Programming Manual for STUDIO III

September 1977

Programming Manual for STUDIO III

September 1977

CONTENTS

INTRODUCTION

P. K. Baltzer

The STUDIO III system is an RCA CDP-1802 microprocessor based video game system. STUDIO III represents an extension of the earlier black and white, single-tone STUDIO II system. These systems have been designed to provide upward and downward software compatibility (STUDIO II games will play on STUDIO III as white images on a blue background, and STUDIO III programs will play on a STUDIO II system in black and white).

The STUDIO II interpreter ROM and, therefore, the STUDIO II language are used in the STUDIO III system - to provide the above-mentioned software compatibility. The two new functions of STUDIO III are programmable image color, programmable background color, and programmable sound. These features are implemented by machine-language subroutines in STUDIO III which can be called by the STUDIO II language, as described in the appropriate sections of this manual.

STUDIO II LANGUAGE

J. A. Weisbecker

DEBUGGING TIPS

1. Always save a cassette copy of a newly entered program in case it destroys itself upon a first run attempt.
2. You can manually insert branches which loop on themselves to stop an undebugged program at any desired point. Examine registers and memory to determine if operation is proper to this point. Move breakpoint branch further along in program and try again.
3. When using the interpretive Studio II in a RAM system remember that your program bugs can also destroy the language interpreter. Reload from cassette if you suspect that this has happened.

STUDIO II LANGUAGE

0MMM - Execute Machine Language Routine at 0MMM, P = 3, D4 to Return
1MMM - Go to 0MMM
2MMM - Do Subroutine at 0MMM (C0 to Return)
C0 - Return from Subroutine
3XMM - Go to MM (Same Page) If VX ≠ 00
4XMM - Go to MM (Same Page) If VX = 00
5XKK - Skip 2 bytes if VX ≠ KK
6XKK - KK → VX
70MM - V0 -1, Go to MM (Same Page) If V0 ≠ 00 (Note V0 = 01 if No Branch)
7XKK - If X ≠ 0; VX + KK
8XY1 - VX/VY → VX, VB
8XY2 - VX·VY → VX, VB
8XY3 - VX ⊕ VY → VX, VB
8XY4 - VX + VY → VX (01 → VB If VX + VY > FF, 00 → VB If VX + VY ≤ FF)
8XY5 - VX - VY → VX (01 → VB If VX ≥ VY, 00 → VB If VX < VY)
9XY0 - Skip 2 Bytes If VX ≠ VY
9XY1 - VX → VY
9XY2 - M(08VY) → VX
9XY4 - VX → M(08VY)
9XY8 - VX → M(08VY , VY + 1, VY + 2) Decimal, VY(FINAL) = VY + 2
AMMM - 0MMM → A (Memory Pointer), for TVA (TV address).
BNKK - KK → M(A), A + N (Same Page)
CXKK - RR·KK → VX, X ≠ 0
DKMM - KEY(K) → VB · Go to MM (Same Page)
"A" Key if VA = 1, "B" Key if VA = 0, Do next instruction if KEY(K) is not pressed. If K = F, Go to MM if KEY(VB) is pressed.
E0 - Erase V9 RAM Pattern
E1 - Shift V9 RAM Pattern f (V9 Direction Byte), NV/NH - 1
Skip if direction ≠ U/D/L/R (2/8/4/6)
E2 - Shift V9 RAM Pattern f (VC), NV/NH-1, Skip if VC ≠ 2/8/4/6
E4 - M(A) H Bytes X0R V9 RAM → V9 RAM, A(FINAL) = A + H, Skip if H = 0
E8MM - Move V9 RAM Pattern → TV(X0R), Go to MM if Hit
FXB6 - VX(LSD) → A(LSD), VX(FINAL) - VX · 0F
FXA6 - M(A) → VX
FXA9 - VX → M(A)
FXB3 - VX → AL0
FXAC - M(A) → VX, A + 1
FXAF - VX → M(A), A + 1
FX4D - Set X (Preset R6 for 0MMM Next)

8XY6 - VY/2→VX, underflow→VB
8XYE - 2*VY→VX, overflow→VB

V9 = Current RAM Pattern # (00-07)
VA = A/B Key Flag (01 = A, 00 = B)
VB = Arithmetic Overflow/Key Input Byte
VC = Direction Byte
VD = Tone Timer (Q On if VD ≠ 00)
VE = Timer
VF = Timer

} Special-Purpose Variables

* Programs begin at 0400. In ROM systems 400-7FF program storage will be read only. Do not use 3 byte COSMAC instructions in machine language routines.

STUDIO II ADDITIONAL STANDARD SUBROUTINES

Keyboard Functions

220B - Key debounce: Tone and debounce delay, Return after A/B Key Off f(VA)
23C3 - Wait for A/B key release f (VA)
23D8 - Read A/B direction key (2/4/6/8) into VB if pressed (tone + delay) waits for release of 2/4/6/8 otherwise sets VB = 00.

Scorekeeping Functions

2388 - A/B score → V3, f(VA). Changes V2, V3, VB.
A score = M(0881), B score = M(0880)
238F - A/B score + V1, f(VA). Changes V2, V3, VB.
2368 - Show A/B score, f(VA). 3 digit score shown.
Changes A(Memory Pointer), V0, V2, V3, V9, VC.
23A3 - Show A(Memory Pointer) and B scores, 2 sec. delay
Changes A(Memory Pointer), V0, V2, V3, V9, VA, VC, VE
23D1 - Erase old A/B score and show new A/B score f(VA)
Changes A(Memory Pointer), V0, V2, V3, V9, VB, VC. New = old + V1
23B0 - Sound warble tone and stop if A/B score = V1
Changes V2, V3, VA, VB

Misc. Functions

22F9 - Short tone + 160 ms delay
0363 - Roll display down one spot position
2336 - Write V9 RAM Pattern at TVA
02F2 - Clear RAM page from M(A) to M(00) of specified page.
229E - Repeat and shift V9 pattern: Write V9 pattern to TVA,
Shift V9 Pattern f(VC), & Repeat V0 times.
233D - Sound warble tone
2344 - Draw border (use only at beginning of program)
2396 - Set A = pattern address, for digit at M(08V2). V2-1. uses V3
2372 - Show digits at M(08V3, V3-1, V3-2) f (V9). Changes A(Memory Pointer),V0,V2,V3,VC
23BF - Sound warble tone and stop
23E6 - Repeat RAM 1 pattern on TV screen. M(A) = list of TV destination address (TVA's) for pattern in RAM 1. V0 = number of entries in M(A) list of TVA's. Preset RAM 1 H, in location 08F1, to pattern height.

COSMAC REGISTER UTILIZATION:

R

0 - DMA pointer for TV Refresh
1 - Interrupt pointer for TV refresh subroutine
2 - Stack pointer
3 - Micro/machine language program counter
4 - Call subroutine program counter
5 - Macro program counter
6 - 08CX at beginning of Macro. Used to address VX.
(Do not change R6.1)
7 - 08CY at beginning of Macro. Used to address VY.
8 - Used in interrupt subroutine.
9 - Random number storage. Incremented by 1 in interrupt subroutine.

A - Memory pointer used by some Macros.
B - Used in interrupt routine to initialize R0.
Increment/Decrement by 08 to roll display.

C,D,E,F, - used within Macro interpretive routines but can be used without conflict by user generated machine language routines.

Timing Data

TV display circuits cause interrupts 60 times per second (16.7ms intervals). Interrupt routine requires 8 - 9 ms.

VD, VE, VF are decremented by 1 at a rate of 60 times per second. No decrementing occurs when the variable = 00.

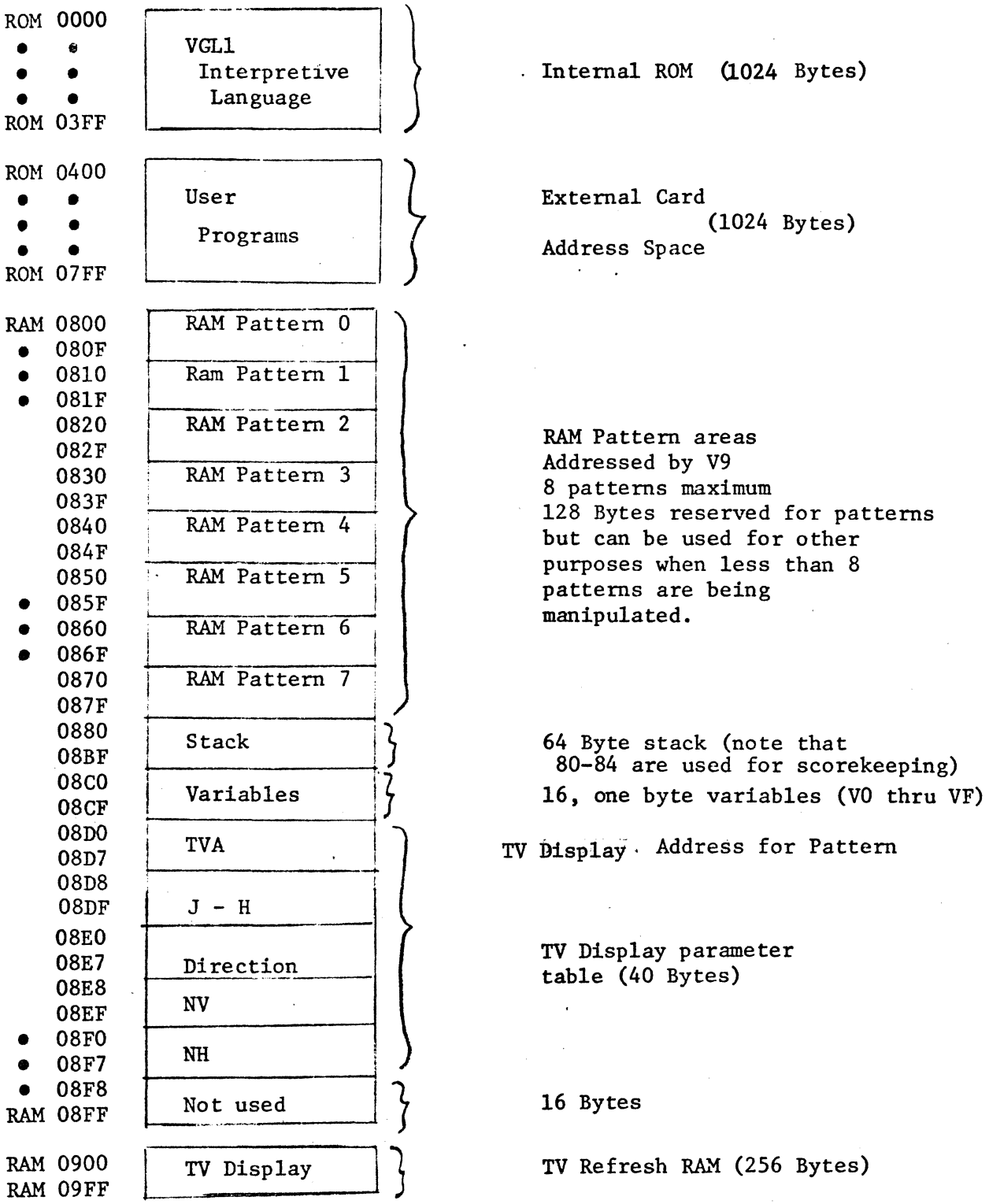
When VD$\geq$02 a tone is sounded via the built-in speaker.

Delays or tones up to 4 seconds long can be generated by setting VD/VE/VF to a value, and waiting for it to be decremented to 00.

The following subroutine generates a 1 second tone.

M	Instruction	Comments
400	6D3C	Set VD = 3C (decimal 60) and turn on tone
402	3D02	Loop if VD$\neq$00

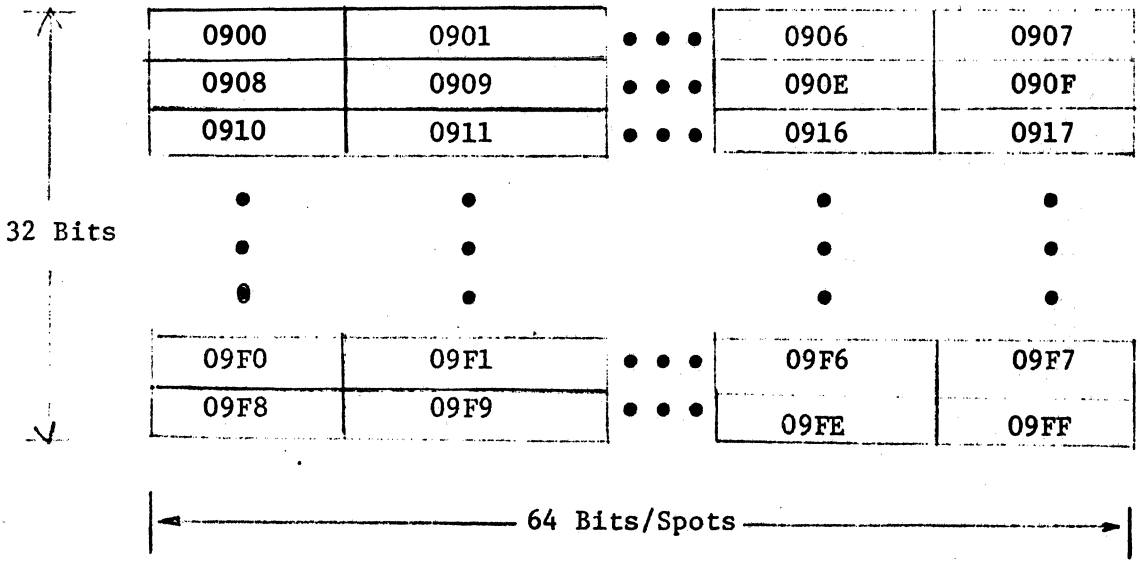
STUDIO II MEMORY MAP



* Note: RAM 0800 thru 089F initially cleared to 00

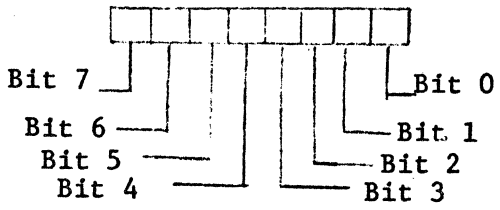
* Note: RAM 0900 thru 09FF initially cleared to 00 (blank screen)

180 VGL1 - TV Display RAM Map (See Fig. 1 for Image Storage Locations)



Byte locations 0900 - 09FF are shown on the TV screen as indicated above:

Each Byte is shown as follows:



* 1 = spot on (white)
  0 = spot off (black)

Bit 7 of the Byte at M(0900) would be at the upper left corner of the TV screen and Bit 0 of Byte M(09FF) at the lower right corner.

Keyboard Layout

Keyboard instructions use the 8 bit binary code for the 10 digit keys. Pattern Manipulation instructions use 2 for up, 4 for left, 6 for right, and 8 for down.

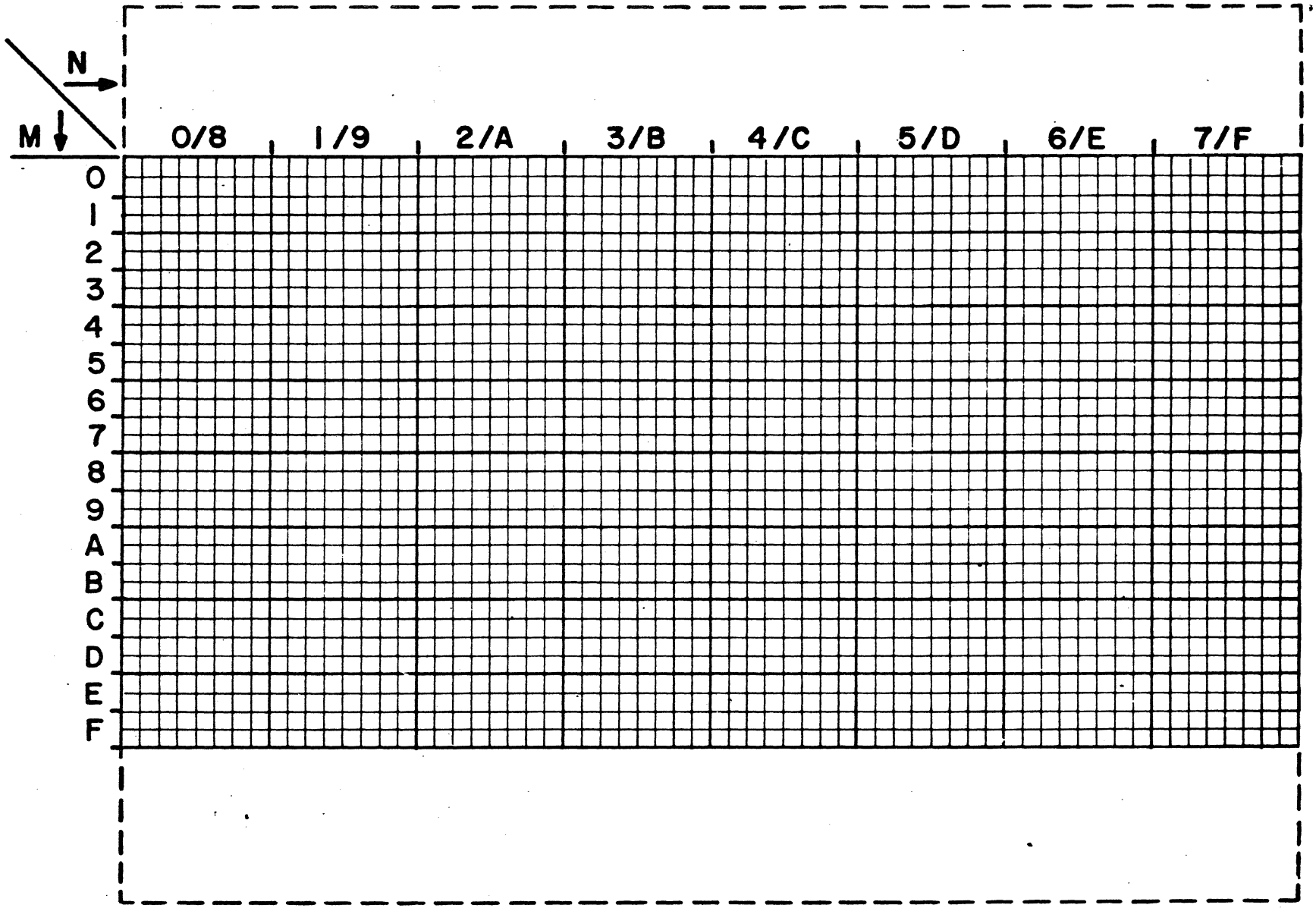
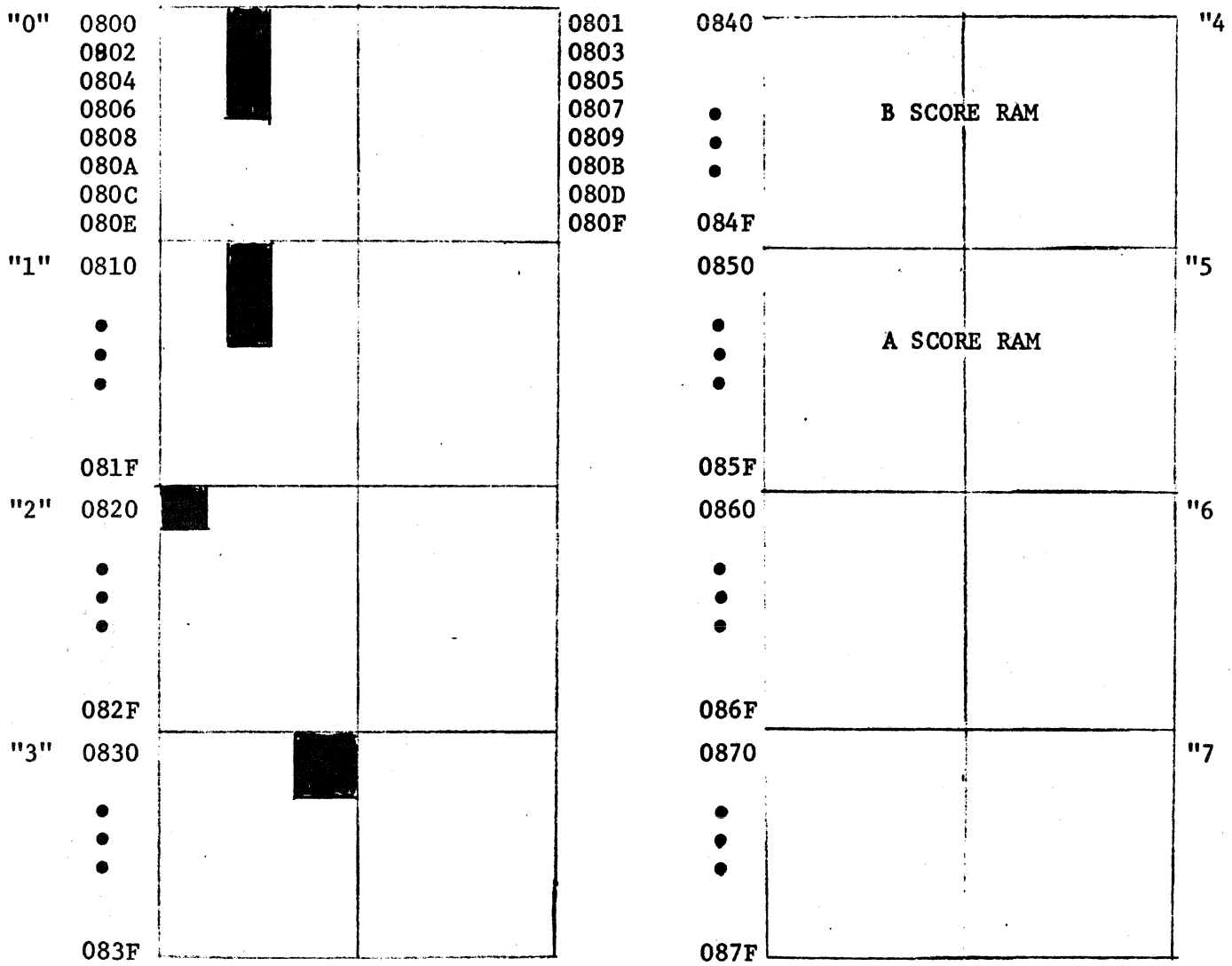


IMAGE STORAGE LOCATIONS, 09MN

FIGURE 1

RAM PATTERN AREAS AND INITIAL CONTENTS

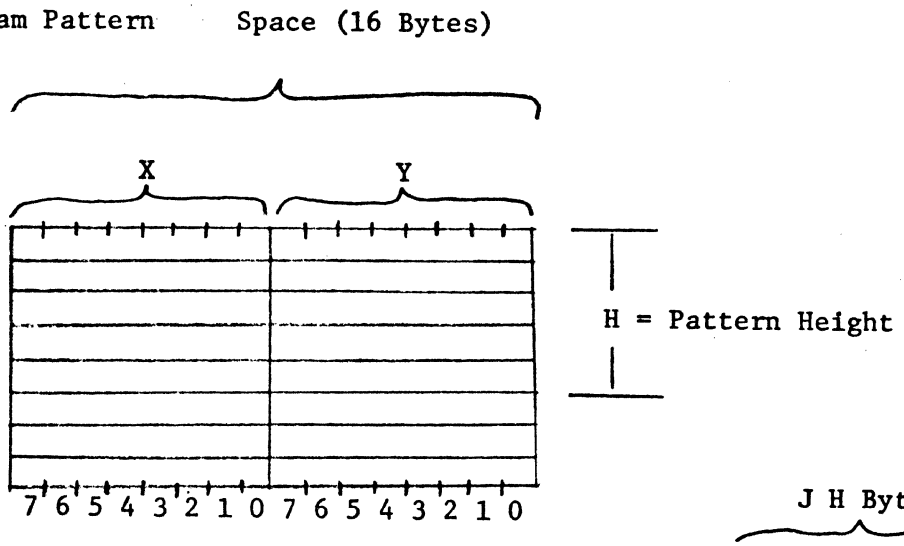


TV DISPLAY PARAMETER TABLE

	TVA	JH	DIR	NV	NH
RAM0 →	08D0- XX	D8 - 04	E0 - XX	E8 - XX	F0 - XX
RAM1 →	D1 XX	D9 - 04	E1 - XX	E9 - XX	F1 - XX
RAM2 →	D2 F8	DA - 01	E2 - XX	EA - XX	F2 - XX
RAM3 →	D3 XX	DB - 02	E3 - XX	EB - XX	F3 - XX
RAM4 →	D4 15	DC - 05	E4 - XX	EC - XX	F4 - XX
RAM5 →	D5 11	DD - 05	E5 - XX	ED - XX	F5 - XX
RAM6 →	D6 XX	DE - XX	E6 - XX	EE - XX	F6 - XX
RAM7 →	D7 - XX	DF - XX	E7 - XX	EF - XX	F7 - XX

Note 1 XX = Unknown initially
Note 2 0880 = B score Byte
0881 = A score Byte
0882,3,4 = Temporary storage for Byte to 3 digit decimal conversion routine.

RAM PATTERN MANIPULATION RULES:



E4 Instruction:
Moves H bytes from M(A)
into X bytes of RAM pattern space
starting at top.

Right Shift Rules: (→X → Y—)
If X bit Ø = 1 and J = Ø, 8 → J and TVA + 1
If Y bit Ø = 1 and J = 8, Ø → J and TVA +1

Left Shift Rules: (—X → Y←)
If Y bit 7 = 1 and J = Ø, 8 → J and TVA - 1
If X bit 7 = 1 and J = 8, Ø → J and TVA - 1

E1/E2 Instructions

Up Shift = TVA - Ø8

Down Shift = TVA + Ø8

E8 Instruction:
If J = Ø, X → TVA and Y → TVA - Ø1
If J = 8, Y → TVA and X → TVA - Ø1

*Note: Maximum pattern size = 8 bits x 8 bits
for normal operation

STUDIO II - PROGRAMMING EXAMPLES

Moving a spot around the screen (19 Bytes)

```
Ø4ØØ    69Ø3     Let V9 = Ø3  (spot pattern RAM)
Ø4Ø2    6AØ1     Let VA = Ø1  (select a keyboard)
Ø4Ø4    A8D3     Let A = Ø8D3  (TVA for RAM 3 pattern)
Ø4Ø6    BØØØ     ØØ → M(A)  (Let TVA3 = ØØ)
Ø4Ø8    E8ØA ┐   Show V9 RAM pattern  <-------------------------------+
Ø4ØA    23D8 <┘  A direction Key/ØØ → VB                               |
Ø4ØC    9BC1     VB → VC                                               |
Ø4ØE    E81Ø ┐   Erase pattern                                         |
Ø41Ø    E2   <┘  Shift pattern f (VC)                                  |
Ø411    14Ø8     Repeat  ------------------------------------------------+
Ø413    ----
```

New 2/4/6/8 Key/00 → VB Subroutine (16 Bytes)

```
Ø5ØØ    D2ØB     Key 2 → VB  →  |
Ø5Ø2    D4ØB     Key 4 → VB  →  |
Ø5Ø4    D6ØB     Key 6 → VB  →  |
Ø5Ø6    D8ØB     Key 8 → VB  →  |
Ø5Ø8    6BØØ     ØØ → VB        |
Ø5ØA    CØ       Return         |
Ø5ØB    6FØ1     Ø1 → VF  <-----+
Ø5ØD    3FØD     Wait for VF = ØØ
Ø5ØF    CØ       Return
```

Change M(Ø4ØA) to 25ØØ for continuous, fast, soundless spot motion when direction key is pressed.

Make following changes to eliminate blinking while spot is stationary.

```
Ø4ØE    4CØA     Go to Ø4ØA if VC = ØØ
Ø41Ø    E812 ┐   Erase pattern
Ø412    E2   <┘  Shift pattern f (VC)
Ø413    1408     Repeat Ø4Ø8
Ø415    ----
```

STUDIO II-PROGRAMMING EXAMPLES

Generate Key selected tone sequence (31 Bytes)

```
Start → Ø4ØØ   6AØ1     Set VA = Ø1  (select a keyboard)
        Ø4Ø2   6BØ9     Set VB = Ø9
        Ø4Ø4   DFØC     Go to ØC if VB key on
        Ø4Ø6   4BØ2     Go to Ø2 if VB = ØØ
        Ø4Ø8   7BFF     VB - Ø1
        Ø4ØA   14Ø4     Repeat
        Ø4ØC   9BD1     VB → VD  (tone on)
        Ø4ØE   3DØE     Wait for VD = ØØ  (tone off)
        Ø41Ø   9BE1     VB → VE  (Set delay)
        Ø412   3E12     Wait for VE = ØØ  (end delay)
        Ø414   Ø5ØØ     Skip next instruction if EF3 = 0
        Ø416   14ØC     Loop
        Ø418   14Ø2     Go to
        Ø41A   ——

Machine→Ø5ØØ   36Ø4     EF3 = 1?  Yes
Lang.   Ø5Ø2   1515     R5 + 1, R5 + 1
Sub-    Ø5Ø4   D4       Return
routine
```

Randomly moving spot (25 Bytes)

```
        Ø4ØØ   69Ø3     Let V9 = Ø3  (spot pattern RAM)
        Ø4Ø2   A5ØØ     Let A = Table
        Ø4Ø4   E8Ø6     Show spot
        Ø4Ø6   C1Ø3     RR·Ø3 → V1
        Ø4Ø8   F1B3     V1 → AL0
        Ø4ØA   FCA6     M(A) → VC
        Ø4ØC   6FØ4     Ø4 → VF
        Ø4ØE   3FØE     Wait for VF = ØØ
        Ø41Ø   E812     Erase spot
        Ø412   E2       Shift spot f (VC)
        Ø413   14Ø4     Repeat

Table → Ø5ØØ   Ø2       Up
        Ø5Ø1   Ø4       Left
        Ø5Ø2   Ø6       Right
        Ø5Ø3   Ø8       Down
```

STUDIO II - PROGRAMMING EXAMPLES

Showing a keyboard digit on TV (30 Bytes)

```
Ø4ØØ   6AØ1    Let VA = Ø1  (sel. A keyboard)
Ø4Ø2   69Ø4    Let V9 = Ø4  (sel. RAM 4 space)
Ø4Ø4   6BØ9    Ø9 → VB
Ø4Ø6   DFØE    Go to ØE if VB key on
Ø4Ø8   4BØ2    Go to Ø2 if VB = ØØ
Ø4ØA   7BFF    VB - Ø1
Ø4ØC   14Ø6    Repeat
Ø4ØE   A21Ø    21Ø → A  (Dec. Dig. to pattern table)
Ø41Ø   FBB6    VB(LSD) → A(LSD)
Ø412   F1A6    M(A) → V1
Ø414   F1B3    V1 → AL0
Ø416   E818    Show RAM f (V9) - Erase TV
Ø418   EØ      Erase RAM pattern f (V9)
Ø419   E4      M(A) pattern → RAM f (V9)
Ø41A   E81C    Show RAM f (V9)
Ø41C   14Ø6    Repeat
Ø41E
```

Showing Random Bytes on TV in 3 decimal digit format (14 Bytes)

```
Ø4ØØ   6AØ1    Ø1 → VA  (select a score)
Ø4Ø2   2368    Show A/B score f (VA)
Ø4Ø4   C1FF    RR· FF → V1
Ø4Ø6   23D1    Erase Old A/B score and show New + V1
Ø4Ø8   6FØF    ØF → VF
Ø4ØA   3FØA    Wait for VF = ØØ
Ø4ØC   14Ø4    Repeat
```

PROGRAMMING COLOR

P. K. Baltzer

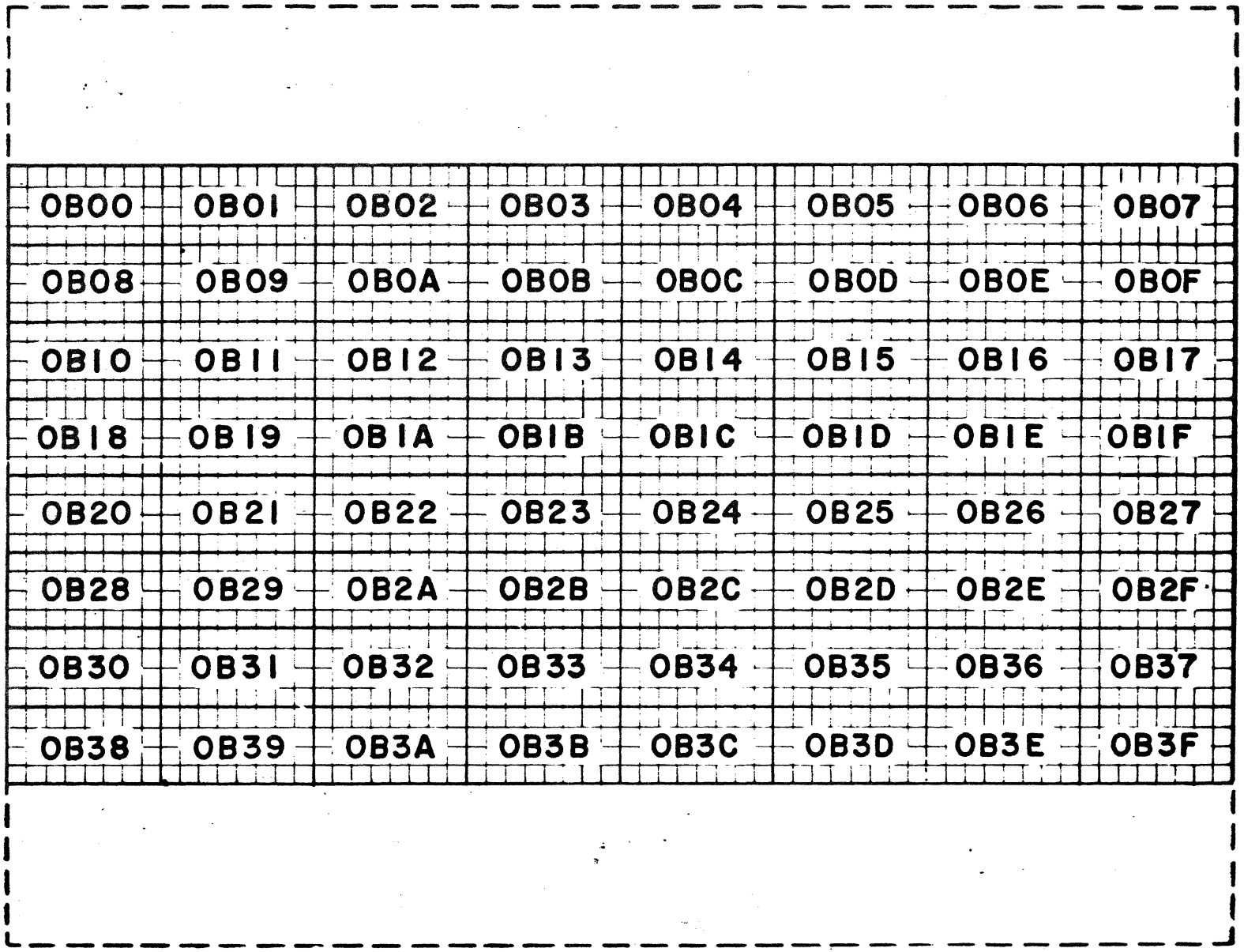
A color-map technique is used in STUDIO III to define the color of the image (or "Luminence") information, written using the STUDIO II language. The above colored image information is shown by the STUDIO III on a uniform programmable color background, which is separately defined using a special COSMAC I/O instruction. Since neither the image color nor the background color are explicitly considered in the STUDIO II language, these functions must be implemented in STUDIO III using machine-language subroutines called by the STUDIO II instruction 0MMM. Each function will be separately described and examples of typical machine code subroutines given.

Image Color Programming

Image color is defined in terms of blocks covering 4x8 areas of image spots. Images or parts of images occupying the same color block will have the same color. There are 64 color blocks in the display field and each block can be any one of eight colors. The information defining these 64 color blocks must be placed in the 64 consecutive locations 0B00 to 0B3F. The geometric correspondence of these memory locations and the actual display is indicated in Fig. 1. Each of the above memory locations must have one of the following byte codes, corresponding to the desired color:

00 - Black
01 - Red
02 - Blue
03 - Violet (blue & red)
04 - Green
05 - Yellow (green & red)
06 - Aqua (green & blue)
07 - White.

When the STUDIO III system is cleared (or reset), the system hardware initializes to an all white luminence, which will remain so until the color map memory 0B00 - 0B3F is addressed by software. Note that the STUDIO II



COLOR MAP STORAGE LOCATIONS

FIGURE 1

interpreter erases all spot memory before first executing the starting op-code at location 0400. Therefore, it is good practice to program the initial state of the color map before any images are written on display.

A typical program* for setting a consecutive number of color map locations (such as the bottom row of color blocks) to a single color is given below:

Memory Location	STUDIO II Op Codes	Main Program
0440	AB38	Set A to first location in color-map.
0442	0500	Call machine language subroutine at location 0500.
0444	0801	Data for machine language subroutine.

No. of blocks — Color (red)

- -

	Machine Language Code	Subroutine
0500	45,AF	Read No. of blocks and put low Reg.F.
0502	45,AE	Read color and put low Reg. E.
0504	8E,5A,1A	Get low E and store M(A); Inc.A
0507	2F,8F,3A04	Dec. F; get low F and branch if ≠ 0.
050B	D4	Return

Background Color Programming

The reset state for the STUDIO III background color is blue. The background color is changed by sequencing through a cyclic sequence of colors:

Blue → Black → Green → Red

This sequence is incremented by the machine code "61" instruction, that does not need any data from memory. Since a 61 instruction increments the X register, it is necessary to set X to some unused register to provide a dummy X register that cannot influence other portions of the program.

*Courtesy of A. A. Modla.

A typical machine language subroutine that could be used is given below:

Memory Location	Machine Code	Explanation
0600	EF,61*	Set X=F, Sequence Background
0602	EF,61*	Set X=F, Sequence Background
0604	EF,61*	Set X=F, Sequence Background
0606	D4	Return

To sequence background blue to red call program at 0600; to sequence background blue to green call program at 0602; and to sequence background blue to black call program at 0604.

---

* Note that this "61" instruction applies to an actual STUDIO III system. Since the VIP normally uses this instruction to turn off the video chip, programs run on an unmodified VIP should use the "65" instruction in the above program.

PROGRAMMING SOUND

J. J. Winsor

This programmable tone generator is designed to produce 255 possible frequencies from a single frequency input clock. It is directly compatible with the CDP 1802 microprocessor. An eight bit data input allows programming the divide rate of the tone generator. A fixed internal predivide scales the input clock down so that the programmable divide rates all provide frequencies in the audio range.

FREQUENCY DIVISION OF STUDIO III TONE GENERATOR

3.579545 MHz	Color Frequency
÷2 = 1.789772 MHz	Clock for Studio III Prototype
÷8 = 223.721 KHz	TPB 1802 mP Signal
÷8 = 27.965 KHz	Maximum Frequency Without Programmable Divide Rate

÷N Variable divide rate which can be altered by program. Divides the maximum frequency by 1 through 255 (01_{16} through FF_{16}). Hex value represents programmed divide rate which is a delay or inversely proportional to the resulting frequency.

If N = 01 Resulting frequency = 27.965 KHz

If N = FF Resulting frequency = 108 Hz

To calculate the proper hex code to provide a given frequency divide 27.965 KHz the maximum frequency by the frequency you want to attain, round off, and convert the resulting number to Hex.

$$\left[(27.965 \text{ KHz} \;/\; \text{Desired Freq.}) \pm .5\right] \longrightarrow \text{HEX}$$

GENERAL INFORMATION

In the Studio II/III System there are no interpretive instructions to handle programming the tone generator. This requires the user to call machine language subroutines to control this operation.

The Studio III Prototype System provides for an initial frequency to be programmed into the tone generator. The purpose of this is to allow the older Studio II cartridges, which have no tone programming operations, to be plugged into a Studio III and have the same tone as in a Studio II.

In addition to programming the frequency of the tone generator the tone itself must be turned on and this can be done by an interpretive instruction. Loading variable D with some value turns the tone on. This value is decreased by 1 during each execution of the display refresh until Variable D equals 0 and then the tone is turned off. In this manner single tones up to 4 seconds long can be created.

PROGRAMMING THE TONE GENERATOR

Programming the frequency is not as easy as turning on the tone. First an interpretive instruction which calls a machine language routine must be used. The machine routine must execute a 64 instruction to program the tone generator with the desired value. This is an output instruction and so must refer to some memory location. This location can be pointed to by any register and that register must be pointed to by the X register.

The values used to program the tone generator may be stored in a table of data. This will require that some register be set up each time the routine is executed, before the X register is set and the 64 instruction used. This is useful for strings of tones.

Alternately the values used to program the tone generator may be combined with the interpretive or machine language code. This only requires setting the X register equal to the program counter of the code with which the data is combined.

Example 1 = Table of Tones

a) the values are stored in a table in memory
b) the first location of the table is set into a RAM memory location by program
c) the data at the RAM location is set in some machine register
d) the X register is set to the number of the machine register
e) a 64 instruction loads the data into the tone generator and increases the value of the pointer register by 1
f) the value of the register is saved in the RAM location
g) steps c through f are repeated for each new frequency to be programmed into the tone generator

Example 2 = Tone Data Combined with Instructions

a) set X register to point to the interpretive code counter register
b) execute a 64 instruction
c) return to interpretive code

Code for Example 2:

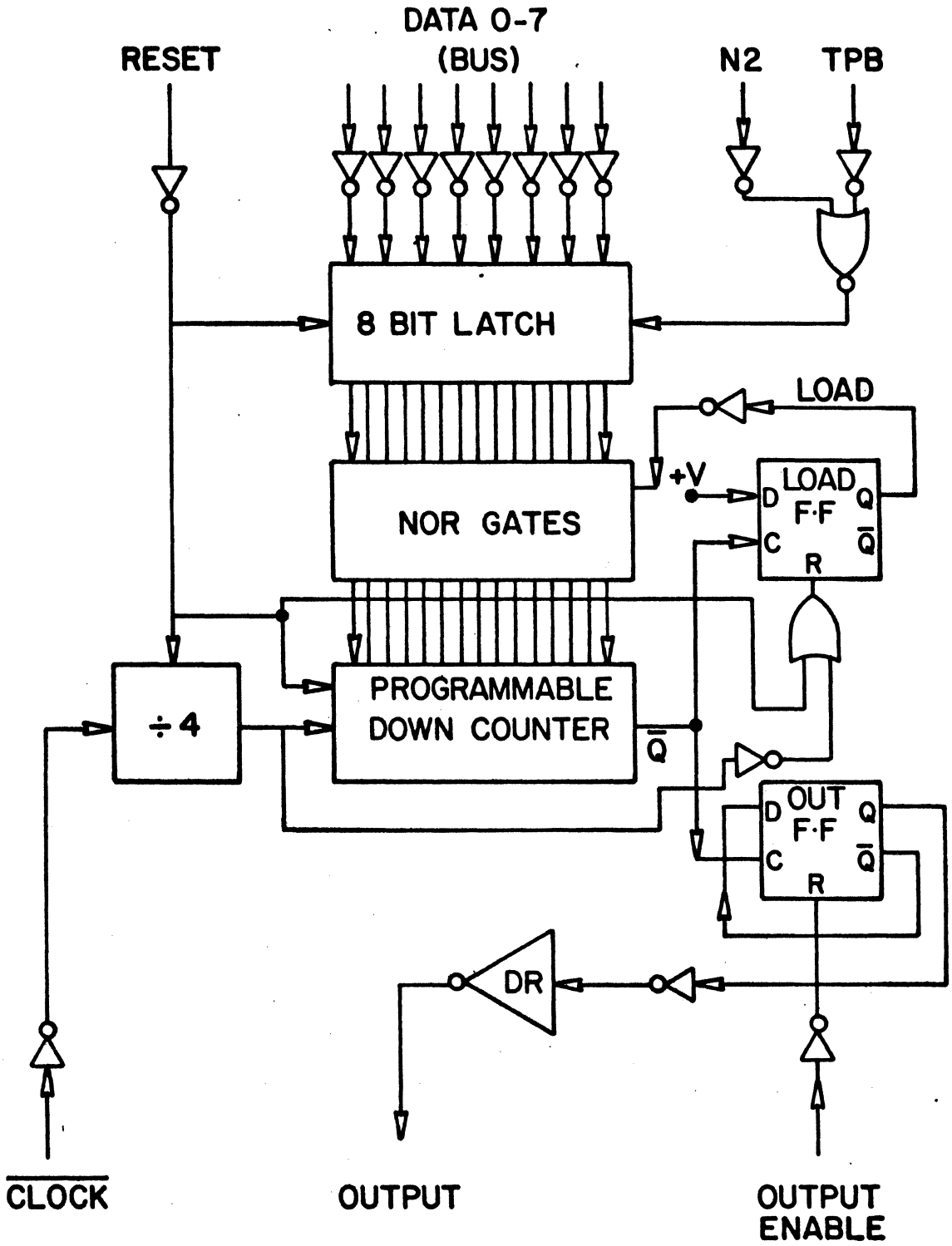
Interpretive Code		Machine Code	
Mem. Loc.	Instruction	Mem. Loc.	Instruction
:	:	0600	E5 X=5
0480	0600 Machine Call	0601	64 Output RX+1
0482	B5 Data	0602	D4 P=4
0483	:		
:	:		

The Interpreter Program Counter for Macro Code is R5. Interpretive instruction 0600 branches to the machine routine at address 0600. The X pointer points to R5. The 64 instruction outputs the data, B5, to the tone generator and increments R5 so that returning to interpretive code picks up at location 0483.

HOW IT WORKS

As the 64 instruction is implemented, the data at the memory location, pointed to by the register indicated by X, is put onto the bus. The 64 instruction also causes output line N2 of the 1802 to go high. This is ANDed with TPB of the 1802 to create a strobe at the time the data is valid on the bus.

This strobe is used to latch the data into the 8 bit latch of the tone generator as shown in Figure 1. The clock feeding the tone generator is scaled down and further divided by the programmable counter. When the counter gets down to zero, the Output Flip Flop toggles and the output also toggles provided the output enable is high. At the same time the Load Flip Flop causes the contents of the latch to preset the programmable counter to the desired divide rate. It can be seen that the latch is operated independently of the counter circuit.



BLOCK DIAGRAM - PROGRAMMABLE TONE GENERATOR

FIGURE 1

STUDIO II INTERPRETER

J. A. Weisbecker

STUDIO II INTERPRETER

LABEL	LOCATION	CODE	DESCRIPTION
START	0000	90 B1	00→R1.1
	02	B4 A5 AB	00→R4.1,R5.0,RB.0
	05	F8 08	08→D
	07	B2 B6 B8	08→R2.1,R6.1,R8.1
	0A	F8 1C A1	"INT"→R1.0
	0D	F8 BF A2	"STK" R2.0
	10	F8 6B A4	"CALL"→R4.0
	13	F8 03 B5	03→R5.1
	16	D4	GO TO→0300 (MACRO)
0EI →	0017	7A	0→Q
END I →	0018	42 F6	STK→D,STK+1,D→DF
	1A	42 70	STK→D,STK+1,STK→XP,STK+1
INT →	001C	22 78	STK-1,T→STK
	1E	22 73	STK-1,D→STK,STK-1
	20	C0 00 23	LONG GO TO
	23	7E 52	DF→D→STK
	25	19	R9+1
	26	F8 09 B0	09→R0.1
	29	F8 D0 A8	"DO"→R8.0 (VF+1)
	2C	8B A0I E2	RB.0→R0.0
	2F	20 A0I E2	
	32	20 A0I E2	(I=8 DMA CYCLES)
	35	20 A0I 80	
	38	20 A0I	
	3A	3C 2F	EF1=1? NO
	3C	20 A0I	
	3E	34 3C	EF1=1? YES
	40	28	R8-1
	41	08 32 47	M8→D,D=00? YES
	44	FF 01 58	D-01→M8
	47	88 FB CD	R8.0→D,D=CD
	4A	3A 40	D=00? NO
	4C	08 32 17	M8 D,D=00? YES
	4F	7B 30 18	1→Q, GO TO END I
CXKK →	0052	19 89 AE	9+1,R9.0→RE.0
	55	93 BE	00→RE.1
	57	99 EE F4	R9.1→D,E→X,D+ME
	5A	56	D→M6
	5B	F6 E6 F4	SH.R,6→X,D+M6
	5E	B9 56	D→R9.1→M6
	60	45	M5→D.5+1
	61	F2 56 D4	D•M6→M6,RET
	64	00	SPARE
	5	00	
TV ON →	0066	E2 22 69	2→X,STK-1, TV ON (BUS→STK,D)
	69	12 D4	STK+1, RETURN

LABEL	LOCATION	CODE	DESCRIPTION
CALL →	006B	96 B7	08→R7.1
P=4	6D	94 BC	00→R1.C (MACRO DECODE TABLE)
	6F	45 AC AF	M5→D,5+1,D→RC.0,D→RF.0
	72	F6 F6	SHR,SHR
	74	F6 F6	SHR,SHR
	76	32 94	D=00? YES → M.L.
	78	F9 E0 AC	D/E0→RC.0
	7B	8F FA 0F	RF.0→D,D·0F
	7E	F9 C0 A6	D/C0→R6.0 [08CX→R6]
	81	05 F6 F6	M5→D,SHR,SHR
	84	F6 F6	SHR,SHR
	86	F9 C0 A7	D/C0→R7.0 [08CY→R7]
	89	4C B3 8C	MC→D,C+1,D→R3.1,RC.0→D
	8C	FC 0F AC	D+0F→RC.0
	8F	4C	MC→D.C+1
	90	A3 D3	D→R3.0,3→P→M.L.
	92	30 6B	GO TO → CALL
M.L. →	0094	8F	RF.0→D
	95	FA 0F B3	D•0F→R3.1
	98	45 30 90	M5→D,5+1, GO TO
[8XYN] →	009B	22 E2	STK-1,2→X
	9D	F8 D3 73	"D3"→STK,STK-1
	A0	45 F9 F0	M5→D,5+1,D/F0
	A3	52 E6	D→STK,6→X
	A5	47 D2	M7→D,7+1,2→P (EXEC. 2 STK. INSTR.)
	A7	56	D→M6 (RESULT VX)
	A8	F8 CB A6	"CB"→R6.0
	AB	91 7E	00→D,DF→D,VB
	AD	56 D4	D→M6, RETURN
[IC] →	00AF	86 FB C0	R6.0→D,D@C0
	B2	3A 52	D=00? NO → CXKK
[C0] →	B4	42 B5	STK→R5.1,STK+1
	B6	42 A5 D4	STK→R5.0,STK+1,RET.
[6XKK] →	00B9	45 56 D4	M5→M6,5+1,RET.
	00BC	64	100_{10}
	00BD	0A	10_{10}
	00BE	01	1_{10}
[3XMM]	00BF	06 3A C7	M6→D, D=00? NO
GO TO MM IF VX≠00	C2	15 D4	5+1, RETURN
[4XMM]	00C4	06 3A C2	M6→D,D=00? NO
GO TO MM IF VX=00	C7	05 A5 D4	M5→D,D→R5.0 RETURN
[2MMM]	00CA	15 85 22	5+1,R5.0→D,STK-1
BR. & LINK	CD	52 95 22	D→STK,R5.1 D,STK-1
	D0	52 25	D→STK,5-1
[1MMM]	00D2	45 A5	M5→D,5+1,D R5.0
GO TO 0MMM	D4	86 FA 0F	R6.0→D,D•0F
	D7	B5 D4	D→R5.1, RETURN

LABEL	LOCATION	CODE	DESCRIPTION
AMMM →	00D9	86 FA OF	R6.0→D,D·OF
0MMM→A	DC	BA	D→RA.1
	DD	45 AA D4	M5→RA.0,5+1,RETURN
I TABLE →	00E0	00	
R1	1	00	
	2	00	
	3	00	
	4	00	
	5	02	
	6	00	
	7	02	
	8	00	
	9	02	
	A	00	
	B	02	
	C	00	
	D	02	
	E	01	
	F	02	
I TABLE →	00F0	00	
R0	1	D2	
	2	CA	
	3	BF	
	4	C4	
	5	4E	
	6	B9	
	7	3D	
	8	9B	
	9	56	
	A	D9	
	B	E5	
	C	AF	
	D	BF	
	E	00	
	F	A4	

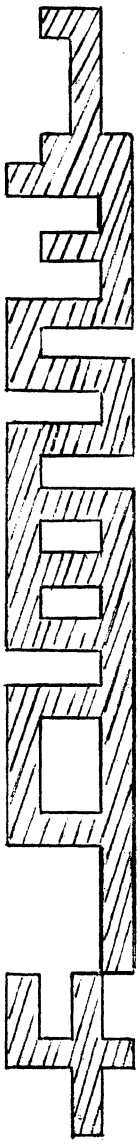
LABEL	LOCATION	CODE	DESCRIPTION
EN ⟶	0100	F8 C9 A7	"C9"→R7.0
P=3	03	07 FE FE	M7→D,SHL,SHL
	06	FE FE A6	SHL,SHL,D→R6.0
	09	F8 D0	"D0"→D
	0B	E7 F4 A7	7→X,MX+D→R7.0
	0E	F8 02 BC	02→RC.1
	11	F8 92 AC	"SET RF"→RC.0
	14	DC 10	SET RF=DIRECTION BYTE $+16_{10}$
	16	0F BD	MF→RD.1
	18	DC 08 0F	SET RF=JH,MF→D $+8_{10}$
	1B	FA 0F AE	D•0F RE.0
	1E	0F	MF→D
	1F	FA 80 BE	D•80 RE.1
	22	07 AD	M7→RD.0
	24	25 45	5-1,M5→D,5+1
	26	F6 33 4E	SHR,DF=1? YES ⟶E1
	29	F6 33 49	SHR,DF=1? YES ⟶E2
	2C	F6 33 3D	SHR,DF=1? YES ⟶E4
	2F	F6 33 BC	SHR,DF=1? YES ⟶E8MM
EO ⟶	0132	F8 10 AF	10→RF.0=N
ERASE V8	35	91 56 16	00→M6,6+1 ⟵
RAM PATTERN	38	2F 8F	RF-1,RF.0→D
(ALL 16 BYTES)	3A	3A 35	D=00? NO⟶
	3C	D4	RETURN ⟵
E4 ⟶	013D	E6	6→X
	3E	8E 32 3C	→RE.0→D,D=00? YES, H=00⟶
M(A)H BYTES	41	4A F3 56	MA→D,D⊕M6→M6,A+1
RAM (XOR)	44	16 16 2E	6+1,6+1,E-1
SKIP IF H=0	47	30 3E	⟶REPEAT
E2 ⟶	0149	F8 CC AF	CC→RF.0
	4C	0F BD	MF→RD.1 RD.1=VC DIRECTION BYTE
E1 ⟶	014E	9D FB 02	RD.1→D,D⊕02
SHIFT RAM	51	32 63	D=00? YES⟶SH.V.
PATTERN	53	9D FB 08	RD.1→D, D⊕08
	56	32 6D	D=00? YES⟶SH.D.
	58	9D FB 04	RD.1→D,D⊕04
	5B	32 97	D=00? YES⟶SH.L
	5D	9D FB 06	RD.1→D,D⊕06
	60	32 72	D=00? YES⟶SH.R.
	62	D4	RETURN (DIR.≠02/04/06/08)

LABEL	LOCATION	CODE	DESCRIPTION
SHV	0163	07 FF 08	M7→D(TVA) D-08 TVA-8
	66	57	D→M7 ←+24 10
	67	DC 18	SET RF=NV
	69	93 F5 5F	01→D,MX-D→MF NV-1
	6C	D4	RETURN
SHD	016D	07 FC 08	M7→D(TVA),D+08 TVA+8
	70	30 66	GO TO ——→
SHR	0172	8E 32 F9	→RE.0→D,D=00? YES H=00→END SR/SL
	75	06 F6 56	M6→D,SHR→DF,D→M6
	78	3B 81	DF=1? NO ——→ X0
	7A	9E 3A 81	RE.1→D,D=00? NO ——→ J1
	7D	F8 80	80→D
	7F	BE 1D	D→RE.1,RD+1 [1→J,TVA+1]
	81	16 06	6+1,M6→D ←
	83	76 56	SHR DF→D→DF,D→M6
	85	3B 93	DF+1? NO ——→ Y0
	87	26 06	6-1,M6→D
	89	F9 80	D/80→D
	8B	56 16	D→M6,6+1
	8D	9E 32 93	RE.1→D,D=00? YES ——→ J0
	90	91 BE 1D	00→RE.1,RD+1 [0→J,TVA+1]
	93	16 2E	6+1,E-1 [H-1] ←
	95	30 72	REPEAT
SHL	0197	8E 32 F9	→RE.0→D,D=00? YES H=00→END SR/SL
	9A	06 FE 56	M6→D,SHL→DF,D→M6
	9D	3B A5	DF=1? NO ——→ X0
	9F	9E 32 A5	RE.1→D,D=00? YES ——→ J0
	A2	91 BE 2D	00→RE.1,RD-1 [0→J,TVA-1]
	A5	16 06	6+1,M6→D ←
	A7	7E 56	SHL DF→D→DF,D→M6
	A9	3B B8	DF=1? NO ——→ Y0
	AB	26 06	6-1,M6→D
	AD	F9 01	D/01→D
	AF	56 16	D→M6,6+1
	B1	9E 3A B8	RE.1→D,D=00? NO ——→ J1
	B4	F8 80	80→D
	B6	BE 2D	D→RE.1,RD-1 [1→J,TVA-1]
	B8	16 2E	6+1,E-1 ←
	BA	30 97	REPEAT
	01BC		

LABEL	LOCATION	CODE	DESCRIPTION
E8MM →	01BC	93 BC	01→RC.1
MOVE RAM	01BE	F8 EB AC	XX→RC.0
PATT→TV	C1	91 AF	00→RF.0 [RF.0=HIT FLAG]
GO TO MM	C3	9E 32 D4	RE.1→D,D=00? YES, J=00
IF HIT	C6	2D	RD-1 [TVA_0-1]
	C7	8E 32 E1	RE.0→D,D=00? YES,H=00 END MOVE
	CA	DC 1D	DOXXM6→MD, 6+1,D+1
	CC	DC 8D	DOXXM6→MD, 6+1,RD.0→D
	CE	FC 07 AD	D+07→RD.0
	D1	2E 30 C7	E-1 [H-1] , REPEAT
	D4	8E 32 E1	RE.0→D,D=00? YES,H=00 END MOVE
	D7	DC 2D	DOXXM6→MD,6+1,D-1
	D9	DC 8D	DOXXM6→MD,6+1,RD.0→D
	DB	FC 09 AD	D+09→RD.0
	DE	2E 30 D4	E-1[H-1] REPEAT
END MOVE →	01E1	8F 3A E6	RF.0=00? NO PREV. HIT
	E4	15 D4	5+1,RET.
	E6	05 A5 D4	M5→D,D→R5.0,RET.
	01E9	16 D3	6+1,RET. TO P3
XX →	01EB	F8 09 BD	09→RD.1
M6→MD	FE	ED 06 F2	D→X,M6→D, D·MX
P=C	F1	32 F4	D=00? YES - NO HIT
	F3	AF	D→RF.0 [HIT FLAG]
	F4	06 F3 5D	M6→D, D⊕MX→MX
	F7	30 E9	GO TO
END SR/SL →	C1F9	DC 20 OF	SET RF=NH BYTE $+32_{10}$,MF→D
	FC	FF 01 5F	D-01→MF [NH-1]
	FF	DC 08 OF	SET RF=JH BYTE + 8_{10},MF→D
	0202	FA OF	D·OF→D [0H]
	04	5F 9E F1	D→MF,RE.1→D,D/MF→D
	07	5F	D→MF [SET NEW J]
	08	8D 57 D4	RD.0→M7 [RESTORE TVA] RET.
KEY DEBOUNCE →	020B	22 F9	DO TONE + DELAY
	020D	23 C3	DO WAIT FOR A/B KEY OFF
	020F	C0	RETURN
DEC.PATT.	0210	2F	
TABLE	11	1A	
	12	25	
	13	1F	
	14	38	
	15	23	
	16	27	
	17	33	
	18	29	
	19	2B	

Decimal display pattern table (35 Bytes)

	M	Byte	Pattern 7 6 5 4 3 2 1 0
"1" →	021A	60	
	1B	20	
	1C	20	
	1D	20	
	1E	70	
"3" →	021F	F0	
		10	
		70	
		10	
"5" →	0223	F0	
		80	
"2" →	0225	F0	
	26	10	
"6" →	0227	F0	
	28	80	
"8" →	0229	F0	
	2A	90	
"9" →	022B	F0	
	2C	90	
	2D	F0	
	2E	10	
"0" →	022F	F0	
	30	90	
	31	90	
	32	90	
"7" →	0233	F0	
	34	10	
	35	10	
	36	10	
	37	10	
"4" →	0238	A0	
		A0	
		F0	
		20	
		20	



LABEL	LOCATION	CODE	DESCRIPTION
[7XKK] ⟶	023D	86 FA OF	R6.0→D,D•OF
VX+KK	40	32 47	D=00? YES ⟶ 70MM
VX≠V0	42	E6 45	6→X,M5→D,5+1
	44	F4 56 D4	MX+D→M6,RET.
V0-1	47	06 FF 01	M6→D,D-01
GO TO MM	4A	3A E1	D=00? NO ⟶ [D→M6,M5→R5.0
IF VX=V0	4C	15	5+1 [RETURN]
	4D	D4	RET.
[5XKK] ⟶	024E	45	M5→D,5+1
SKIP 2	4F	E6 F3	6→X,MX⊕D
IF VX≠KK	51	32 4D	D=00? YES→RET.
	53	15 15 D4	5+1,5+1,RET.
[9XYN] ⟶	0256	96 BC	08→RC.1
	0259	07 AC 45	VY→RC.0,M5→D,5+1
	5B	F6 33 9B	SHR,DF=1? YES ⟶ 9XY1
	5E	F6 33 6A	SHR,DF=1? YES ⟶ 9XY2
	61	F6 33 6D	SHR,DF=1? YES ⟶ 9XY4
	64	F6 33 70	SHR,DF=1? YES ⟶ 9XY8
	67	07 30 4F	M7→D, GO TO
9XY2 ⟶	026A	0C 56 D4	MC→M6,RET.
M(08VY)→VX			
9XY4 ⟶	026D	06 5C D4	M6→MC,RET.
VX→M(08VY)			
9XY8 ⟶	0270	E6 06 BF	6→X,M6→RF.1 (VX→RF.1)
CONV.VX	73	91 BE	00→RE.1
TO 3 DEC.DIGITS	75	F8 BC AE	BC→RE.0 [M(E)=100$_{10}$]
M(08VY,VY+1,VY+2)	78	2C	C-1
	79	1C 91 5C	C+1,00→MC
	7C	0E F5	ME→D,MX-D
	7E	3B 87	DF=1? NO ⟶ OVERFLOW
	80	56 0C	D→M6,MC→D
	82	FC 01 5C	D+01→MC
	85	30 7C	REPEAT
	87	4E F6	ME→D,E+1,SHR
	89	3B 79	DF=1? NO
	8B	9F 56	RF.1→M6 [RESTORE VX]
	8D	8C 57 D4	RC.0→M7,RET. [VY +2→VY]
	0290	EF D3	F→X,RET TO P3
R7+KK→RF ⟶	0292	96 BF	08→RF.1
P=C	4	87 E3	R7.0→D.3→X
[SET RF]	6	F4 AF 13	MX+D→RF.0,3+1
	9	30 90	GO TO RET.
9XY1 ⟶	029B	06 57 D4	VX→VY,RET.

	LABEL	LOCATION	CODE	DESCRIPTION
*	REPEAT PATT → V0 TIMES	029E	23 36 E2	DO WR.PATT,SH
		2A1	70 9E	V0-1≠00 →
		2A3	C0	RETURN
	FXNN →	02A4	45 A3	M5→R3.0,5+1
	FXA6 →	02A6	0A 56 D4	MA→M6,RET.
	FXA9 →	02A9	06 5A D4	M6→MA,RET.
	FXAC →	02AC	4A 56 D4	MA→M6,A+1,RET.
	FXAF →	02AF	06 5A	M6→MA
		02B1	1A D4	A+1,RET.
	FXB3 →	02B3	06 AA D4	M6→RA.0,RET.
	FXB6 →	02B6	06 FA 0F	M6→D,D•0F
		B9	56 E6	D→M6,6→X
		BB	8A F1	RA.0→D,D/MX
		BD	AA D4	D→RA.0,RET.
	DKMM →	02BF	F8 CB A7	"CB"→R7.0
		C2	86 FA 0F	R6.0→D,D•0F
		C5	AF FB 0F	D→RF.0,D⊕0F
		C8	3A CC	D=00? NO →
		CA	07 AF	M7→RF.0 VB→RF.0
		CC	E2 22	2→X,STK-1 ←
		CE	8F 52 62	RF.0→STK,STK→KEY,STK+1 VB→KEY
		D1	F8 CA A6	"CA"→R6.0
		D4	06 32 DB	M6→D,D=00? YES →
		D7	36 DD	EF3=1? YES → A ON →
		D9	15 D4	5+1,RET. ←
		DB	3F D9	EF4=1? NO →
		DD	8F 57	RF.0→M7 ←
		DF	30 E2	SKIP →
	FROM 70MM →	02E1	56	D→M6
		E2	05 A5 D4	M5→R5.0,RET. ←
	BNKK →	02E5	455A	M5→D,5+1,D→MA
		E7	22 E2	STK-1,2→X
		E9	86 FA 0F	R6.0→D,D•0F→STK
		EC	52 8A F4	RA.0→D,D+MX→RA.0
		EF	AA 12 D4	STK+1,RET.
	CLEAR RAM →	02F2	91 5A	00→MA ←
		4	8A 2A	RA.0→D,A-1
		6	3A F2	D=00? NO →
		8	D4	RET.
*	30MS 160MS TONE + DELAY →	02F9	6D 02	05→VD(SET 30MS TONE)
		FB	6E 0A	0A→VE(SET 160MS DELAY)
		FD	3E FD	VE≠00↻
		FF	C0	RETURN

* STUDIO II LANGUAGE

LABEL	LOCATION	CODE	DESCRIPTION
* INITIALIZE ⟶	0300	6D 04	04→VD (TONE)
RAM	02	A9 FF	09FF→A
&	04	02 F2	CLEAR RAM (TV PAGE)
PATT.	06	A8 9F	089F→A
TABLES	08	02 F2	CLEAR RAM (TOP OF STACK & RAM PATTERNS)
	0A	00 66	TV ON
	0C	A8 D2	SET A=8D2 (TVA 2)
	0E	B2 F8	F8→TVA2,+2
	10	B1 15	15→TVA4,+1
	12	B3 11	11→TVA5,+3
	14	B1 04	04→JH0,+1
	16	B1 04	04→JH1,+1
	18	B1 01	01→JH2,+1
	1A	B1 02	02→JH3,+1
	1C	B1 05	05→JH4,+1
	1E	B0 05	05→JH5,+0
	20	A3 2B	SET A=PL1
	22	69 04	04→V9
	24	79 FF	MA→RAM (A+H) ↩
	25	E4	V9-1
	27	39 24	V9≠00 ⟵
	29	14 00	GO TO 400 (GAME SELECT)
PL1 ⟶	032B	03 03	
	2D	80	
	2E	18 18	
	30	18 18	
	32	18 18	
	34	18 18	
* WRITE PATT ⟶	0336	E8 3A	RAM V9→TV HIT ↩
	38	13 3C	SKIP
	3A	E8 3C	RAM V9→TV
	3C	C0	RETURN
* WARBLE TONE ⟶	033D	60 0F	0F→V0
	3F	22 F9	DO TONE + DELAY ↩
	41	70 3F	V0-1≠00 ⟵
	43	C0	RETURN
* DRAW □ ⟶	0344	69 02	02→V9
	0346	6C 06	RIGHT→VC
	48	60 7F	127→V0
	4A	22 9E	DO REPEAT_PATT. ⟹ 10
	4C	23 54	DO DRAW↓ 32
XX1 ⟶	034E	6C 06	RIGHT→VC
	50	E2	SH f(VC)
	0351	23 54	DO DRAW↓ 32
	53	C0	RETURN

* STUDIO II LANGUAGE

	LABEL	LOCATION	CODE	DESCRIPTION
*	DRAW ↓ 32 ——→	0354	6C 08	DOWN→VC
		56	60 20	32_{10}→V0
		58	22 9E	DO REPEAT PATT.
		5A	C0	RETURN
*	DRAW NET ——→	035B	69 03	03→V9
	MUST FOLLOW	5D	A8 D3	8D3→A
	DRAW ☐	5F	B0 03	03→TVA
		61	13 4E	GO TO→ XX1
	RB-08 ——→	0363	8B FF 08	RB.0→D,D-08
		66	AB D4	D→RB.0,RET.
*	SHOW A/B SCORE——→	0368	69 04	04→V9
	f(VA)	6A	89 A4	V9+VA→V9 [V9=A/B SCORE RAM]
	3 DIGIT SCORE	6C	23 88	DO A/B SCORE→V3
		6E	62 82	82→V2
		70	93 28	V3→M(0882,83,84) 3 DEC.DIGITS
*	CONV. & SHOW——→	0372	E0	ERASE RAM V9
	V9	73	60 02	02→V0
		75	6C 06	06→VC (RIGHT)
		77	23 96	DO CONVERT M (08V2) DIGIT,V2-1
		79	E4	MA→RAM V9
		7A	E2 E2	SH,SH
		7C	E2 E2	SH,SH
		7E	E2 E2	SH,SH
		80	70 77	V0-1≠00 ←
		82	23 96	DO CONVERT M(08V2) DIGIT,V2-1
		84	E4	MA→RAM V9
		85	E8 87	RAM V9→TV
		87	C0	RETURN
*	A/B SCORE→V3——→	0388	62 80	80→V2
		8A	82 A4	V2+VA→V2, VB M(08V2)=A/B SCORE BYTE
		8C	93 22	M(08V2)→V3
		8E	C0	RETURN
*	A/B SCORE+V1——→	038F	23 88	DO A/B SCORE→V3
		91	83 14	V3+V1→V3
		93	93 24	V3→M (08V2)
		95	C0	RETURN
*	CONVERT M(08V2)→	0396	A2 10	210→A=DEC.CONV.TABLE
	DEC.PATT.ADDRESS	98	93 22	M(08V2)→V3
	FOR M(08V2)DIGIT	9A	F3 B6	V3(LSD)→A(LSD)
	→A,V2-1	9C	F3 A6	MA→V3
		9E	F3 B3	V3→A LO
		A0	72 FF	V2-1
		A2	C0	RETURN

* STUDIO II LANGUAGE

LABEL	LOCATION	CODE	DESCRIPTION
* SHOW A & B SCORES +1.5 SEC DELAY →	03A3	6A 01	01→VA
	A5	23 68	DO SHOW A/B SCORE
	A7	6A 00	00→VA
	A9	23 68	DO SHOW A/B SCORE
	AB	6E 7F	7F→VE
	AD	3E AD	VE≠00 ↺
	AF	C0	RETURN
* STOP IF A/B SCORE≥V1 →	03B0	6A 01	01→VA
	B2	23 B9	DO V1 STOP TEST
	B4	6A 00	00→VA
	03B6	23 B9	DO V1 STOP TEST
	B8	C0	RETURN ←
* V1 STOP TEST →	03B9	23 88	DO A/B SCORE V3
	03BB	83 15	V3-V1→V3,VB
	BD	4B B8	VB=00 ———
*WARBLE STOP →	03BF	23 3D	DO WARBLE TONE
	C1	13 C1	STOP LOOP ↺
* WAIT FOR A/B KEY OFF →	03C3	4A C8	VA=00 ———
	C5	03 CB	DO MLA
	C7	C0	RETURN
	C8	03 CE	DO MLB ←
	CA	C0	RETURN
MLA →	CB	36 CB	EF3=1? YES ↺
	CD	D4	RET.
MLB →	CE	37 CE	EF4=1? YES ↺
	D0	D4	RET.
* ERASE & SHOW NEW A/B SCORE+V1 →	03D1	23 68	DO SHOW A/B SCORE f(VA) (ERASE)
	D3	23 8F	DO A/B SCORE + V1
	D5	23 68	DO SHOW NEW A/B SCORE
	D7	C0	RETURN
* DIR.KEY/00→VB→	03D8	D2 E3	KEY 2→VB →
	DA	D4 E3	KEY 4→VB →
	DC	D6 E3	KEY 6→VB →
	DE	D8 E3	KEY 8→VB →
	E0	6B 00	00→VB
	E2	C0	RETURN
	E3	22 0B	DO KEY DEBOUNCE ←
	E5	C0	RETURN

* STUDIO II LANGUAGE

LABEL	LOCATION	CODE	DESCRIPTION
* SHOW TVA LIST ⟶	03E6	61 D1	SET V1=TVA1 ADDR.
	E8	69 01	01→V9
	EA	F2 AC	MA→V2,A+1 ←
	EC	92 14	V2→M(08V1)
	EE	E8 FO	RAM 1 TV
	FO	70 EA	VO-1≠00 ——┘
	F2	CO	RETURN
	03F3	00	
* USED ⟶	03F4	22 0B	DO KEY DEBOUNCE
IN B KEY	6	92 11	V2→V1
TEST	8	23 D1	DO ERASE & SHOW A/B+V1
	A	61 64	
	C	23 B9	
	E	17 F8	GO TO 07F8 (KEY B TEST)

* STUDIO II LANGUAGE

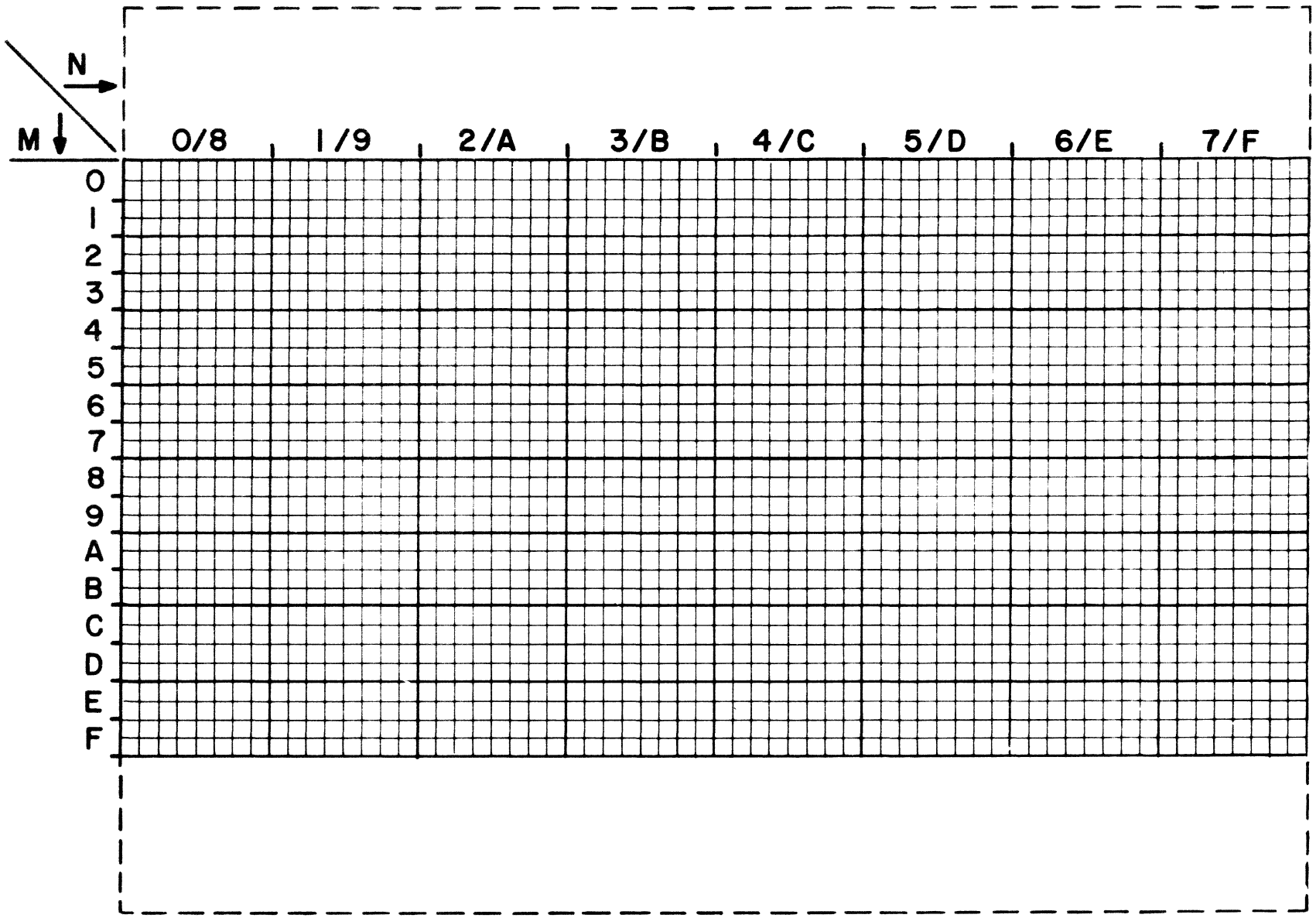


IMAGE STORAGE LOCATIONS, 09MN

0MMM - Execute Machine Language Routine at 0MMM, P = 3, D4 to Return
1MMM - Go to 0MMM
2MMM - Do Subroutine at 0MMM (C0 to Return)
C0 - Return from Subroutine
3XMM - Go to MM (Same Page) If VX ≠ 00
4XMM - Go to MM (Same Page) If VX = 00
5XKK - Skip 2 bytes if VX ≠ KK
6XKK - KK → VX
70MM - V0 -1, Go to MM (Same Page) If V0 ≠ 00 (Note V0 = 01 if No Branch)
7XKK - If X ≠ 0; VX + KK
8XY1 - VX/VY → VX, VB
8XY2 - VX·VY → VX, VB
8XY3 - VX ⊕ VY → VX, VB
8XY4 - VX + VY → VX (01 → VB If VX + VY > FF, 00 → VB If VX + VY ≤ FF)
8XY5 - VX - VY → VX (01 → VB If VX ≥ VY, 00 → VB If VX < VY)
9XY0 - Skip 2 Bytes If VX ≠ VY
9XY1 - VX → VY
9XY2 - M(08VY) → VX
9XY4 - VX → M(08VY)
9XY8 - VX → M(08VY , VY + 1, VY + 2) Decimal, VY(FINAL) = VY + 2
AMMM - 0MMM → A (Memory Pointer), for TVA (TV address).
BNKK - KK → M(A), A + N (Same Page)
CXKK - RR·KK → VX, X ≠ 0
DKMM - KEY(K) → VB · Go to MM (Same Page)
"A" Key if VA = 1, "B" Key if VA = 0, Do next instruction if KEY(K) is not pressed. If K = F, Go to MM if KEY(VB) is pressed.
E0 - Erase V9 RAM Pattern
E1 - Shift V9 RAM Pattern f (V9 Direction Byte), NV/NH - 1
Skip if direction ≠ U/D/L/R (2/8/4/6)
E2 - Shift V9 RAM Pattern f (VC), NV/NH-1, Skip if VC ≠ 2/8/4/6
E4 - M(A) H Bytes XOR V9 RAM → V9 RAM, A(FINAL) = A + H, Skip if H = 0
E8MM - Move V9 RAM Pattern → TV(XOR), Go to MM if Hit
FXD6 - VX(LSD) → A(LSD), VX(FINAL) - VX ·0F
FXA6 - M(A) → VX
FXA9 - VX → M(A)
FXB3 - VX → ALO
FXAC - M(A) → VX, A + 1
FXAF - VX → M(A), A + 1
FX4D - Set X (Preset R6 for 0MMM Next)
8XY6 - VY/2→VX, underflow→VB
8XYE - 2*VY→VX, overflow→VB

V9 = Current RAM Pattern # (00-07)
VA = A/B Key Flag (01 = A, 00 = B)
VB = Arithmetic Overflow/Key Input Byte
VC = Direction Byte
VD = Tone Timer (Q On if VD ≠ 00)
VE = Timer
VF = Timer

} Special-Purpose Variables

* Programs begin at 0400. In ROM systems 400-7FF program storage will be read only. Do not use 3 byte COSMAC instructions in machine language routines.