out the power supplies and eventually, reach and destroy the heavily guarded nerve centre.

CONGO BONGO

Big arcade fun as you move through a myriad of adventure screens, jumping rivers, dodging coconuts, avoiding poisonous snakes, constantly chasing the elusive gorilla. 1-2 players.

STAR JACKER

Pilot your fleet of star ships through space, shoot down attackers avoiding enemy fire, make as many runs as you can down the length of the enemy star base. Dock with your mother ship to re-fuel and get reinforcements. 1-2 players.

VIDEO FLIPPER

The most realistic pinball game you're ever likely to see on your own television. A must for every pniball wizard.

GLOSSARY

Accessory Devices — additional equipment which attaches to the computer and extends its functions and capabilities. Included are preprogrammed cartridges* and units which send, receive or store computer data. These are often called peripherals.

Array — A collection of numeric or string variables, arranged in a list or matrix for processing by the computer. Each element in an array is referenced by a subscript* describing its position in the list.

ASCII — The American Standard Code for Information Interchange, the code structure used internally in most personal computers to represent letters, numbers, and special characters.

BASIC — an easy-to-use popular programming language used in most personal computers. The word BASIC is an acronym for "Beginners All purpose Symbolic Instruction Code".

Baud — commonly used to refer to bits per second.

Binary — a number system based on two digits, 0 and 1. The internal language and operations of the computer are based on the binary system.

Branch — a departure from the sequential performance of program statements.

An unconditional branch causes the computer to jump to a specified program line every time the branching statement is encountered. A conditional branch transfers program control based on the result of some arithmetic or logical operation.

Breakpoint — a point in the program specified by the STOP command where program execution can be suspended.

During a breakpoint, you can perform operations to help you to locate program errors. Program execution can be resumed with a CONT command, unless editing took place while the program was stopped.

Bug — a hardware defect or programming error which causes the intended operation to be performed incorrectly.

Byte — a string binary* digits (bits) treated as a unit, often representing one data character*. The computer's memory capacity is often expressed as the number of bytes available. For example, a computer with 16K bytes of memory has about 16,000 bytes available for storing programs and data.

Cartridges — preprogrammed ROM* modules which are easily inserted in the SEGA computer to extend its capabilities.

Character — a letter, number, punctuation symbol, or special graphics symbol.

Command — an instruction which the computer performs immediately. Commands are not a part of a program and thus are entered with no preceding line number.

Concatenation — linking two or more strings* to make a longer string. The "=" is the concatenation operator.

Constant — a specific numeric or string* value. A numeric constant is any real number such as 1.2 or -9054. A string constant is any combination of up to 248 characters enclosed in quotes, such as "HELLO THERE" or "275 FIRST ST."

Cursor — a symbol which indicates where the next character* will appear on the screen when you press a key.

Data — basic elements of information which are processed or produced by the computer.

Default — a standard character or value which the computer assumes if certain specifications are omitted within a statement* or a program*.

Device — (see Accessory Devices).

Disk — a mass storage device capable of random and sequential access.

Display — (noun) the video screen; (verb) to cause characters to appear on the screen.

Execute — to run a program; to perform the task specified by a statement* or command*.

Exponent — a number indicating the power to which a number or expression* is to be raised; usually written at the right and above the number.

For example, 2 =2x2x2x2x2x2x2x2x.

In SEGA BASIC the exponent is entered following the letter "E" in scientific notation*. For example, 2 =2 8; 1.3 x 10 =1.3E25.

Expression — a combination of constants, variables, and operators which can be evaluated to a single result. Included are numeric, string, and relational expressions.

File — a collection of related data records stored on a device; also used interchangeably with device* for input/output equipment which cannot use multiple files, such as a line printer.

Function — a feature which allows you to specify as "single" operations a variety of procedures, each of which actually contains a number of steps; for example, a procedure to produce the square root via a simple reference name.

Graphics — visual constructions on the screen, such as graphs, patterns, and drawings, both stationary and animated. SEGA BASIC has built-in subprograms which provide easy-to-use colour graphic capabilities.

Hardware — the various devices which comprise a computer system, including memory, the keyboard, the screen, disk drives, line printers, etc.

Hertz (HZ) — a unit of frequency.

One Hertz=one cycle per second.

Hexadecimal — a base .16 number system using 16 symbols, 0.9 and A-F. It is used as a convenient "shorthand" way to express binary* code. For example, 1010 in binary = A in hexadecimal, 11111111 = FF. Hexadecimal is used in constructing patterns for graphics characters in the PATTERN subprogram.

Increment — a positive or negative value which consistently modifies a variable*.

Input — (noun) data* to be placed in computer memory; (verb) the process of transferring data into memory.

Input Line — the amount of data* which can be entered at one time. In SEGA BASIC, this is 248 characters.

*See definition in Glossary

Internal data-format — data* in the form used directly by the computer.

Internal numeric data is 8 bytes* long plus 1 byte which specifies the length.

The length for internal string data is one byte per character in the string* plus one length-byte.

Integer — a whole number, either positive, negative or zero.

I/O — Input/Output; usually refers to a device function. I/O is used for communication between the computer and other devices (e.g. keyboard, disk).

Iteration — the technique of repeating a group of program statements; one repetition of such a group. See Loop.

Line — see input line, print line, or program line.

Loop — a group of consecutive program lines which are repeatedly performed usually a specified number of times.

Mantissa — the base number portion of a number expressed in scientific notation*. In 3.264E+4, the mantissa is 3.264.

Mass Storage Device — an accessory device*, such as a cassette recorder or disk drive, which stores programs and/or data* for later use by the computer. This information is usually recorded in a format readable by the computer, not people.

Memory — see RAM, and ROM, and mass storage device.

Noise — various sounds which can be used to produce interesting sound effects. A noise, rather than a tone, is generated by the SOUND subprogram* when Commands 4 and 5 are specified.

Null String — a string* which contains no characters and has zero length.

Number Mode — the mode assumed by the computer when it is automatically generating program line* numbers for entering or changing statements.

Operator — a symbol used in calculations (numeric operators) or in relationship comparisons (relational operators). The numeric operators are +,-,*,/, . The relational operators are , ,=, =, .,= , =,=

Overflow — the condition which occurs when a rounded value greater than 9.9999999999999E+99 or less than -9.9999999999999E-99 is entered or stops.

Output — (noun) information supplied by the computer; (verb) the process of transferring information from the computer's memory onto a device, such as a screen, line printer, or mass storage device*.

Parameter — any of a set of values that determine or affect the output of a statement* or function*.

Print Line — a 38-position line used by the PRINT statement.

Program — a set of statements which tell the computer how to perform a complete task.

Program Line — a line containing a single statement*. The maximum length of a program line is 248 characters*.

Pseudo-random number — a number produced by a definite set of calculations (algorithm) but which is sufficiently random to be considered as such for some particular purpose.

A true random number is obtained entirely by chance.

RAM — random access memory; the main memory where program statements and data* are temporarily stored during program execution*. New programs and data can be read in, accessed, and changed in RAM. Data stored in RAM is erased whenever the power is turned off or BASIC is exited.

Reserved Word — in programming languages, a special word with a predefined meaning. A reserved word must be spelled correctly, appear in the proper order in a statement* or command*, and cannot be used as a variable* name.

ROM — read-only memory; certain instructions for the computer are permanently stored in ROM and can be accessed but cannot be changed. Turning the power off does not erase ROM.

Run Mode — when the computer is executing* a program, it is in Run Mode. Run Mode is terminated when program execution ends normally or abnormally. You can cause the computer to leave Run Mode by pressing BREAK during program execution (see Breakpoint*).

Scientific Notation — a method of expressing very large or very small numbers by using a base number (mantissa*) times ten raised to some power (exponent*). To represent scientific notation in SEGA BASIC enter the sign, then the mantissa, the letter E, and the power of ten (preceded by a minus sign if negative). For example, 3,264E4; -2.47E -17.

Scroll — to move the text on the screen so that additional information can be displayed.

Software — various programs which are executed by the computer, including programs built into the computer.
Cartridges* programs, and programs entered by the user.

Statement — an instruction preceded by a line number in a program. In SEGA BASIC, more than one statement is allowed in a program line*.

String — a series of letters, numbers and symbols treated as a unit.

Subprogram — a predefined general-purpose procedure accessible to the user through the statement in SEGA BASIC. Subprograms extend the capability of BASIC and cannot be easily programmed in BASIC.

Subroutine — a program segment which can be used more than once during the execution* of a program, such as a complex set of calculations of a print routine. In SEGA BASIC a subroutine is entered by a GOSUB statement and ends with a RETURN statement.

Subscript — a numeric expression which specifies a particular item in an array*. In SEGA BASIC the subscript is written in parentheses immediately following the array name.

Underflow — the condition which occurs when the computer generates a numeric value greater than -1E-100, les sthan 1E-100, and not zero. When an underflow occurs, the value is replaced by zero.

Variable — a name is given to a value which may vary during program execution. You can think of a variable as a memory location where values can be replaced by new values during program execution.

*See definition in Glossary

SEGA USERS CLUB SUBSCRIPTION

This is the official Sega Users Club Magazine. Six issues of the magazine are available for $39.95 by subscribing to the Users Club.

In addition to the magazines you will also receive an excellent Sprite Editor cassette and a Music Editor cassette, to assist you in creating your own programs.

To join please make your cheque payable to Grandstand Leisure Ltd and complete the form below.

I would like to get the very best from my Sega SC3000 Computer and wish to join the Users Club. I enclose a cheque for $39.95.

Name ....

Address ....

....

....

Post to Grandstand Leisure Ltd

PO Box 2353

AUCKLAND

Please allow 14 days delivery.

COMPUTER INPUT

WIN A HOME COMPUTER

magazine each month brings you competitions, help in buying computers, regular columns on **Commodore 64, Spectrum, Colour Genie, Sega, JR 100, VIC 20 and ZX81** computers. We review the latest Hardware, Peripherals and Software available in New Zealand and feature over 12 pages of Software Listings for you to load on most popular computers.

JOIN THE TEAM!

ERIC McCALL	**TOM CHERRY**	Published by:
JACK NOBLE	**MARTIN HALL**	
ASHLEY NOBLE	**FAYE HALL**	
JOEL NIELSEN	**SHAYNE BURBERY**	NOMAC
PHILLIPPA CHAPMAN	**BRUCE JOHNSTON**	**PUBLISHING**
OLWEN WILLIAMS	**PETER FINLAY**	**LTD**
BRIAN BROWN		

SEND ME A 12 MONTH SUBSCRIPTION TO
COMPUTER INPUT MAGAZINE AT THE SPECIAL RATE OF $13.00 (11 ISSUES)

NAME: ....

ADDRESS: ....

....

.... PH ....

Computer Model.. I Own Would Like

COMPUTER INPUT

SUBSCRIBE TODAY ONLY $13
NO STAMP REQUIRED

I enclose payment: ✓

☐ Visa ☐ Bankcard

Cardholder signature..

Date card expires ....

Cheque ☐ Cheque No...

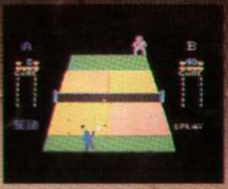
Postal Order ☐ Postal note ☐

FREE POST NO. 671
Nomac Publishing Ltd,
P.O. Box 39-278
Auckland West

Or Phone us AK: 496-943

WHAT GOOD IS THE LATEST TECHNOLOGY IF YOU CAN'T AFFORD IT?

Congo Bongo


Champion Tennis

Monaco GP

Music

BUT NOW, AT ONLY

$399

YOU CAN.

The best technology at the best price. That's the Sega promise. So, if you want to know what state-of-the-art really means, familiarize yourself with the new standards of performance set by Sega.

The latest Sega SC 3000 will keep the family entertained for hours with its vast selection of arcade games. Enjoy the challenge of these thrilling games, recreated on your own TV screen with colourful, high-definition, quality graphics.

The streamlined keyboard ensures the easiest operating ever for those itchy trigger fingers. And if all the famous arcade games aren't hard enough, you can even programme your own.

Because the Sega SC 3000 is a computer, not just a video games machine, its other diverse uses include being an invaluable educational tool. A means of creative expression. Even an indispensible financial organiser.

Imagine how much more readily children will study their school subjects with the help of the SC 3000. Or how inspired any enthusiastic artist will be by the musical and 16 colour, graphic cartridges available.

And your Sega SC 3000 can grow with your family's changing needs. Joystick, printer and cassette deck are just some of its many low-cost expansion capabilities.

Seeing is believing, so don't buy a home computer without seeing the Sega SC 3000 first. You'll be convinced that you need look no further.

GRANDSTAND

SEGA®

HOME COMPUTERS.

SPECIFICATIONS
— 32 independent sprites
— 48K expandable memory
— 16K video ram
— screen text 38 x 24

THE BRAINS OF THE FAMILY.