SYSTEM SOFTWARE
BULLETIN NO. 32
FEBRUARY 9, 1983

TO: Distribution

SUBJECT: BLUE WHALE USER'S MANUAL

Attached is Whale's manual for 1610 development systems.

Please distribute this manual to the 1610 programmers.

DISTRIBUTION

Hugh Barnes	728-12	Douglas Lapp	801-13
Gabriel Baum	729-12	Don Mills	728-12
Ron Carlson	726-12	Mike Minkoff	729-12
Dave Chandler	729-12	Tom O'Brien	729-12
Jan Chodak	801-13	Larry Pumphrey	724-12
Don Daglow	729-12	Keith Robinson	729-12
Richard Decker	726-12	Chuck Rudd	726-12
Bill Fisher	729-12	Tim Scalan c/o. R.Timmins	729-12
Russ Haft	729-12	Joey Silvain	729-12
Jim Haupt	726-12	Ron Surrat	729-12
Bob Hogue	726-12	Rick Timmins	729-12
Les Hutchinson	728-12	Mark Urbaniec	729-12
George Jump	729-12	Mike Winans	729-12
		System Software Group	

Please check M/S

Blue Whale User's Manual

1. Introduction

This manual contains information about Blue Whale Development System's configuration and operational procedures. The Whale provides user up to 20K decles of working memory, as well as features like memory and register manipulation, software breakpoint and single step, uploading capability, interrupt key, checksum generation, and pattern search.

2. Hardware Set-Up

2.1 List of items

2.1.1 Keyboard Component.
2.1.2 Master Component with software interrupt mod.
2.1.3 Keyboard Serial Interface.
2.1.4 1610 Development Program Cartridge (Basic Cartridge).
2.1.5 Universal T-card with 8K decles of RAM.
2.1.6 TV set.
2.1.7 C. Itoh, CIT-101 terminal with AUX port feature.
2.1.8 One (1) long serial cable from CIT-101 to computer system. CIT end is female connector. System connector and "switchness" will have to be determined.
2.1.9 One (1) short serial "straight" cable from serial interface board to CIT-101 AUX port approximately 10 feet. Both connectors are male.

2.2 Memory Space Available

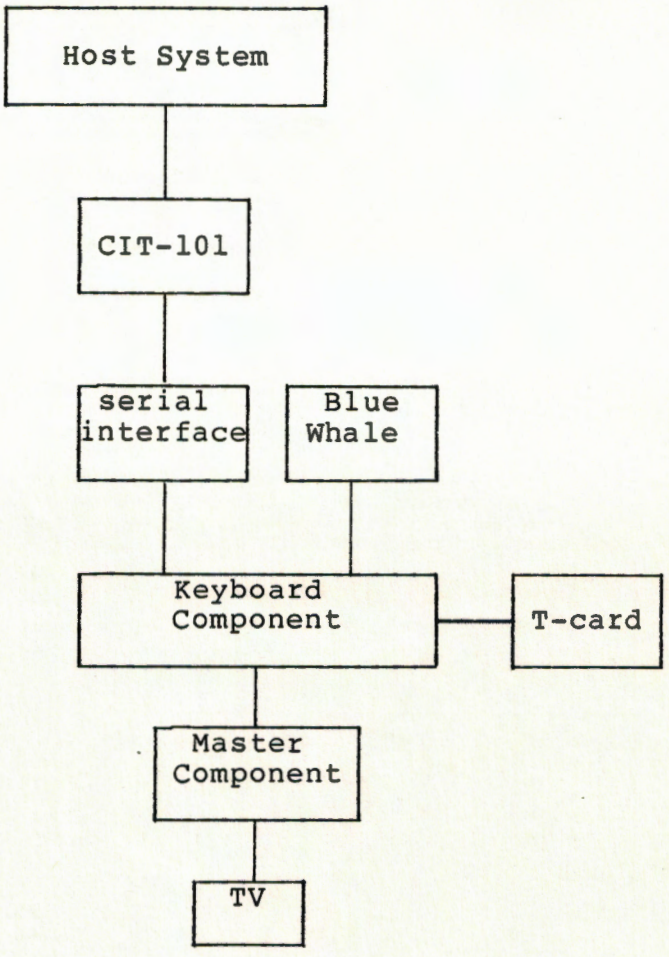
2.2.1 T-card (8K decle)

5000 - 5FFF
6000 - 6FFF

2.2.2 Dual Port RAM (12K decle)

9000 - 9FFF
A000 - AFFF
B000 - BFFF

2.3 Functional Diagram



3. Operational Procedure

3.1 Insure that the CIT-101 is communicating in a normal manner with the host computer. If not, press the Set-Up key and then the reset (0) key.
3.2 Set AUX port to 8 bits per character.
3.3 Reset Master Component.
3.4 Type in "X" <return> on the Keyboard Component.
3.5 The TV screen will display the message
"WELCOME TO 1610 DEVELOPMENT SYSTEM"
"VERSION 5 F"
3.6 Now you are able to perform operations on Whale.

List of commands for the Blue Whale Development System on VAX(TSX)

4. Commands to VAX

The character "%" is the VAX prompt. The addresses are in Hex-ASCII format.

4.1 Blue Whale Format Command

```
% blue file1 > file2
```

Converts the output file file1 from the linker to the Blue Whale format, and directs it to file2.

4.2 Download Command

```
% whale [infile]
```

Downloads infile in Blue Whale format to the Blue Whale. If infile not specified, it directs the video terminal to the Blue Whale.

4.3 Upload Command

```
% upwhale outfile addr1>addr2
```

Uploads the contents of the memory location from addr1 thru addr2 to outfile in Blue Whale format.

5. Commands to TSX

The character "." is the TSX prompt.

5.1 Download Command

```
.orca<return>
 Infile<return>
```

Downloads infile from TSX system to Whale.

5.2 Upload Command

```
.spout<return>
 Outfile<return>
```

Uploads 8K memory, starting from 5000 to 6FFF, to the specified file.

5.3 Invoke Whale Command

.useron<return>

This command invokes the Whale. After that your terminal will b
talking to Whale.

6. Commands to Blue Whale

The character ">" is the Blue Whale prompt. All addresses and dat
in Hex-ASCII format. The letters representing commands are in uppe
case. The brackets "[]" represent for optional parameters.

6.1 Go and Execution Command

>[addr]G<return>

Starts execution from location addr. If addr not specifie
starts at location (1041)H to start cartridge execution.

>P<return>

Starts execution from present program location.

6.2 Single Step Command

>[addr]T

Single step starts at the location addr. If addr not specified
starts at present program location. Since the single step is softwar
implemented, it is not possible to trace code through ROM locations.

6.3 Memory Manipulation Command

>[addr]M

Displays 8 consecutive memory location contents starting from th
address addr. If addr not specified, assumes the current memory pointer.

>addr/datal data2

Alters the memory location addr's content datal for data2. Thi
command terminates with a Line Feed or Carriage Return.

>addr1<addr2,addr3$M

Moves the content of memory location addr2 to addr1, then increment

addr1 and addr2, repeats the move until addr1> addr3. This command ca
be used to fill memory with a pattern of data. First, store the desire
data into addr2. Then let addr1=addr2+1 and do the above command. Memor
from addr1 through addr3+1 will be filled with data. In a similar fashion
multiple decles can also fill memory.

>addr1,addr2V

Displays the Data-IO checksums (low and high) and IMI checksum o
the memory location addr1 to addr2.

>data,addr1,addr2S

Searches memory location with content data from addr1 to addr2, an
displays all matching addresses. If data is specified with one or tw
digits, only the low 8 bits of the 16 or 10 bit memory word is searched for
If 3 or 4 digits are specified, all 16 bits are checked. Control-S can b
used to stop the display.

>ZERO<return>

Zeroes the memory locations 5000-5FFF, 6000-6FFF, 9000-9FFF. Onl
the letter "Z" is echoed back.

6.4 Register Manipulation Command

>RR

Displays the content of the registers.

>n,dataR<return>

Sets the specified register to new value, where n is a digit (0-8
that specifies the register (n=8 for the status flag register) and dat
is the new value in hex-ascii.

6.5 Breakpoint Command

>addr$B

Sets the address addr as a breakpoint. One can set up to eigh
breakpoints. Setting breakpoint in ROM location is meaningless, sinc
the breakpoint is software implemented.

>addr$R

Resets the breakpoint addr.

>$D

Displays all current breakpoint addresses.

>$C

Clears all current breakpoints.

<esc> Key

This break key, whenever applied, stops the user's program displays the content of the registers and returns the control over 161(back to Whale debugger. It only works after a G or P command.

MISCELLANEOUS - These commands are used automatically by system programs and are not needed by most people using the whale to write games.

>addrGO<return>

This command causes the 6502 whale code to branch to a new 650(address. This is used when testing new versions of whale code with blowing more EPROMS.

>addr;

Displays the byte addressed using 6502 address. Is otherwise like '/'.

>addr1,addr2,countU

Uploads data records starting from addr1 and continuing through addr2. Each record has a length of 'count' 16 bit words. Each record begins with a : character. It takes four characters per 16 bit word. The low four bits of each character contains the data. Each character can range from @ABCDEFGHIJKLMNO. Then checksum results from adding up all the 16 bit words and ignoring overflow. The whale will wait for an ack or nak character.

If ack (control-F) is received, the next record is transmitted. If nak (control-U) is received, the previous record is re-transmitted.
This command is used for uploading MR. COLOR files.

>:

Download data to the whale. The next character determines if the downloaded data is in 1610 format or 6502 format. The 6502 format is standard Intel format. Whale format is of the form:

:aaaadd.....;c

The first four characters is the 16 bit address. Successive character pairs form succeeding decles. The ; finishes the data record and a 5 bit checksum follows. Notice that a 5 bit checksum is not adequate for transmission over error prone lines, like telephone lines. If you get even one BAD RECORD message while downloading, something should be fixed. Data is held in the low 5 bits of each character. The whale returns control-F, carriage return, line feed, > if the record was received correctly. Otherwise, control-U is returned.

Bringing up new versions of the whale:

To test out a new version of whale, first use MICEV to download the 6502 version. Do a 20BDGO<return> to start the new 6502 code. This address should be label NXCOM. Then use ORCA to download the new 1610 code and start it with B061G<return>. The new 6502 code module is called BCOM. The new 1610 code is called DDBBGG.

There are several things that must be changed from the debugging version of the whale firmware before burning a set of EPROMS.

1. Modify module B1610.ASM to not JSR DNLOAD.
2. Modify module BEQAT.ASM to start 1610 code at $8800 instead of $B000.
3. Modify module BEXEC.ASM to clear memory to $C000 instead of only to $A000. This step is usually skipped to allow loading new versions of whale into locations $A000-$AFFF new 6502 code and $B000-$BFFF (new 1610 whale code).
4. Change BA6502.CCL to load program into $E000 instead of $2000.
5. Change DDBBGG.CCL to load program into $8800 instead of $B000.