Operating Manual for the Orbit 32 Portable Programmer

Stag Programmers Ltd.

Silver Court, Watchmead,
Welwyn Garden City,
Hertfordshire,
AL7 1LT, UK

Tel +44 1707 332148
Fax +44 1707 371 503
Email: sales@stag.co.uk
www.stag.co.uk

803 1142 Issue 3
CIS 2367

Contents

The Orbit 32 is a hand held portable EPROM and EEPROM Programmer which is extremely simple to use. In Local operation, all functions are accessed directly from the keypad in conjunction with menus and prompts displayed on the in-built LCD.

In Remote mode using the optional software, Orbit 32 is controlled from either a Windows or DOS graphical environment.

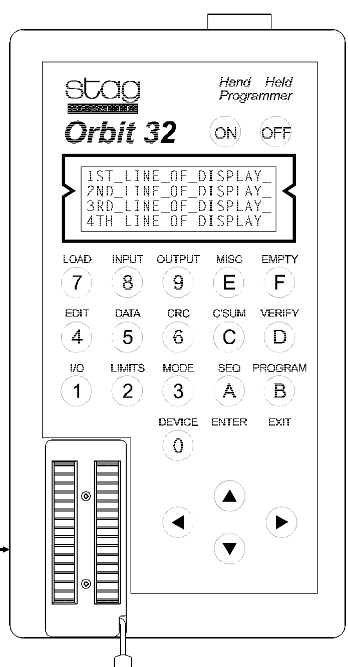
Orbit 32 is powered from internal batteries which can be charged from the supplied mains unit.

! Before powering-up the Orbit 32, please see Section 1.4

Control Panel (Keypad & LCD) 1.1

The Control Panel is located on the top face of the Orbit 32. It consists of an LCD to display status, errors, edit data, etc. and a full hexadecimal keypad with dedicated function keys and cursor keys.

A description of the key functions is given in Section 1.5 - The Keypad.



Power On and Off Keys

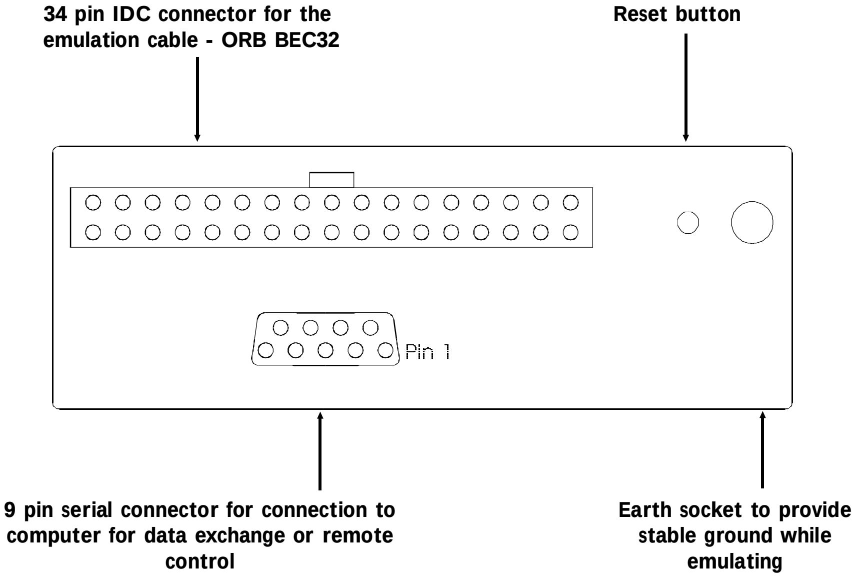
Liquid Crystal Display (LCD)

Functions and Hexadecimal Keypad

32 pin Wide Blade ZIF Socket

Cursor Movement Keys

Rear Panel 1.2



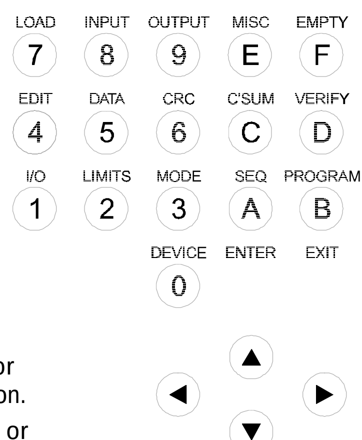
Power Connector 1.3

The mains power unit plugs into a socket on the right hand side of the Orbit 32.

Before Powering Up... 1.4

!	**The unit should NOT be powered up with a device in the socket.**

ON	powers up Orbit 32
OFF	powers down Orbit 32 EXCEPT WHEN PERFORMING A DEVICE FUNCTION - THIS PROTECTS THE DEVICE
CRC	calculates a Cyclic Redundancy Check of data in RAM.
CSUM	calculates the checksum of data in RAM.
DATA	performs additional manipulation functions of data in RAM and allows selection of emulation.
DEVICE	select a device by manufacturer and type.
EDIT	to manually edit data in RAM.
EMPTY	to perform an empty-check on a device.
ENTER	to accept a mode or function setting.
EXIT	to exit from a mode or function.
INPUT	to input data from the serial port into RAM.
I/O	to set all input/output parameters.
LIMITS	to over-ride the default limits for RAM and device data.
LOAD	to load data from a master device into RAM.
MISC	to perform miscellaneous additional functions and provide battery charge status information.
MODE	to set the bit-mode, e.g.: 8, 16 or 32.
OUTPUT	to output data from RAM to the serial port.
PROGRAM	to program data from RAM into a device.
SEQ	to set the programming sequence.
VERIFY	to compare data in RAM against data in a device.



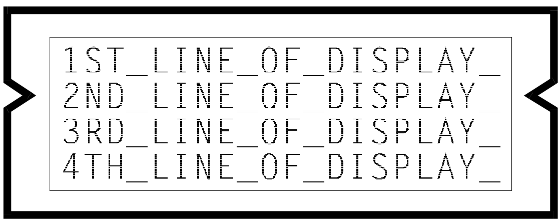
The keys labelled 0-9, A-F are also used to enter numeric data when required.

↑	to scroll data up the screen.
↓	to scroll data down the screen.
←	to move cursor left or display previous option.
→	to move cursor right or display next option.

Local Operation 2.1

All functions are menu driven. Use the ↑ and ↓ keys to select the required option, then press ENTER.

The option which will be selected is always the 2nd row on the display. This is indicated by the pointers to either side of the LCD.

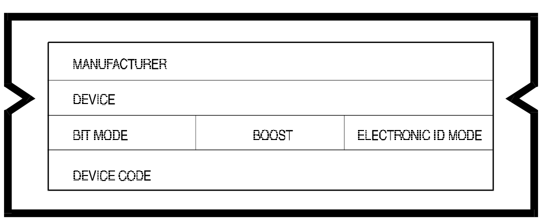


To abort from any menu:

EXIT

The pointing finger symbol denotes a dedicated function key press.

The display consists of 6 message areas - shown in the diagram below.



The BOOST message is only displayed when the Orbit 32 is on boost charge, during which time, no device functions may be executed.

DEVICE

After DEVICE has been pressed:

the device code may be entered directly using 0-9, A-F. (The device codes are given in the Device Support List supplied with the Orbit 32 or any subsequent software upgrades). To edit the code use the ← and → keys. When correct press ENTER.

OR

use the ↑ and ↓ keys to select the required manufacturer, then press ENTER.

Now use the ← and → keys to select the required family or size of device, then the ↑ and ↓ keys for the exact device, finally press ENTER.

See also Section 2.4.3 - Electronic Identifier

Device Limits 2.1.2

All device functions (e.g. Load or Program) have 3 associated parameters:

DEV START	the device address from which the function should start;
DEV STOP	the device address at which the function should stop;
RAM START	the RAM address from which the function should start;

These are also used when calculating the checksum and CRC, and can be altered by the user.

LIMITS

Enter the addresses (in Hexadecimal) using 0-9, A-F.

The cursor can be moved using the ↑ and ↓ keys. When correct press ENTER.

If invalid addresses are chosen (e.g. DEV START higher than DEV STOP) the ENTER key will not let the user out of the function until valid addresses have been selected.

If the EXIT key is pressed the limits will not be changed from their previous values.

> **! The default limits for a device (corresponding to its size) will be used when a new device is selected.**

Setting Up the I/O 2.1.3

The I/O port can be used to input and output data to or from the Orbit 32's internal RAM. It can also be used for computer remote control of the Orbit 32.

See also:	Section 3.5	Transferring Data via the Port and Sections 3.5.1/2
	Section 5	Remote Control.

Setting Up the Port. 2.1.4

I/O

then select **PORT**

A list of parameters is displayed.

These can be scrolled up and down using the ↑ and ↓ keys.

The option - displayed on the second line - may be changed using the ← and → keys.

When the whole menu is set-up as required press ENTER.

SPEED:	The serial port may be set to:	1200, 2400, 4800, 9600, 19K2, 38K4 or 115.2K baud.
PARITY:	Three options are available:	EVEN parity with 7 data bits; ODD parity with 7 data bits; NONE i.e. no parity with 8 data bits.
	Note that for binary transmissions	(e.g. STAG BINARY) NONE should be selected.
STOP BITS:	The number of stop bits transmitted after each byte of data may be set to 1 or 2.	

Select Data Transfer Formats 2.1.5

This function enables the user to select the data format for input and output.

I/O

then select **FORMAT**. A list of available I/O formats is displayed.

The list can be scrolled up and down using the ↑ and ↓ keys. Typically, you might have a choice between: STAG HEX, BINARY, STAG BINARY, ASCII HEX SPACE, INTEL 16 BIT, INTEL 32 BIT, MOTOROLA S-REC.

Select the required format using the ↑ and ↓ keys, then press ENTER.

Bleeper Control 2.1.6

After each function the bleeper will sound to indicate pass or fail (2 bleeps for pass, 5 bleeps for fail). This function may be disabled or enabled.

I/O

then select **BLEEP**

Select disabled or enabled using the ← and → keys followed by ENTER.

You can also have the bleeper sound for each key press. Select disabled or enabled using the ← and → keys followed by ENTER.

Entering Remote Control 2.2

To put the unit into remote control:

I/O

then select REMOTE CONTROL

I/O

To quit from remote back into local mode, power down the unit, then power up with the EXIT key pressed.

See also Section 5 - Remote Control.

Bit Mode 2.3

The user has a choice of bit modes. If an 8-bit wide PROM is selected the options are: 8 BIT, 16 BIT or 32 BIT. If a 16-bit wide PROM is selected then the options are: 16 BIT and 32 BIT.

The bit mode is used in all device functions (e.g. Load or Program), and is also used when calculating the checksum and CRC.

MODE

then select required mode. If the device is 16 bits wide then two bytes of RAM are required to store each device word. This can be done either high byte first / low byte last (default), or low byte first / high byte last.

Having selected the bit mode as detailed subsequently you will then be asked to specify the byte order. To do this, use the cursor keys to make the selection, then press ENTER when ready.

8 Bit Mode 2.3.1

In this mode, assuming no offset is used, each byte in RAM is programmed to a corresponding address in a single target device.

16-bit Mode 2.3.2

Byte Wide Devices
In 16-bit mode the RAM data will be split into ODD and EVEN bytes. When performing any device function (such as Load or Program) other than Empty Check, the Orbit 32 will ask the user which device is required. Press 0 for the device corresponding to EVEN bytes and 1 for the device corresponding to ODD bytes.

Word Wide Devices (Word wide being 16 bits).
It is necessary to set whether the even bytes map to D0 - D7 or D8 - D15 of the device, i.e. which way round the bytes are ordered in the device.

32-bit Mode 2.3.3

This is similar to 16-bit mode.

Byte Wide Devices
Requires the operator to specify 0, 1, 2 or 3 for the device to be operated on.

Word Wide Devices (Word wide being 16 bits wide).
Requires the operator to specify 0 or 1 for the device to be operated on.

Programming Sequence 2.4

This allows the user to define what functions are performed when a device operation is required.

☞ **SEQ**

Sub-menus are selected using the ↑ and ↓ keys, then pressing ENTER.

Pre Program Checks 2.4.1

☞ **SEQ**

then select **PRE-PROGRAM**

Before a device is programmed, the device can be automatically checked with either an empty check or an illegal bit check or neither.

The empty check tests each location of the device (within the specified device limits) to determine whether or not it is empty.

The Illegal bit check tests each location of the device (within the specified device limits) to determine whether it has bits which are programmed and required to be empty by the RAM data.

Select using the ↑ and ↓ keys, then press ENTER.

See also Section 2.5 - Displaying information about failures.

Marginal Verify Testing 2.4.2

☞ **SEQ**

then select **MARGINAL TESTING**

After programming, during illegal bit test, and when the VERIFY key is pressed, the device is verified with the RAM. This can either be done at the manufacturer's recommended V_{CC} voltages (Marginal verify disabled), or at 4.5V and 5.5V (Marginal verify enabled).

Note: Marginal testing also applies to empty testing and illegal bit testing.

Select the required option using ← and → , then press ENTER.

See also Section 2.5 - Displaying information about failures.

Electronic Identifier 2.4.3

SEQ

then select **ELECTRONIC ID**

An Electronic Identifier exists in most EPROM and EEPROM devices. It can be used to check or select a device before load/verify/empty check or program.

Three options are given: check, automatic, none.

NONE will not check the electronic identifier in any way.

CHECK will check that the device in the socket is the same as that selected. If not, the error message WRONG PART will be displayed. NO SIG will be be displayed if no signature can be read from the device.

AUTOMATIC will read the identifier and try to select the correct device code to match. It can only select devices of the same family as that already selected. If a different family's device is inserted then the error message MISMATCHED PARTS will be displayed.

Select the required option using the $\uparrow$ and $\downarrow$ keys, then press ENTER.

Security Fuses 2.4.4

SEQ

then select **SECURITY**

If the device has a security fuse or fuses to secure the data once programmed, the user can select to program them or leave them intact using the $\leftarrow$ and $\rightarrow$ keys followed by ENTER. With devices that have more than one security fuse they can be selected using the $\uparrow$ and $\downarrow$ keys to display the other fuses, ENTER is then pressed once to enter all the fuses.

On some EEPROMs the security feature can be used to make the device write protected.

> **! The security setting is reset to not secure when a new device is selected.**

The display failures function must first be enabled if a failure log is to be displayed about a subsequent device function.

SEQ

then select **FAILURES**

then press ← or → to toggle the function on or off, then press ENTER.

If a device fails when the VERIFY key is pressed, the location and data of the failure can be displayed.

When enabled and a failure occurs, the following will be displayed:

VERIFYING
FAIL ADDR= aaaaaaaa
RAM r1
DEV d1

where: aaaaaaaa is the address of the fail;
r1 is the data in the RAM;
d1 is the data in the device;

All values are in hexadecimal.

The next fail is displayed by pressing ↓, or the function aborted by pressing EXIT.

> **!** **While the failures are being displayed the device is powered up and should not be removed from the socket.**

Machine's Statistics 2.6.1

MISC

then select **STATISTICS**

This function will show the following information:

FLASH software revision
(the boot block's software revision is displayed on power up);
the RAM size (in bytes);
the FLASH size (in bytes);

Saving and Restoring the Machine's Set-up 2.6.2

The following information is stored automatically on power down:

The device - manufacturer and type;

all I/O selections;
the mode;
the programming sequence selections.

These settings are automatically restored on power up.

MISC

then select **CHECK BATTERY**

This function will indicate the battery charge level.

Battery Charging and Management 2.6.3.1

Orbit 32 constantly monitors the charge state of its batteries. If the charge level becomes too low, Orbit will automatically shut down to preserve the integrity of its RAM after issueing the following message:

WARNING !!
Batteries Low
Powering Down

Recharging the batteries is achieved by plugging the supplied charger into the socket on the right hand side of the Orbit 32, and then connecting to the mains electricity supply. Boost charge mode will then be entered automatically, indicated in the Orbit's display. This will continue until the batteries are fully charged. The Orbit then switches to trickle charge mode.

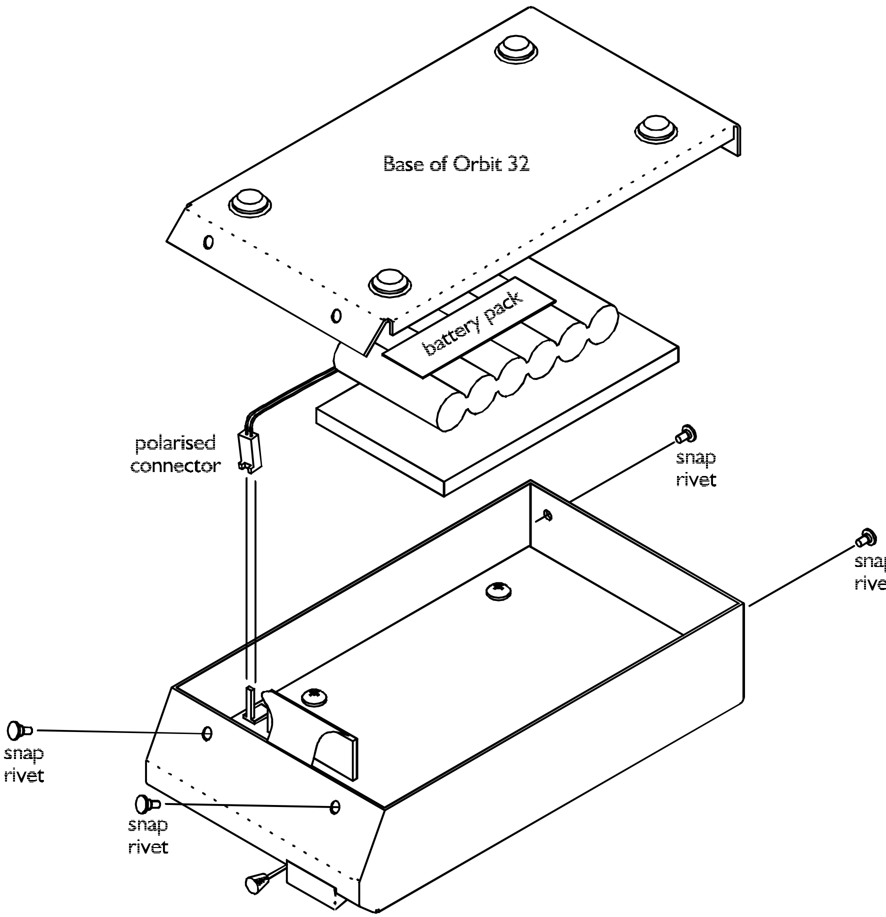
Device operations such as PROGRAM, LOAD, etc, cannot be performed during Boost mode.

Battery Removal 2.6.3.2

Should it become necessary to remove the battery pack, proceed as follows:

1 Ensure there is no device socketed
2 Ensure that data in RAM is not required
3 Power down unit and disconnect power unit, comms cables, etc
4 Invert unit and place face down on clean, smooth surface
5 Lever out the 4 snap rivets - 2 along the front edge and 2 on the rear panel
6 Remove base panel to reveal main PCB and battery pack
7 Disconnect battery pack at polarised connector and lift away
8 Reassembly is the reverse procedure to dissassembly.

DISSASSEMBLY DIAGRAM ON FOLLOWING PAGE



Dissassembly of Orbit 32 to remove battery pack

To update the software, first load the software from STAG into the unit's RAM. This can be done using either the I/O ports or by loading master devices.

MISC

then select **UPDATE**

Orbit 32 will then check that the data in the RAM has the correct format and CRC. If it has the FLASH memory will now be updated. When complete the unit will re-start itself, as if just powered up.

Automatic Power Down If No Keypress 2.6.5

Selects the maximum time allowed between consecutive key presses before Orbit 32 automatically shuts down to conserve power.

MISC

then select **KEY TIMEOUT**

A list of time out values is displayed:

NEVER
5 MIN
10 MIN
15 MIN
20 MIN
25 MIN
30 MIN

This list can be scrolled using the up and down cursor keys.

Select the required option and press ENTER.

RAM Functions 3

Editing the RAM 3.1

This section details the functions which allow the user to alter data in the Orbit 32's RAM.

Listing and Changing the RAM 3.1.1

EDIT

The editor displays 4 addresses in the following format:

aaaaaaaa hh ddd c
where:

aaaaaaaa	is the RAM address in hexadecimal;
hh	is the hexadecimal value stored at that location;
ddd	is the decimal value stored at the same location;
c	is the ASCII for that byte if printable (if not, a □ character is displayed).

The address can be changed using 0-9, A-F and by moving the cursor using ← and →.
To edit the data move the cursor right to the hexadecimal or decimal data fields, then overwrite the data. To edit the next or previous byte use ↑ or ↓.
When complete, press ENTER then EXIT.

Data can be listed by changing the address as above and then pressing ENTER, or by using the ↑ and ↓ to view the previous or next location. Press EXIT when finished.

RAM Data Manipulation 3.2

The following functions can be performed on the Orbit 32's internal RAM:

FILL RAM	BLOCK MOVE
INSERT BYTES	DELETE BYTES
COMPLEMENT RAM	STRING SEARCH

DATA

Select the function required using the ↑ and ↓ keys, then press ENTER.

This function allows you to fill the RAM between selected limits with a selected bit pattern.

DATA

then select **FILL RAM**

On selecting ‘FILL RAM’ the following options are available:

Fill with Zeros	(fill the RAM with 00 hex)
Fill with Ones	(fill the RAM with FF hex)
Fill with Empty	(fill the RAM with the empty state of the selected device)
Fill with Pattern	(fill the RAM with a user defined pattern)

Select the option required using the ↑ and ↓ keys, then press ENTER.

If ‘Fill with Pattern’ is selected the desired pattern should be entered in hexadecimal using the keys 0-9 and A-F.
The ← and → keys may be used to move the cursor to edit the pattern. The ASCII value of the hexadecimal numbers is displayed underneath (if a printable value is entered).

When correct press ENTER.

> **! Note that patterns are only considered legal if they are 2, 4 or 8 hexadecimal characters long - according to selected bit mode.**

All the options will then ask for the address range over which the fill is to take place. The options are as follows:

ENTIRE MEMORY: This function fills the entire RAM with the specified pattern.

DEVICE LIMITS: This will only fill the RAM used for the selected part, taking account of the selected device limits (see Section 2.1.2) and mode (see Section 2.3). Note that if the device address limits are set to only partially cover the device then this function may actually fill non-contiguous regions of RAM.

ARBITRARY LIMITS: This function will enable the user to fill RAM between entirely arbitrary RAM limits. On selecting this option the address limits should be entered in hexadecimal using 0-9, A-F and ↑,↓, ← and → to move the cursor as required.

Select the option required using the ↑ and ↓ keys, then press ENTER.

Move a Block of Data 3.2.2

DATA

then select **BLOCK MOVE**

This function allows data to be moved from one section of RAM to another.

There are no restrictions on the positioning of either the source block or the destination block, other than that they must both fit within the physical available RAM.
Source and destination blocks may even overlap, should this be required.

On selecting 'BLOCK MOVE' the RAM address of BLOCK START, BLOCK END and DESTINATION should be entered in hexadecimal using 0-9, A-F and ↑ ↓ ← and → to move the cursor as required. When correct press ENTER.

Inserting Bytes into RAM 3.2.3

DATA

then select **INSERT BYTES**

This function allows a pattern of bytes to be inserted into RAM at a specific location.

All data at or beyond (i.e. at higher addresses than) the insertion address will be moved upward in memory by the number of bytes inserted. No data bytes are overwritten at the insertion position - instead they move up to make room for the new data.

> **! As a result of this operation, the very last byte(s) in memory will be lost.**

First enter the address in RAM to insert the first byte, use 0-9, A-F, and ↑ ↓ ← and → to move the cursor. When correct press ENTER.

Then the desired pattern should be entered in hexadecimal using 0-9, A-F.
The ← and → keys may be used to move the cursor to edit the pattern. The ASCII value of the hexadecimal numbers is displayed underneath (if a printable value is entered). Up to 32 characters may be entered.

When correct press ENTER.

Deleting Bytes from RAM 3.2.4

DATA

then select **DELETE BYTES**

This function allows a number of bytes to be deleted from RAM. All data at or beyond (i.e. at higher addresses than) the deletion address will be moved down in memory by the number of bytes specified.

Enter the address in RAM to delete the first byte and the number of bytes to be deleted (in hexadecimal), use 0-9, A-F, and ↑ ↓ ← and → to move the cursor. When correct press ENTER.

Complementing the RAM 3.2.5

DATA

then select **COMPLEMENT RAM**

This function allows the data in RAM to be complemented between selected limits. This means that every binary 1 in the RAM data is changed to a binary 0, and vice versa.

The address range over which the complement is to take place should then be selected. The options are:

ARBITRARY LIMITS
ENTIRE MEMORY
DEVICE LIMITS

Select the option required using the ↑ and ↓ keys, the press ENTER.

The three functions available are as follows:

ENTIRE MEMORY: This function complements the entire RAM.

DEVICE LIMITS: This will only complement the RAM used for the selected part, taking account of the selected device limits (see Section 2.1.2) and mode (see Section 2.3).
Note that if the device address limits are set to only partially cover the device then this function may actually complement non-contiguous regions of RAM.

ARBITRARY LIMITS: This function will enable the user to complement RAM between entirely arbitrary RAM limits. On selecting this option the address limits should be entered in hexadecimal using 0-9, A-F and ↑ ↓ ← and → to move the cursor as required.

When correct press ENTER.

This function allows you to search for a string of bytes within specified RAM limits.

DATA

then select **STRING SEARCH**

The desired pattern should be entered in hexadecimal using 0-9, A-F .The ↑ ↓ ← and → keys may be used to move the cursor to edit the pattern. The ASCII values of the hexadecimal numbers are displayed underneath (if printable values are entered).
Up to 32 characters may be entered.

When correct press ENTER.

The address range over which the search is to take place should then be selected.
The options are:

ARBITRARY LIMITS
ENTIRE MEMORY
DEVICE LIMITS

Select the option required using the ↑ ↓ keys, then press ENTER.

The three functions available are as follows:

ENTIRE MEMORY:	This function searches the entire RAM.
DEVICE LIMITS:	This will only search the area of RAM used for the selected device, taking account of the selected device limits (see Section 2.1.2) and mode (see Section 2.3).
ARBITRARY LIMITS:	This function will enable the user to search RAM between entirely arbitrary RAM limits. On selecting this option the address limits should be entered in hexadecimal using 0-9, A-F and ↑ ↓ ← and → keys to move the cursor as required.

When correct press ENTER.

If the string search is successful then the address of the first byte of the string will be displayed.

Press Enter to search for the next occurrence of the string, or EXIT to return to the top level. If no further occurrences are found ‘String not found’ will be displayed, pressing any key will now return you to the top level display.

Emulation 3.3

Emulation mode allows Orbit 32 to emulate a variety of 27xxx type devices up to 4Mbit in size.

DATA

then select **EMULATION**

Once enabled, the Orbit continues to run normally but the target system has priority access to the Orbit's RAM.

Emulation Procedure 3.3.1

Ensure that the emulation cable is configured correctly for the device to be emulated.

Connect the cable to the emulation port on the rear of the Orbit 32.

Plug the cable into the target system. If the socket on the target system is for a 24 or 28-pin device then plug in a turned pin socket of the appropriate size first. This will ensure that the unused pins on the cable do not short to anything on the target system.

Load the emulation data into the bottom of the Orbit's RAM.

Enter emulation mode via the DATA menu.

To modify the data use the Orbit's editor in the usual way. This feature may not work with all target systems. Systems that tie OE or CE low will prevent editing all together and systems that run very fast may make editing unreliable.

To down load a completely new set of data, emulation must be disabled before the download begins.

If a function that is not compatible with emulation is attempted, the message:

ERROR
EMULATING

will be displayed

RAM Emulation 3.3.2

By using the write line from the emulation cable it is possible to emulate RAM. If emulating RAM it is important that the control signals are correctly implemented to avoid contention on the data bus. This means that the OE and WR lines should not be low at the same time.

The emulator cable (ORB BEC32) may be used to easily emulate devices from the 2764 up to the 27040. Smaller devices may be emulated provided certain conditions are met.

All unused address lines must be pulled low including pins that would correspond to VPP or PGM on a small device. This is achieved simply for devices in the range 2764 up by switching the appropriate switches, marked A13 to A19, to the off position.

e.g. For a 27128 A13 would be on and all other addresss switches would be off.

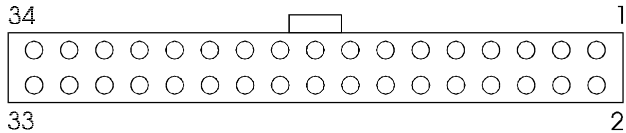
All Vcc switches should be off except for the one corresponding to the size of the selected device. The related address line should be turned off.

e.g. For a 2764 VCC(28) would be on and A17 would be off.

For devices smaller than a 2764 address, switches aren't available so care should be taken to ensure that unused address lines are pulled low.

Emulator Cable (ORB BEC32) Pinout 3.3.4

Signal	32-pin	28-pin	24-pin	24-pin	28-pin	32-pin	Signal
A19	1					32	VCC(32)
A16	2					31	A18
A15	3	1			28	30	A17/VCC(28)
A12	4	2			27	29	A14
A7	5	3	1	24	26	28	A13/VCC(24)
A6	6	4	2	23	25	27	A8
A5	7	5	3	22	24	26	A9
A4	8	6	4	21	23	25	A11
A3	9	7	5	20	22	24	OE
A2	10	8	6	19	21	23	A10
A1	11	9	7	18	20	22	CE
A0	12	10	8	17	19	21	D7
D0	13	11	9	16	18	20	D6
D1	14	12	10	15	17	19	D5
D2	15	13	11	14	16	18	D4
GND	16	14	12	13	15	17	D3



Pin	Signal	Pin	Signal
1	GND	2	D3
3	D2	4	D4
5	D1	6	D5
7	D0	8	D6
9	A0	10	D7
11	A1	12	CE
13	A2	14	A10
15	A3	16	OE
17	A4	18	A11
19	A5	20	A9
21	A6	22	A8
23	A7	24	A13
25	A12	26	A14
27	A15	28	A17
29	A16	30	A18
31	A19	32	NC
33	NC	34	WR

Checksum of RAM Data 3.4

CSUM

This function will return the checksum of the whole RAM, the device's limits of RAM, or arbitrary limits defined by the user. Select the option required. If the device limits are chosen, then a checksum will be displayed, calculated according to the current bit mode.

Cyclic Redundancy Check of RAM Data 3.5

CRC

Cyclic Redundancy check provides a better representation of the RAM data than a checksum as it takes account of the order of the data.
The format is the same as checksum.

Transferring DATA via the Port 3.6

Before loading or outputting data via the port, it is first necessary to ensure the following:

the correct interface format is selected (see Section 2.1.5);
the correct port set-up is selected (see Section 2.1.4).

When using serial the interface, it must be used with either hardware handshaking or the Xon/Xoff protocol.

See also Section 5 which contains information on pin assignments.

INPUT

On pressing INPUT, three further options may be entered:

OFFSET
RAM START
RAM STOP

These are used to define where in RAM to store the data. The OFFSET value is subtracted from the address of the incoming data and the RAM ADDRESS is added on. Data beyond the RAM STOP address will be truncated.

A rotating wheel is displayed to indicate that data are being received.

Transmitting Data TO the Port 3.6.2

OUTPUT

On pressing OUTPUT, three further options may be entered:

OFFSET
RAM START
RAM STOP

OFFSET	is used to generate the first transmitted address
RAM START	gives the location to find the first byte of data, then the transmitted address and the RAM address are incremented until the RAM address equals: RAM STOP

A rotating wheel is displayed to indicate that data are being transmitted.

Device Functions 4

All device functions will perform a connect test to ensure that the device is present in the socket followed by a reverse part check to ensure that the device is the correct way round. If a part is faulty it may also fail this test.

Devices should be inserted towards the front of the ZIF with pin 1 towards the rear of the machine.

> ! **Before a device function is executed the user should ensure that the device used is the same as the device selected. (See also Section 2.4.3 Electronic Identifier)**

> ! **Devices should be inserted into the socket with the ZIF handle up; the handle should then be lowered. The handle should be raised before the part is removed.**

> ! **Devices should not be removed or socketed during a device function.**

At the end of the function the display will indicate whether the function has passed or failed.

Loading the Orbit 32's RAM from a Master Device 4.1

The Master device should be placed in the socket.

LOAD

On pressing LOAD the data in the device will be copied into the Orbit 32's internal RAM.

Remove master device from the socket.

Verify 4.2

This function compares the contents of RAM with the data in the device. Some devices will be verified twice at different V_{CC} values as directed by the manufacturer's specification.
This will also happen if marginal verify is selected (see Section 2.4.2). If a device fails, and the failure function is enabled, failures will be displayed (see Section 2.5).

Ensure device to be checked is socketed correctly

VERIFY

See also Section 2.5 - Displaying information about failures.

Empty 4.3

This function will check that the devices are unprogrammed. If an electrically erasable part is selected, the part can be erased during programming, so new devices may not be shipped in their empty state.

Ensure device to be checked is socketed correctly

EMPTY

See also Section 2.5 - Displaying information about failures.

Program 4.4

This function initiates the automatic programming sequence.

The device is first checked with the pre-program check (see Section 2.4.1), programmed with the data in the RAM to the manufacturer's specification, verified, then the security fuses may be blown if applicable (see Section 2.4.4).

Ensure device to be programmed is socketed correctly

PROGRAM

See also Section 2.5 - Displaying information about failures.

Orbit 32 may be controlled remotely through the serial port.
The unit is put into remote mode by a key sequence in local mode (see Section 2.2).
On power down, the mode of operation is remembered so it will power back up still in remote, unless the selftest fails.
To return to local, either issue the Z command or power up with the EXIT key pressed.

Remote Control Commands 5.1

Remote control commands are case insensitive, and so may be transmitted in either upper, lower or mixed case. Spaces and tabs are ignored. (The only exception to these rules is the remote control 'D' command). In the following table, anything printed in UPPER CASE should be sent literally, while anything in lower case represents a parameter which you should substitute with an appropriate value. Some of the commands cause Orbit 32 to transmit information back to the host, others do not. In either case, Orbit 32's response (if any) is followed immediately by a carriage-return, line feed, status-code (see Section 5.2), carriage-return, line feed, prompt (a greater-than symbol).

S0 manufacturer device	Set the programmer for the specified manufacturer and device. Each of the parameters consists of exactly three hexadecimal characters which can be found in the supplied device support list.
S1 format	Set the I/O format. The parameter is a single ASCII character, and may be one of the following: 4 (Intel hex); 5 (Motorola S-Record); 8 (Stag- hex); 9 (ASCII-hex-space); A (Stag Binary); D (Binary); H (POF); I (Intel 32). Additional formats may be added to this list by Stag at a later date.
S3 security	Set the security flags. The parameter is a hexadecimal number between 00 and FF. Each bit corresponds to one security bit for the currently selected device: bit 0 corresponds to fuse 1, through to bit 6 corresponding to fuse 7. Bit 7 should be set if the device is to be encoded with encryption data. Not all devices support these features.
S4	Fill the RAM between RAM-START and RAM-STOP with the device's unprogrammed state.

SM units sets width	Set the bit-mode. The three parameters are each two-digit decimal numbers. Their meanings are as follows: **units** = number of units per set (a unit is a contiguous region of RAM consisting of a sequence of data-words, each being "width" bits wide); **sets** is the number of identical copies of each unit; **width** is the bit-mode-width, measured in bits. For 8-bit wide devices, legal combinations are: 010116 (16 BIT); 010132 (32 BIT).
SR ram_start	Set the RAM-START address to the specified hexadecimal value. Note that this operation is not carried out immediately, but instead is deferred until after the SE command is issued - therefore you MUST supply these commands in the order SR followed by SE. It is legal to omit the ram_start parameter if you do not wish to modify it but intend to use SE.
SE ram_stop	Set the RAM-STOP address to the specified hexadecimal value. Note that this command also makes permanent the RAM-START address specified by a previous SR command, therefore you MUST supply these commands in the order SR followed by SE. It is legal to omit the ram_stop parameter if you do not wish to modify it but intend to use SR.
SD device_start	Set the DEVICE-START address to the specified hexadecimal value. Note that this command also sets the DEVICE-STOP address to: (DEVICE-START + (RAM-range / bit-mode-width)), so you MUST set the RAM-START, the RAM-STOP and the bit mode BEFORE using this command. It is legal to omit the device_start parameter if you do not wish to modify it but need to modify the device stop address (having previously modified the RAM range).
SO offset	Set the I/O offset to the specified hexadecimal value.
ST margin_mode	Set marginal testing to on (1) or off (0).
SY eid_mode	Set the electronic identifier mode to OFF (0), CHECK (1) or AUTOMATIC (2).
R0	Read the manufacturer and device code. This command outputs a six character hexadecimal number consisting of the Stag manufacturer and device codes for the currently selected device.

R1 Read the interface format. This command outputs a single ASCII character representing the currently selected I/O format, which will correspond to one of the options available for the S1 command.

R3 Read the security fuse setting. This command outputs a two character hexadecimal number representing the current security fuse settings. Each bit corresponds to one security bit for the currently selected device: bit 0 corresponds to fuse 1, through to bit 6 corresponding to fuse 7. Bit 7 will be set if the device is to be encoded with encryption data.

R4 Read the CRC. This command outputs a four character hexadecimal number.

R5 Read the RAM size. The output is a six character hexadecimal number representing the topmost RAM address available.

R6 Read the FLASH software revision number. The output consists of the ASCII string "102-" (which identifies this product as the Orbit 32) followed by two (or sometimes three) fields consisting of decimal numbers. The fields are separated by a period character ('.').

R7 Read the checksum. This command outputs a four character hexadecimal number.

R9 Read device description. Output consists of three fields separated by a "/" character. The first field is the maximum possible (hexadecimal) device address; the second field is the (decimal) device width measured in bits; and the last field is the empty state of the device - ('0' meaning all zeros; '1' meaning all ones; and '2' meaning unknown or indeterminate).

RM Read the current bit mode. Output consists of six digits which comprise three fixed-size fields with no separator. The first field (2 decimal digits) is the number of units per set; the second field (2 decimal digits) is the number of sets; and the last field (2 decimal digits) is the bit-mode-width measured in bits - see also SM.

RR Read the current RAM-START address.
Output consists of six hexadecimal characters.

RE Read the current RAM-STOP address.
Output consists of six hexadecimal characters.

RD	Read the current DEVICE-START address. Output consists of six hexadecimal characters.
RO	Read the current I/O offset. Output consists of eight hexadecimal characters.
RT	Read the current marginal-test setting. Output is ASCII '0' for off, or '1' for on.
RY	Read current electronic identifier setting. Output is ASCII '0' for Off, '1' for Check or '2' for Automatic.
RP	Read FLASH PROM size. Output is a six character hexadecimal number representing the amount of FLASH PROM currently installed in your machine.
P0	Program device with pre-program illegal bit check.
P1	Program devices with no pre-program check.
L	Load device.
E	Empty check device.
V	Verify device.
I	Input data from the serial I/O port using the currently selected I/O format into the Orbit 32's RAM, using the currently selected RAM-START, RAM-STOP and I/O-OFFSET settings.
O	Output data from the Orbit 32's RAM to the currently selected port using the currently selected I/O format, and using the currently selected RAM-START, RAM-STOP and I/O-OFFSET settings.
F pattern	Fill RAM between the current RAM-START and RAM-STOP limits using the specified pattern. The pattern must consist of either 2, 4 or 8 hexadecimal characters.
H number	Sound the horn (beeper) the specified number of times. The single parameter should be a decimal number between 1 and 20.

Dstring Display a string on Orbit 32's LCD. This is the only command for which spaces and tabs are not ignored and, for which, case is important. There should be no space between the D and the first character of the string. The string may not contain line feeds or carriage-returns - but it CAN however contain ANSI X3.64-1979 console escape sequences (so new line can be simulated by transmitting the two bytes $9B followed by $45).

K Wait for any key on the Orbit 32's keypad to be pressed. There is no way of detecting which key is pressed.

Z Exit remote control mode.

M0 Emulation Off

M1 Emulation On

M Read Emulator Status

Status codes returned by the Orbit 32 consist of two hexadecimal characters.
The following responses may be obtained:

00	Command executed successfully.
01	No Blow. Device failed to program.
02	Device failed to verify.
04	Device failed empty test.
05	Device failed connect test.
06	Device found to be reversed or faulty.
08	WARNING: Command executed successfully, but something minor went wrong.
09	Security bit(s) failed to program.
0A	Command parameters incorrect.
0B	Error in inputting data.
0D	RAM failure.
11	Wrong part. Electronic identifier check failed.
12	Mismatched parts. Different parts in different sockets.
13	Illegal or out of range address.
14	FLASH fail. Software in FLASH PROM has become corrupted.
15	No signature. Device could not be recognised by electronic identifier.
1C	Out of memory.
1E	Function aborted.
1F	Unrecognised command or syntax error.

Pin N°	Signal Name		Comment
1	DCD	carrier detect	not used but pulled high
2	RXR	receive data	
3	TXD	transmit data	
4	DTR	data terminal ready	
5	SG	signal ground	
6	DSR	data set ready	
7	RTS	request to send	
8	CTS	clear to send	
9	RI	ring indicator	not used