Bob Kahn

Director of Special Projects

Atari, Inc.

Box 427

1196 Borregas

Sunnyvale, CA, 94086

PECEIVED

JUN 1 5 1983

SPECIAL PROJECTS

Dear Bob:

Enclosed are originals and backup for

- Coin-Op FORTH, Swarthmore Disk 2, Disk 2, and QE Disk, containing bootable versions of FORTH and appropriate some screens (see Section I. 1 of the document, "What this is")
- 5 AtaviWriter dishs of discumentation marked "AtaviWriter (1)" to "AtaviWriter (5)".
  - Instructions for printing out document (on the following page).
    - One copy of printent.

el hepe you find I salisfactory. We think it's pretty dann good.

- Regards Gene Klotz

# Instructions For Printing Document

This document is on 5 AtaniWriter disks, marked "AtaniWriter O" to "AtaniWriter O".

Two copies of each are enclosed.

For a complete printout you only need to start printing the files indicated.

Disk Number	Beginning of Chain Files	Sections of Documents
	TITLE, PG	I
(2)	FIG. MAN	1.1
	EXCL. FIG	II.2
3	ATARI, EXT	II. 3 - II.5
	WTOBJOLD,MAN	
4	[V]. \	[V.1 - IV.6
(5)	70.17	D.17
	I	<u>J</u>
	Gen Klotz	6/1/83

# F.I.G. FORTH EDITOR (7/8/80 - SRC)

NOV 1 o 1982

SPECIAL PROJECTS

n k EDIT

- Enter editor vocabulary on screen n.

EDITOR

- VOCABULARY name - used to get to the Editor words.

SCR

- VARIABLE containing the current edit screen (...

L

- List current screen.

N

- List next screen.

LL

- List lower 8 lines for screen editor.

UL

- List upper 8 lines for screen editor.

DOIT

- Take the top 16 lines on the TV display and put into the upper or lower half of the edit SCR. (whichever was last displayed). Use the colleen cursor and edit keys to change the screen. Move the cursor to the DOIT line and hit RETURN.

DO NOT HIT RETURN while editing the LL or UL screens.

SRC DEST COPY

- copy block SRC to block #DEST.

FIRST LAST SHOW

- list all blocks inclusive.

n LIST

- set SCR to n and list the block.

#### Line Edit Functions:

n TL - Type line n and save it in PAD.

n HL - Hold line n in PAD.

n DL - Delete line n - save it in PAD.

n IL - Insert PAD after line n.

n RL - Replace line n with PAD.

n SL - Spread to create blank line n by pushing every thing

down and losing line F.

n BL - Blank line n - not saved in PAD.

n SCR CL - Move line n of Screen #SCR to PAD.

n \$ - Put text following \$ up to RETURN onto line n.

n % - Insert text following % up to RETURN after line n.

FLUSH - RETURNS to FORTH Vocabulary. Writes out changed

blocks.

Currently 7/8/80 8 Load loads the assembler, Ed Logg's Colleen I/O and Graphics words and this Editor.

1+! (addr->)	add one to word at addr
1- (n → n-1)	subtract one from TOS
OSET (addr→)	set word at address to 0
2* (n → n)	mult. TOS by 2 faster than
2/ (n -> n)	div. Tos by 2 24*,241/ spaces
CHH (b → )	type value of LSB of TOS as 2-digit HEX
HH (u → )	type value of TOS in unsigned HEX
CH? (addr →)	type value of byte at addr in HEX
H? (addr )	type value of word at addr in unsigned HEX
J ( → n)	return Index for next outer most DO-LOOP
TBL	;defines ROM table (like DECUS, CALTECH IARRAY)
ALLOC $(n \rightarrow )$	allocate n words in RAM
ARRAY (n -> )	CREATE named ARRAY, n words long, returns addr of first element when exe
s? ( -> )	TYPE contents of param stack, without removing them.
RP ( -> n)	point to TOP of RETURN (control) stack
R? ( -> )	TYPE contents of return stack (non-destructive)
BDUMP (addr ,addr -> )	Display contents of addr to addr (inclusive) always in increments of 8 bytes.

```
TERMINAL INPUT-OUTPUT
                                                  Print number.
                                                 Print number, right-justified in field.
.R
                       n fieldwidth →
                                                  Print double-precision number.
D.
                      (d - )
D.R
                       d fieldwidth →
                                                  Print double-precision number, right-justified in field.
                                                  Do a carriage return.
CR
                        -
                        → )
SPACE
                                                 Type one space.
SPACES
                      (n →
                                                  Type n spaces.
                             )
                                                 Print message (terminated by ").
                        → )
                                                 Dump u words starting at address.
DUMP
                       addr u →
                       addr u →
                                                 Type string of u characters starting at address
TYPE
                      ( addr → addr+1 u )
                                                  Change length-byte string to TYPE form.
COUNT
                                                 True if terminal break request present.
?TERMINAL
                        - f)
                                                 Read key, put ascii value on stack.
KEY
                        - c)
                      (c - )
                                                  Type ascii value from stack.
EMIT
                                                 Read n characters (or until carriage return) from input to address.
EXPECT
                       addr n →
                                                 Read one word from input stream, using given character (usually blank) as delimiter.
WORD
                      ( c -
                            )
INPUT-OUTPUT FORMATTING
                                                  Convert string at address to double-precision number.
NUMBER
                      (addr - d)
<#
                        → )
                                                 Start output string.
                                                 Convert next digit of double-precision number and add character to output string.
                       d \rightarrow d)
                                                 Convert all significant digits of double-precision number to output string.
#S
                      (d - 00)
                                                 Insert sign of n into output string.
                      (nd \rightarrow d)
SIGN
                                                 Terminate output string (ready for TYPE).
                      (d - addr u)
#>
                                                 Insert ascii character into output string.
HOLD
                      (c → )
DISK HANDLING
LIST
                       screen →
                                                 List a disk screen.
LOAD
                                                 Load disk screen (compile or execute).
                       screen →
                                                 Read disk block to memory address.
BLOCK
                       block → addr )
B/BUF
                                                 System constant giving disk block size in bytes.
                        \rightarrow n)
                        → addr )
BLK
                                                 System variable containing current block number.
                                                 System variable containing current screen number.
                        → addr )
SCR
UPDATE
                                                 Mark last buffer accessed as updated.
FLUSH
                                                 Write all updated buffers to disk.
EMPTY-BUFFERS
                                                 Erase all buffers.
DEFINING WORDS
                                                 Begin colon definition of xxx.
                        -
                                                 End colon definition.
VARIABLE XXX
                      (n →
                                                 Create a variable named xxx with initial value n; returns address when executed.
                             )
                       xxx: ( → addr )
                      (n →
                                                 Create a constant named xxx with value n; returns value when executed.
CONSTANT XXX
                        xxx: (
CODE xxx
                                                 Begin definition of assembly-language primitive operation named xxx.
                                                 Used to create a new defining word, with execution-time "code routine" for this data
;CODE
                                                   type in assembly.
                                                 Used to create a new defining word, with execution-time routine for this data type in
<BUILDS ... DOES> does: (
                              → addr )
                                                   higher-level Forth.
VOCABULARIES
CONTEXT
                                                 Returns address of pointer to context vocabulary (searched first).
                           addr )
CURRENT
                                                 Returns address of pointer to current vocabulary (where new definitions are put).
                           addr )
                                                 Main Forth vocabulary (execution of FORTH sets CONTEXT vocabulary).
FORTH
                                                 Editor vocabulary; sets CONTEXT.
EDITOR
                                                 Assembler vocabulary; sets CONTEXT.
ASSEMBLER
                                                 Sets CURRENT vocabulary to CONTEXT.
DEFINITIONS
                                                 Create new vocabulary named xxx.
VOCABULARY XXX
                                                 Print names of all words in CONTEXT vocabulary.
MISCELLANEOUS AND SYSTEM
                                                 Begin comment, terminated by right paren on same line; space after (.
FORGET XXX
                                                 Forget all definitions back to and including xxx.
                        -
                                                 Error termination of operation.
ABORT
                                                 Find the address of xxx in the dictionary; if used in definition, compile address.
' XXX
                        -
                           addr
HERE
                           addr
                                                 Returns address of next unused byte in the dictionary.
                                                 Returns address of scratch area (usually 68 bytes beyond HERE).
PAD
                        → addr
                                                 System variable containing offset into input buffer; used, e.g., by WORD.
                           addr
                        -
IN
SP@
                           addr )
                                                 Returns address of top stack item.
                                                 Leave a gap of n bytes in the dictionary.
ALLOT
                      (n -
                                                 Compile a number into the dictionary.
                      ( n
```

## FORTH HANDY REFERENCE

Stack inputs and outputs are shown; top of stack on right. This card follows usage of the Forth Interest Group (S.F. Bay Area); usage aligned with the Forth 78 International Standard.

For more info: Forth Interest Group P.O. Box 1105
San Carlos, CA 94070.

Operand key: n, n1, . . . 16-bit signed numbers
d, d1, . . . 32-bit signed numbers
u 16-bit unsigned number
addr address
b 8-bit byte

7-bit ascii character value boolean flag C

#### STACK MANIPULATION

DUP	$(n \rightarrow nn)$	Duplicate top of stack.
DROP	(n → )	Throw away top of stack.
SWAP	( n1 n2 → n2 n1 )	Reverse top two stack items.
OVER	$(n1 n2 \rightarrow n1 n2 n1)$	Make copy of second item on top.
ROT	( n1 n2 n3 - n2 n3 n1 )	Rotate third item to top.
-DUP	$(n \rightarrow n?)$	Duplicate only if non-zero.
>R	(n → )	Move top item to "return stack" for temporary storage (use caution).
R>	( → n)	Retrieve item from return stack.
R	( → n)	Copy top of return stack onto stack.

#### **NUMBER BASES**

DECIMAL	( - )	Set decimal base.
HEX	( - )	Set hexadecimal base.
BASE	( → addr)	System variable containing number base.

## ARITHMETIC AND LOGICAL

+	( n1 n2 → sum )	Add.	
D+	( d1 d2 → sum )	Add double-precision numbers.	
-	( n1 n2 → diff )	Subtract (n1-n2).	
*	( n1 n2 → prod )	Multiply.	
/	( n1 n2 → quot ) .	Divide (n1/n2).	
MOD	( n1 n2 → rem )	Modulo (i.e. remainder from division).	
/MOD	( n1 n2 → rem quot )	Divide, giving remainder and quotient.	
*/MOD	( n1 n2 n3 → rem quot )	Multiply, then divide (n1*n2/n3), with double-precision intermediate.	
*/	( n1 n2 n3 → quot )	Like */MOD, but give quotient only.	
MAX	( n1 n2 → max )	Maximum.	
MIN	( n1 n2 → min )	Minimum.	
ABS	(n → absolute)	Absolute value.	
DABS	( d → absolute )	Absolute value of double-precision number.	
MINUS	( n → -n )	Change sign.	
DMINUS	( d → -d )	Change sign of double-precision number.	
AND	( n1 n2 → and )	Logical AND (bitwise).	
OR	( n1 n2 → or )	Logical OR (bitwise).	
XOR	$(n1 n2 \rightarrow xor)$	Logical exclusive OR (bitwise)	

#### COMPARISON

<	( n1 n2 → f )	True if n1 less than n2.
>	( n1 n2 → f )	True if n1 greater than n2.
=	( n1 n2 → f )	True if top two numbers are equal.
0<	(n → f)	True if top number negative.
0=	(n → f)	True if top number zero (i.e., reverses truth value).

#### **MEMORY**

@	( addr → n )	Replace word address by contents.
Ĭ	( n addr → )	Store second word at address on top.
C@	(addr → b)	Fetch one byte only.
CI	(b addr → )	Store one byte only.
?	(addr → )	Print contents of address.
+!	( n addr → )	Add second number on stack to contents of address on top.
CMOVE	( from to u → )	Move u bytes in memory.
FILL	(addrub→)	Fill u bytes in memory with b, beginning at address.
ERASE	(addru → )	Fill u bytes in memory with zeroes, beginning at address.
BLANKS	(addru → )	Fill u bytes in memory with blanks, beginning at address.

#### **CONTROL STRUCTURES**

DO LOOP	do: ( end+1 start → )	Set up loop, given index range.
1	( → index )	Place current index value on stack.
LEAVE	( - ).	Terminate loop at next LOOP or +LOOP.
DO +LOOP	do: ( end+1 start → ) +loop: ( n → )	Like DO LOOP, but adds stack value (instead of always '1') to index.
IF (true) ENDIF	if: ( f → )	If top of stack true (non-zero), execute. [Note: Forth 78 uses IF THEN.]
IF (true) ELSE (false) ENDIF	if: (f → )	Same, but if false, execute ELSE clause. [Note: Forth 78 uses IFELSETHEN.]
BEGIN UNTIL	until: ( f → )	Loop back to BEGIN until true at UNTIL. [Note: Forth 78 uses BEGIN END.]
BEGIN WHILE	while: (f → )	Loop while true at WHILE; REPEAT loops unconditionally to BEGIN.

TED: BOB Here is the documentation on Fig-Forth 1.4 (Calfee). The address for "Going Forth" is: Creative Solutions 14625 Tynewick Terrace Silver Springs MD. 20906 This stuff doesn't seem to cover the editor, so I'm sending a Xerox of the paper stuff. The editor is pretty simple, just whatever you do don't hit RETURN. Ed Rotberg has an improved (less lethal) version. You might try him at 745-1090 (Videa) Sorry, My brain malfunctions when overheated by over work good luck, call again Monday if problems arise.

Cn> PTRIG 1-1	2.0		PR A PR PR C PR		
Oppressed, 1=not pressed.  Cn) STICK 1-1 n=0-3 Reads joystick <n>. Lower 4 bits represent 4 directions. (O=DN 1=DFF) D3-right, D2-left, D1-down, D0-up aright, D2-left, D1-down, D0-up compressed, 1=not pressed.  Cn&gt; STRIG 1-1 n=0-3 Reads joystick <n> trigger button. O=pressed, 1=not pressed  Cn&gt; GRAPHICS 1-0 Open screen in graphics mode <n> (BASIC mode #, NOT ANTIC). Add 16 (dec.) to eliminate split screen. Add 32 (dec.) to prevent clearing screen  Cn&gt; COLOR 1-0 Set "color" (pixel data) for subsequent PLOT or DRAWTO.  Cx&gt; Cy&gt; PLOT 2-0 Plot a point at (<x>,<y>) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  Cx&gt; Cy&gt; DRAWTO 2-0 Draw a line from current position to (<x>,<y>). Open current position to (<x>,<y>). Does not plot, used with PUT, GET.  Cn&gt; PUT 1-0 Put the value of pixel data at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  Cx&gt; Cy&gt; LOCATE 2-1 Same as <x> Cy&gt; POS. GET  Cn&gt; Cc&gt; Cl&gt; SE. 3-0 Set color register <n> to color <c>, "distortion" <d>, and volume <x>, "distortion" <dd>, and volume <x>, "distortion" <dd>, and volume <x>, "</x></dd></x></dd></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></x></d></c></n></x></x></x></y></x></y></x></y></x></n></n></n>	<n></n>		PADDLE	1-1	n=0-7 Reads paddle <n>. Value is 0-FF</n>
Ompressed, lenot pressed.  Cn> STICK 1-1	<n></n>		PTRIG	1-1	n=0-7 Reads paddle triccon button (-)
represent 4 directions. (O=DN 1=OFF) D3-right. D2-left. D1-down. D0-up   Cn) STRIG 1-1 n=0-3 Reads joystick <n> trigger button. O=pressed, 1=not pressed  Cn&gt; GRAPHICS 1-0 GR. Dpen screen in graphics mode <n> (BASIC mode #, NOT ANTIC). Add 16 (dec.) to eliminate split screen. Add 32 (dec.) to prevent clearing screen  Cn&gt; COLOR 1-0 Set "color" (pixel data) for subsequent PLOT or DRAWTO.  Cx&gt; <y> PLOT 2-0 Plot a point at (<x>,<y>) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  Cx&gt; <y> DRAWTO 2-0 Draw a line from current position to (<x>,<y>).  Cx&gt; <y> POSITION 2-0 Set current position to (<x>,<y>). Does not plot, used with PUT, GET.  Cn&gt; PUT 1-0 Put the value <n> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  Cx&gt; <y> LOCATE 2-1 Same as <x> <y> POS. GET  Cn&gt; <c> &lt;1&gt; Set color register <n> to color <c>, luminance &lt;1&gt;.  Cn&gt; Color Colo</c></n></c></y></x></y></x></x></n></y></x></y></y></x></y></y></x></y></n></n>					O=pressed, 1=not pressed.
represent 4 directions. (O=DN 1=OFF) D3-right. D2-left. D1-down. D0-up   Cn) STRIG 1-1 n=0-3 Reads joystick <n> trigger button. O=pressed, 1=not pressed  Cn&gt; GRAPHICS 1-0 GR. Dpen screen in graphics mode <n> (BASIC mode #, NOT ANTIC). Add 16 (dec.) to eliminate split screen. Add 32 (dec.) to prevent clearing screen  Cn&gt; COLOR 1-0 Set "color" (pixel data) for subsequent PLOT or DRAWTO.  Cx&gt; <y> PLOT 2-0 Plot a point at (<x>,<y>) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  Cx&gt; <y> DRAWTO 2-0 Draw a line from current position to (<x>,<y>).  Cx&gt; <y> POSITION 2-0 Set current position to (<x>,<y>). Does not plot, used with PUT, GET.  Cn&gt; PUT 1-0 Put the value <n> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  Cx&gt; <y> LOCATE 2-1 Same as <x> <y> POS. GET  Cn&gt; <c> &lt;1&gt; Set color register <n> to color <c>, luminance &lt;1&gt;.  Cn&gt; Color Colo</c></n></c></y></x></y></x></x></n></y></x></y></y></x></y></y></x></y></n></n>	(n)		STICK	1-1	n=0-2 Panda sasabiat can t
STRIG 1-1  n=O-3 Reads Joystick (n) trigger button. O=pressed, 1=not pressed  Cn) GRAPHICS 1-0 GR.  Open screen in graphics mode (n) (BASIC mode #, NOT ANTIC). Add 16 (dec.) to eliminate split screen. Add 32 (dec.) to prevent clearing screen  Cn) COLOR 1-0 C.  CDLOR 1-0 C.  Set "color" (pixel data) for subsequent PLOT or DRAWTO.  C.  PLOT or DRAWTO.  CN) Upper left corner. See BASIC reference manual for limits in particula modes.  Cx) Cy) PROMTO 2-0 DRAWTO 2-0 DRAWTO 2-0 DRAWTO 1-0 DR.  Cx) Cy) POSITION 2-0 POS.  Cn) PUT 1-0  Put the value (n) at current position. Current (x) is then incremented.  GET 0-1  Get the value of pixel data at current position. Current (x) is then incremented.  Cx) Cy) LOCATE 2-1  Same as Cx) Cy) POS. GET  Cn) Cc) Cl) SE. 3-0 Set color register (n) to color (c), "distortion" (d), and volume (x).  Consult BASIC reference manual for flAUD, CIAUD, etc. will be easier.  CC  PLAYER/MISSILE Graphics  Cx) Cn) HPOS! 2-0  n=O-7 Set PLAYER/MISSILE (n) horizontal			w r a urt	* *	n-o-s reads Joystick (n). Lower 4 bits
O=pressed, 1=not pressed  O=pressed  O=pressed, 1=not pressed  O=pressed  O=press					D3-right, D2-left, D1-down, D0-up
GR. mode #, NOT ANTIC). Add 16 (dec.) to eliminate split screen. Add 32 (dec.) to prevent clearing screen  Cn> COLOR 1-0 Set "color" (pixel data) for subsequent PLOT or DRAWTO.  Cx> Cy> PLOT 2-0 Plot a point at (x>, xy>) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  Cx> Cy> DRAWTO 2-0 Draw a line from current position to (x>, xy>). Does not plot, used with PUT, GET.  Cn> PUT 1-0 Put the value (n) at current position. Current (x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current (x> is then incremented.  Cx> Cy> LOCATE 2-1 Same as <x> Cy&gt; POS. GET  Cn&gt; Co&gt; (1) SE. 3-0 Set color register <n> to color <c>, "distortion" (d), and volume <v>. Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  Cx&gt; Cn&gt; PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile with different modes.  Cx&gt; Cn&gt; HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></v></c></n></x>	<n></n>		STRIG	1-1	n=O-3 Reads joystick <n> trigger button. O=pressed, 1=not pressed</n>
GR. mode #, NOT ANTIC). Add 16 (dec.) to eliminate split screen. Add 32 (dec.) to prevent clearing screen  Cn> COLOR 1-0	<n></n>		GRAPHIC	S 1-0	Onen censon in anation of the contract
eliminate split screen. Add 32 (dec.) to prevent clearing screen  CDLOR 1-0 Set "color" (pixel data) for subsequent C. PLOT or DRAWTO.  CX> Cy> PLOT 2-0 Plot a point at ( <x>,<y>) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  CX&gt; Cy&gt; DRAWTO 2-0 Draw a line from current position to (<x>,<y>).  CX&gt; Cy&gt; POSITION 2-0 Set current position to (CX&gt;,<y>). Does not plot, used with PUT, GET.  CN&gt; PUT 1-0 Put the value CN&gt; at current position. Current (x&gt; is then incremented.  GET 0-1 Get the value of pixel data at current position. Current (x&gt; is then incremented.  CX&gt; Cy&gt; LOCATE 2-1 Same as CX&gt; Cy&gt; POS. GET  CN&gt; CO&gt; SETCOLOR luminance CI&gt;.  CN&gt; CO&gt; CI&gt; SE. 3-0 Set color register (N&gt; to color Cc&gt;, luminance CI&gt;.  CN&gt; CO&gt; CO&gt; CI&gt; SE. 3-0 Set sound channel (N&gt; to pitch (p&gt;, "distortion" CI&gt; In general, storing to FIAUD, CIAUD, etc. will be easier.  CC&gt; PLAYER/MISSILE Graphics  CN&gt; PLAYER 1-0 Set up player-missile graphics with CN&gt; (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  CX&gt; CN&gt; HPOS! 2-0 n=0-7 Set PLAYER/MISSILE (N&gt; horizontal</y></y></x></y></x>				U 1 U	open screen in graphics mode (n) (BASIC
COLOR 1-0 Set "color" (pixel data) for subsequent C. PLOT or DRAWTO.  C. PLOT or DRAWTO.  Cx> Cy> PLOT 2-0 Plot a point at ( <x>,<y>) on screen.</y></x>			wrst.		mode #/ NO! ANIIC/. Add 16 (dec.) to
CDLOR 1-0 Set "color" (pixel data) for subsequent C.  PLOT or DRAWTO.  (x) <y> PLOT 2-0 Plot a point at (<x>, <y>) on screen.</y></x></y>					eliminate split screen. Add 32 (dec.)
C. PLOT or DRAMTO.  (x) (y) PLOT 2-0 Plot a point at ((x),(y)) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  (x) (y) DRAMTO 2-0 Draw a line from current position to (x),(y).  (x) (y) POSITION 2-0 Set current position to ((x),(y)). Does not plot, used with PUT, GET.  (n) PUT 1-0 Put the value (n) at current position. Current (x) is then incremented.  GET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  (x) (y) LOCATE 2-1 Same as (x) (y) POS. GET  (n) (c) (1) SE. 3-0 Set color register (n) to color (c), luminance (1).  (n) (p) (d) (v) SOUND 4-0 Set sound channel (n) to pitch (p), "distortion" (d), and volume (v). Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  c) PLAYER/MISSILE Graphics  (n) PLAYER 1-0 Set up player-missile graphics with (n) (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) (n) HPOS! 2-0 n=0-7 Set PLAYER/MISSILE (n) horizontal					to prevent clearing screen
C. PLOT or DRAMTO.  (x) (y) PLOT 2-0 Plot a point at ((x),(y)) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  (x) (y) DRAMTO 2-0 Draw a line from current position to (x),(y).  (x) (y) POSITION 2-0 Set current position to ((x),(y)). Does not plot, used with PUT, GET.  (n) PUT 1-0 Put the value (n) at current position. Current (x) is then incremented.  GET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  (x) (y) LOCATE 2-1 Same as (x) (y) POS. GET  (n) (c) (1) SE. 3-0 Set color register (n) to color (c), luminance (1).  (n) (p) (d) (v) SOUND 4-0 Set sound channel (n) to pitch (p), "distortion" (d), and volume (v). Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  c) PLAYER/MISSILE Graphics  (n) PLAYER 1-0 Set up player-missile graphics with (n) (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) (n) HPOS! 2-0 n=0-7 Set PLAYER/MISSILE (n) horizontal	(2)		COL OP	1-0	Cat Bashan ( )
Cx> Cy> PLOT 2-O Plot a point at ( <x>, Cy&gt;) on screen. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  Cx&gt; Cy&gt; DRAWTO 2-O Draw a line from current position to (<x>, Cy&gt;).  Cx&gt; Cy&gt; POSITION 2-O Set current position to (Cx&gt;, Cy&gt;). Does not plot, used with PUT, GET.  Cn&gt; PUT 1-O Put the value Cn&gt; at current position. Current Cx&gt; is then incremented.  GET 0-1 Get the value of pixel data at current position. Current Cx&gt; is then incremented.  Cx&gt; Cy&gt; LOCATE 2-1 Same as Cx&gt; Cy&gt; POS. GET  Cn&gt; C1&gt; SE. 3-O Set color register Cn&gt; to color Cc&gt;, luminance C1&gt;.  Cn&gt; Cp&gt; Cd&gt; C1&gt; SE. 3-O Set sound channel Cn&gt; to pitch Cp&gt;, "distortion" Cd&gt;, and volume Cy&gt;. Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  Cn&gt; CPLAYER/MISSILE Graphics  Cn&gt; PLAYER 1-O Set up player-missile graphics with Cn&gt; (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  Cx&gt; Cn&gt; HPOS! 2-O n=0-7 Set PLAYER/MISSILE Cn&gt; horizontal</x></x>	2011			1-0	Det "Color" (pixel data) for subsequent
PL. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  (x) <y> DRAWTO 2-O Draw a line from current position to (<x>,<y>). OR DRAWTO 2-O DRAWTO 2-O DRAWTO (<x>,<y>). OR DRAWTO 2-O DRAWTO 2-O DRAWTO (<x>,<y>). Does not plot, used with PUT, GET.  (n) PUT 1-O Put the value <n> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  (x) <y> LOCATE 2-1 Same as <x> <y> POS. GET  (n) <c> &lt;1&gt; SETCOLOR Set color register <n> to color <c>, luminance &lt;1&gt;.  (n)  <d> <n> Get sound channel <n> to pitch , "distortion" <d>, and volume <y>.</y></d></n></n></d></c></n></c></y></x></y></x></x></n></y></x></y></x></y></x></y>			<b>.</b>		PLUT OF DRAWTO.
PL. (0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  (x) <y> DRAWTO 2-O Draw a line from current position to (<x>,<y>). OR DRAWTO 2-O DRAWTO 2-O DRAWTO (<x>,<y>). OR DRAWTO 2-O DRAWTO 2-O DRAWTO (<x>,<y>). Does not plot, used with PUT, GET.  (n) PUT 1-O Put the value <n> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  (x) <y> LOCATE 2-1 Same as <x> <y> POS. GET  (n) <c> &lt;1&gt; SETCOLOR SETCOLOR SETCOLOR SOUND 4-O Set sound channel <n> to pitch , "distortion" <d>, and volume <y>. Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  CPLAYER/MISSILE Graphics  (n) PLAYER 1-O Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) <n> HPOS! 2-O n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></n></y></d></n></c></y></x></y></x></x></n></y></x></y></x></y></x></y>	200	200	DIOT	~ ~	
(0,0) is upper left corner. See BASIC reference manual for limits in particula modes.  (x) <y> DRAWTO 2-0 Draw a line from current position to (<x>, <y>).  (x) <y> POSITION 2-0 Set current position to (<x>, <y>). Does not plot, used with PUT, GET.  (n) PUT 1-0 Put the value <n> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  (x) <y> LOCATE 2-1 Same as <x> <y> POS. GET  (n) <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c>, luminance &lt;1&gt;.  (n)  <d> <x> <y> SOUND 4-0 Set sound channel <n> to pitch , "distortion" <d>, and volume <v>.</v></d></n></y></x></d></c></n></c></y></x></y></x></x></n></y></x></y></y></x></y>	~X.2	-47		2-0	Plot a point at ( <x>,<y>) on screen.</y></x>
reference manual for limits in particula modes.  (xx) <y> DRAWTO 2-O Draw a line from current position to DR. (<xx,<yx). (<xx,<yx).="" (n)="" 1-o="" <n="" a="" current="" does="" dr.="" draw="" from="" get.="" line="" not="" plot,="" position="" put="" put,="" the="" to="" used="" value="" with=""> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  (xx) <y> LOCATE 2-1 Same as <x> <y> POS. GET  (n) <c> &lt;1&gt; SE. 3-O Set color register <n> to color <c>, luminance &lt;1&gt;.  (n)  &lt; (d) <v> SOUND 4-O Set sound channel <n> to pitch , "distortion" <d>, and volume <y>. Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  (n) PLAYER 1-O Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) <n> HPOS! 2-O n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></n></y></d></n></v></c></n></c></y></x></y></x></x></xx,<yx).></y>			PL.		(O, O) is upper left corner. See BASIC
woodes.  (x) <y> DRAWTO 2-0  Draw a line from current position to  (x), <y>).  (x) <y> POSITION 2-0  POS.  (x)</y></y></y>					reference manual for limits in particula
DR. ( <x>,<y>).  (x&gt; <y> POSITION 2-0 POS. Set current position to (<x>,<yy>). Does not plot, used with PUT, GET.  (n) PUT 1-0 Put the value <n> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  (x&gt; <y> LOCATE 2-1 Same as <x> <y> POS. GET  (n) <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c>, luminance &lt;1&gt;.  (n)  <d> <y> GOUND 4-0 Set sound channel <n> to pitch , "distortion" <d>, and volume <y>.  Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  .c PLAYER/MISSILE Graphics  (n) PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x&gt; <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></n></y></d></n></y></d></c></n></c></y></x></y></x></x></n></yy></x></y></y></x>					modes.
DR. ( <x>,<y>). (<x>,<y>). (<x>,<y>). (<x>,<y>). (<x>,<y>). (<x>,<y>). (<x>,<y>). (<x>,<y>). (<x>,<y>). Does not plot, used with PUT, GET. (<x>,<y>). Does not plot, used with PUT, GET. (<x>,<y>). Does not plot, used with PUT, GET. (<x>,<y>). Exercise the not plot, used with PUT, GET. (<x>,</x> Current <x> is then incremented. ( Get the value of pixel data at current position. Current <x> is then incremented. (<x> <y> LOCATE 2-1 ( Same as <x> <y> POS. GET <x> <y> LOCATE 2-1 ( Same as <x> <y> POS. GET <x> <y> <y> Color (<x> <y> <y> <y> SETCOLOR (<x> <y> <y> SETCOLOR (<x> <y> <y> <y> SOUND (<x> <y> <y> <y> <y> <y> <y> <y> <y> <y> <y< td=""><td>Jan. 19</td><td>,,</td><td>V1.011 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td><td></td><td></td></y<></y></y></y></y></y></y></y></y></y></x></y></y></y></x></y></y></x></y></y></y></x></y></y></x></y></x></y></x></y></x></y></x></x></x></y></x></y></x></y></x></y></x></y></x></y></x></y></x></y></x></y></x></y></x></y></x></y></x>	Jan. 19	,,	V1.011 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		
Set current position to ( <x>,<y>). Does not plot, used with PUT, GET.  Cn&gt; PUT 1-0 Put the value <n> at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  Cx&gt; <y> LOCATE 2-1 Same as <x> <y> POS. GET  Cn&gt; <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c>, luminance &lt;1&gt;.  Cn&gt;  <d><x> <y> SOUND 4-0 Set sound channel <n> to pitch , "distortion" <d>, and volume <x>.  Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  Cn&gt; CPLAYER/MISSILE Graphics  Cn&gt; PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  Cx&gt; <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></n></x></d></n></y></x></d></c></n></c></y></x></y></x></x></n></y></x>	< X >	<y></y>		5-0	Draw a line from current position to
POS.  Ont plot, used with PUT, GET.  One put the value of at current position. Current (x) is then incremented.  GET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET 0-1 Get the value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-2 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data at current position.  OET 0-1 Get value of pixel data a			DR.		( <x>, <y>).</y></x>
POS.  Ont plot, used with PUT, GET.  One put the value of at current position. Current (x) is then incremented.  GET O-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET O-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET O-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET O-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET O-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET O-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET O-1 Get the value of pixel data at current position.  OET O-1 Get the value of pixel data at current position. Current (x) is then incremented.  OET O-1 Get the value of pixel data at current position.  OET O-1 Get value of pixel data at current position.  OET O-1 Get value of pixel data at current position.  OET O-2 OET O-3 OET O-4 GET O-4			4		
PUS. Not plot, used with PUT, GET.  (n) PUT 1-0 Put the value (n) at current position. Current (x) is then incremented.  GET 0-1 Get the value of pixel data at current position. Current (x) is then incremented.  (x) (y) LOCATE 2-1 Same as (x) (y) POS. GET  (n) (c) (l) SE. 3-0 Set color register (n) to color (c), luminance (l).  (n) (p) (d) (v) SOUND 4-0 Set sound channel (n) to pitch (p), "distortion" (d), and volume (v). Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  (n) PLAYER 1-0 Set up player-missile graphics with (n) (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) (n) HPOS! 2-0 n=0-7 Set PLAYER/MISSILE (n) horizontal	<x></x>	<y></y>		A 5-0	Set current position to ( <x>, <u>). Does</u></x>
Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  (x) <y> LOCATE 2-1 Same as <x> <y> POS. GET <n> <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c> to set color register <n> to color <c> to set color register <n> to pitch  sound channel <n> to pitch  do  SO. "distortion" <d>, and volume   consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easierc PLAYER/MISSILE Graphics <n> PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution. PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes. (x) <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></n></n></d></n></n></c></n></c></n></c></n></y></x></y></x></x></x>					not plot, used with PUT, GFT
Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  (x) <y> LOCATE 2-1 Same as <x> <y> POS. GET <n) <c=""> &lt;1&gt; <setcolor< p=""> SETCOLOR SOUND 4-0 Set sound channel <n> to pitch  do sound the sound channel <n> to pitch  do sound the sound the sound startion of the sound the sound for further info. In general, storing to F1AUD, C1AUD, etc. will be easierc PLAYER/MISSILE Graphics <n> PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution. PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes. (x) <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></n></n></n></n></setcolor<></n)></y></x></y></x></x>					· · · · · · · · · · · · · · · · · · ·
Current <x> is then incremented.  GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented.  <x> <y> LOCATE 2-1 Same as <x> <y> POS. GET  <n> <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c> <c> <li> SETCOLOR luminance &lt;1&gt; </li> <li> SOUND 4-0 Set sound channel <n> to pitch</n></li></c></c></n></c></n></y></x></y></x></x></x>	<n></n>		PUT	1-0	Put the value (n) at current nosition
GET 0-1 Get the value of pixel data at current position. Current <x> is then incremented    (x) <y> LOCATE 2-1 Same as <x> <y> POS. GET   (n) <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c>, luminance &lt;1&gt;.   (n)  <d> <v> SOUND 4-0 Set sound channel <n> to pitch , "distortion" <d>, and volume <v>. Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.   .c PLAYER/MISSILE Graphics   <n> PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.   PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.   <x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></x></n></n></v></d></n></v></d></c></n></c></y></x></y></x>					Current <x> is then incremented</x>
position. Current <x> is then incremented  <x> <y> <y> LOCATE 2-1</y></y></x></x>					
<pre>cx&gt; <y> LOCATE 2-1</y></pre>			GET	0-1	Get the value of pixel data at current
<pre> <x> <y> LOCATE 2-1 Same as <x> <y> POS. GET  <n> <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c>,</c></n></c></n></y></x></y></x></pre>					position. Current <x> is then incremented</x>
<pre><n> <c> &lt;1&gt; SE. 3-0 Set color register <n> to color <c>, luminance &lt;1&gt;.</c></n></c></n></pre> <pre><n>  <q> <q> <q> <q> <q> <q> <q> <q> <q> <q< td=""><td></td><td></td><td></td><td></td><td></td></q<></q></q></q></q></q></q></q></q></q></n></pre>					
<pre><n> <c> <li><li><li><li><li><li><li><li><li><li></li></li></li></li></li></li></li></li></li></li></c></n></pre>	<x></x>	<y></y>	LOCATE	2-1	Same as <x> <y> POS. GET</y></x>
SETCOLOR luminance <1>. <pre> <pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre>					
SETCOLOR luminance <1>. <n> <d><d><d><d><v> SOUND 4-0 Set sound channel <n> to pitch Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easierc PLAYER/MISSILE Graphics <n> &lt;</n></n></v></d></d></d></d></n>	(n)	(C)			Set color register <n> to color <c>,</c></n>
SOUND 4-0 Set sound channel (n) to pitch (p), "distortion" (d), and volume (v). Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  C PLAYER/MISSILE Graphics  (n) PLAYER 1-0 Set up player-missile graphics with (n) (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) (n) HPOS! 2-0 n=0-7 Set PLAYER/MISSILE (n) horizontal			SETCOLOR	?	luminance <1>.
SOUND 4-0 Set sound channel (n) to pitch (p), "distortion" (d), and volume (v). Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  C PLAYER/MISSILE Graphics  (n) PLAYER 1-0 Set up player-missile graphics with (n) (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) (n) HPOS! 2-0 n=0-7 Set PLAYER/MISSILE (n) horizontal	2-5		~ 1 %		
SO. "distortion" <d>, and volume <v>. Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  .c PLAYER/MISSILE Graphics  <n> PLAYER 1-O Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  <x> <n> HPOS! 2-O n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></x></n></n></v></d>	Cn>	 b>			
"distortion" <d>, and volume <v>. Consult BASIC reference manual for further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  .c PLAYER/MISSILE Graphics  <n> PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  <x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></x></n></n></v></d>				4-0	Set sound channel <n> to pitch ,</n>
further info. In general, storing to F1AUD, C1AUD, etc. will be easier.  .c PLAYER/MISSILE Graphics <n> PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  (x) <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></n></n>			SU.		"distortion" <d>&gt;, and volume <v>&gt;.</v></d>
F1AUD, C1AUD, etc. will be easier.  . c PLAYER/MISSILE Graphics <pre></pre>					Consult BASIC reference manual for
F1AUD, C1AUD, etc. will be easier.  . c PLAYER/MISSILE Graphics  Cn> PLAYER 1-0 Set up player-missile graphics with <n></n>					further info. In general, storing to
PLAYER/MISSILE Graphics  One PLAYER 1-0 Set up player-missile graphics with one (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  One of the player					F1AUD, C1AUD, etc. will be easier.
PLAYER/MISSILE Graphics  One PLAYER 1-0 Set up player-missile graphics with one (n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes.  One of the player					
<n> PLAYER 1-0 Set up player-missile graphics with <n> (n=1 or 2) vertical line resolution. PBASE constant Returns base address of player/missile DMA area. This is set by GR., and changes with different modes. <x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></x></n></n>		for her	4 TP 275 200 W 4		
(n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile  DMA area. This is set by GR., and changes with different modes.  (x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n>	FLAY	EK/I	IISSILE G	raphics	
(n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile  DMA area. This is set by GR., and changes with different modes.  (x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n>					
(n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile  DMA area. This is set by GR., and changes with different modes.  (x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n>			Par Asserm		
(n=1 or 2) vertical line resolution.  PBASE constant Returns base address of player/missile  DMA area. This is set by GR., and changes with different modes.  (x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n>	\n>		PLAYER	1-0	Set up player-missile graphics with <n></n>
PBASE constant Returns base address of player/missile  DMA area. This is set by GR., and changes with different modes.  (x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n>					(n=1 or 2) vertical line resolution.
DMA area. This is set by GR., and changes with different modes. <x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></x>					
DMA area. This is set by GR., and changes with different modes. <x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></x>			PBASE CO	nstant	m m m m m m m m m m m m m m m m m m m
with different modes. <x> <n> HPOS! 2-0 n=0-7 Set PLAYER/MISSILE <n> horizontal</n></n></x>					DMA area. This is set by GR., and changes
- CONTRACTOR CONTRACTO					with different modes.
- CONTRACTOR CONTRACTO					
position to <x>. MISSILE O = PLAVER 4 atc</x>					
	Cx>	Cn>	HPOS!	2-0	n=0-7 Set PLAYER/MISSILE <n> horizontal</n>

6	<x> <n> SIZE! 2</n></x>	-0	n=0-4 Set size of PLAYER <n> ( all MISSILEs if n=4 ) to <x>. x=1 -&gt; double size. x=3 -&gt; quad size.</x></n>
	<x> <n> COLPM! 2</n></x>	-0	n=0-3, x=0-254 step 2
$\Phi$ $ ^3$			Set color of PLAYER/MISSILE <n> to <x>. Note</x></n>
5 6			that MISSILE <n> is same color as PLAYER <n> unless "fifth player enable" is selected.  ( see PRIOR in hardware manual )</n></n>
8 9			You should store the desired priority value here. Consult the hardware manual for details.
11	VDELAY b		You should store the "vertical delay" bits here. This is only needed if single line
13 14 15			positioning is needed with double line graphics resolution. Consult the hardware manual.
16 17 18	.c CIO words		
19 20 21	<n> IOCB 1-</n>		n=O-7 Set "current" IOCB to <n>. IOCB O is normally open to E: and should not be changed. IOCB 1 is used by GRAPHICS commands</n>
22 23 24			and should be restored before using them. (ed. note. — I know that this should have been
25 26 27			IOCB 6, and that words shouldn't make such assumptions. I didn't write this stuff, I'm just documenting what exists.)
28 29 30	C10 0-		Calls CIO. Assumes that " <n> IOCB " has been done, and that IOCB n is correctly set up. You should carefully read the O.S. manual.</n>
31 32			The following "constants" will refer to the "current" IOCB:
33 34 35	I	CCOM	Device # in HATABS. (byte) Command (byte)
> 36			Status (byte) Buffer Address (word)
) u 37			Buffer Length (word)
SE 28		1CAX	AUX1 (byte)
\$590	12		AUX2 (byte) (note: these are odd to allow use with fixed-length-name FORTHs)
Moore Business Forms, Inc. 38 39 40 41 42 43	IC	CPTL	"put-byte" address (word) (used by BASIC, shun like plague)
44 45	<n> ACIO 1-</n>		Call CIO with acc=n. Normally used with ICBLL=O
47 48	CIDA O-		Call CIO and put returned acc on stack. Normally used with ICBLL=O
50	<2> <1> <d> OPEN 3</d>		Open current IOCB with AUX1=<1>, AUX2=<2> <d>&gt; pointing to dvc:file.ext. <d> should</d></d>
53 54			be just a pointer (no count) and string should be terminated with non-alphanumeric.
55 56 57 58	CLOSE 0-		Close the current IOCB. One of the quickest ways to hang FORTH is to type CLOSE without previously typing <n> IOCB. This closes</n>
59		1	IOCB O, then returns to the outer interpreter which attempts to read from IOCB O. The resulting error message also tries to get out via IOCB O.
	Cn> PUT 1-		Do a "put character" with char= <n></n>
	-		