AT&T

# AT&T Display Enhancement Board

## Supplement To Systems Programmer's Guide

**NOTICE**

The information in this document is subject to change without notice. AT&T assumes no responsibility for any errors that may appear in this document.

# The Display Enhancement Board

## Table of Contents

# 1 DEB Capabilities

- Introduction
- The DEB Driver
- 16-Color Graphics
- Look-Up Table (LUT)
- Overlay Modes

# Introduction

The Display Enhancement Board (DEB) option adds improved color and graphics functionality to your AT&T PC 6300. When you use the DEB with the PC 6300 color monitor, you can display graphics in up to 16 color combinations simultaneously or treat the screen as two screens in one and overlay one screen treatment on top of the other. When you use the DEB with the PC 6300 monochrome monitor, you have the same capabilities you have with the color monitor, except that colors are displayed as "shades of green."

The DEB is compatible with existing software, so all the programs you have already can be used now as if the DEB were not installed. Of course, these programs may not have access to any of the new capabilities.

This supplement describes the functionality of the DEB device driver. Although it is not necessary to use the driver in order to use the DEB, the driver is designed to work with MS-DOS, GWBASIC, and other AT&T software products. If you wish to program the DEB hardware directly, you must consult the *AT&T Technical Reference Manual*. Such programming is considered a circumvention of the AT&T operating system and we advise against it.

This supplement assumes that you are familiar with video programming through the Interrupt 10H interface and with INTEL® 8086 assembler programming. Information on the Interrupt 10H interface can be found in the *System Programmer's Guide,* in the section on the ROM BIOS Service Routines.

Before you begin writing programs for the DEB, follow the procedures in the DEB Installation Manual for installing the DEB hardware and device driver software.

The DEB is an optional hardware component for the AT&T PC 6300 that works in conjunction with the PC 6300's built-in Video Display Controller (VDC) to provide improved color and graphics functionality.

The built-in VDC contains circuitry and memory that support either 4 color medium resolution ($320 \times 200$ pixels) graphics, 1 color high resolution ($640 \times 200$ pixels) graphics, or 1 color super resolution ($640 \times 400$ pixels) graphics.

The DEB contains additional circuitry and memory that can be combined with the capabilities of the built-in VDC to produce up to 16 color combinations in either high or super resolution. You can also program the VDC and DEB separately, treating them as two separate images that are combined on one screen to produce an overlaying effect. The overlay modes let you use up to 8 colors on the DEB screen and up to 16 colors on the VDC screen.

# The DEB Driver

You load the DEB device driver by entering a "DEVICE" statement in the CONFIG.SYS file (see **Chapter 2, Programming Tips**). The driver installs an Interrupt 10H "filter" during the loading process.

When you are using the DEB and are running some programs that use the DEB and some that do not, the "filter" provides video support for both kinds of programs. For programs that do not use the DEB, the filter passes control to the standard Interrupt 10H ROM BIOS routine.

The DEB driver installs a filter for Interrupt 9H. This filter resets the DEB to transparent mode whenever you warmstart the system through **CTRL/ALT/DEL**. The filter controls scrolling when you press **CTRL/NUMLOCK**.

## 16-Color Graphics

This feature lets you display 16 color combinations in either high resolution (640 × 200) or super resolution (640 × 400). Not only can you use the standard 16 colors, you can also combine colors to form new colors and cause pixels to blink from one color to another.

The DEB provides 5 palettes for you to use when programming in color. At any point in your program, you select one of the palettes as the "active" palette. The color combinations contained in that palette determine what colors and effects show on the screen.

Each of the first 4 palettes contains a default set of 16 color combinations, but to suit the needs of your program you can change the contents of the palette to any one of the following:

- any of the 16 standard colors with which you are already familiar from the standard applications. The standard colors are:

| | |
|---|---|
| 0 = black | 8 = gray |
| 1 = blue | 9 = light blue |
| 2 = green | 10 = light green |
| 3 = cyan | 11 = light cyan |
| 4 = red | 12 = light red |
| 5 = magenta | 13 = light magenta |
| 6 = brown | 14 = yellow |
| 7 = white | 15 = high intensity white |

- a mixture, or "dithering," of any 2 of the 16 standard colors
- an alternation, or blinking, between any 2 of the standard 16 colors

The last palette contains no default combinations. You program the fifth palette by loading color values into a 256-byte array. The DEB device driver uses this special palette to program the DEB's color Look-Up Table (LUT). By using the LUT you can add the capability of dithering or blinking between any four colors.

# Look-Up Table (LUT)

The LUT resides in RAM on the DEB board, and is accessed through write-only hardware registers. The device driver keeps a copy of the register values in the LUT. The register values are accessible to software applications through the device driver. The LUT contains 256 values that determine the colors, blinking, and dithering that appear on the screen. Whether you need to learn about the use and layout of the LUT depends on the application you are writing.

If you use the standard palettes, you need not be concerned with the LUT. The DEB device driver automatically programs the LUT to correspond to the way you set up the palettes. If you program a custom LUT, you greatly increase the color combinations and blinking effects available to you.

# Overlay Modes

The overlay modes let you use the screen to display two images at once, independently. For example, you can display a high resolution color graphics image with its own foreground and background. Then, on "top" of that image, you can display a box of text and scroll the text without affecting the graphics image.

The overlay modes use the DEB to control one image and use the standard controller board to control the other image. You can select from many combinations of graphics, text, color, and high or super resolution in designing the two images.

The overlay modes offer 5 palettes. Each of the first 4 palettes has 8 positions. These four palettes have default colors that you can change to suit your needs. You can choose 8 color combinations from any of the 16 standard colors, or blink between 2 of the standard colors. The dithering combinations of the 16-color graphics modes are not available. You can also use the last palette to custom program the LUT.

# 2 Programming Tips

- Presence of Hardware/Software
- Hardware/Software Compatibility
- Setup

# Presence of Hardware/Software

Whenever you plan an application, it is important to use the DEB device driver to test for the presence of both the DEB and the associated driver. Test for the presence of the hardware by checking for DEB video memory. This is accomplished by writing and reading back data patterns into memory, in the range A000H:0H to B800H:0H. Test for the software device driver by issuing a function call to open the device called "DEBDRIVE," then immediately issuing a call to close "DEBDRIVE." If the open fails (carry set on return from Interrupt 21H) then the driver is not present. No functions are implemented in the driver, which is used only to detect the presence of the software.

# Hardware/Software Compatibility

The driver software has been designed to fit into the structure of MS-DOS programs. The DEB hardware uses the same range of addresses as the standard video ports on any compatible machine. If your application uses a light pen, consult the DEB supplement in the *AT&T Personal Computer Technical Reference Guide.*

The DEB driver makes minor modifications to the ROM BIOS video interrupt. Mode setting and color selection offer additional functionality. Be careful when you use the following functions.

- SET MODE — uses an additional register BL
- SCROLLING — uses an additional register BH
- STATUS — returns an additional register pair ES:DI. No application should count on ES:DI not changing.

# Setup

Install the DEB driver just as you would install any device driver. Be sure the CONFIG.SYS file is in the root directory. Put the line DEVICE = DEDRIVER.DEV in CONFIG.SYS. This line puts the DEB driver in the device driver chain. The driver makes patches in INT 10H and INT 9H to add the new functionality. The driver has two features:

- the INIT function, which deallocates itself after it runs
- chaining, which allows you to test for the driver's presence by issuing an open function call

# 3 How to Program the DEB

- Overview
- Mode Setting
- Setting Colors and Effects
- Displaying Graphics Images

# Overview

There are three steps for video programming that apply whether or not you are using the DEB capability:

1. Set the hardware's mode. You also must set the active page if you are in an overlay mode and want to select the DEB screen.

2. Select the color combinations and effects you want to use.

3. Construct the graphics images you want to display.

This chapter describes each of these steps in detail. This chapter does **not** describe how to program the LUT directly (see **Chapter 5, "Programming the LUT"**).

# Mode Setting

The DEB is controlled by invoking one of the DEB video modes through the Set Mode function (INT 10H, function 0H). The Set Mode function establishes the mode for both the DEB and the VDC. These modes fall into four categories: 16-color graphics, overlay, transparent, and disabled.

## 16-Color Graphics Modes

There are two DEB modes that provide 16-color graphics: high resolution and super resolution. Both these modes let you use 5 palettes and display up to 16 color combinations simultaneously.

## Overlay Modes

When overlaying the VDC on the DEB output, you specify one of the modes for the VDC and one mode for the DEB. The VDC modes are a subset of the modes for non-DEB graphics: 80 × 25 text mode, high and super resolution modes. The DEB modes are both graphics modes: high and super resolution.

If you are using one of the four standard palettes, the VDC's output takes precedence over the output of the DEB, so that if each board writes a pixel to the same screen location, the pixel sent by the VDC is displayed. This precedence is programmed into the LUT. If you want to have the DEB take precedence over the VDC, you must change the values in the LUT. (For more information, see **Chapter 5, "Programming the LUT."**)

**Transparent Mode**

The non-DEB modes, modes 0-40H and mode 48H, work exactly as they work without the DEB device driver installed.

**Disabled Mode**

In the disabled mode, you can cause the output of the VDC, the DEB, or both to be blacked out. This allows you to draw a graphics image or to fill a screen with text and not have them displayed while you are building them. You can then have the image "pop up" by taking VDC or DEB out of the disabled mode. You can also achieve this result by using the programmable palettes and the LUT.

## Setting Colors and Effects

Colors and effects are controlled by the Set Color Palette command, (INT 10H function 0BH). Use this function to set color values in one of the four palettes, to switch between palettes, or to reset palettes to their default values. You also use Set Color Palette to program the LUT directly.

# Displaying Graphics Images

There are two methods for displaying graphics images using the DEB: writing dots at screen locations or directly programming the VDC and DEB memory.

To write dots (pixels) to the screen, use the Write Dot function (INT 10H, function 0CH). Write Dot requires that you specify the display page, the row and column where you want the dot to appear, and the color or pattern for the dot.

If you want to program the VDC and DEB graphics memory directly , you need to learn the details of how the LUT is structured and how LUT addresses are formed (see the section on "Programming the Bit Planes" in Chapter 5).

# 4 Interrupt 10H Functions

- Introduction
- Functions

# Introduction

The following section describes the DEB device driver software functions. This interface is an extension of the INT 10H software function to the PC6300 ROM BIOS that controls the VDC. The ROM BIOS screen driver has 16 functions:

| | |
|---|---|
| 0H | set the display mode |
| 2H | set the cursor position |
| 3H | read the cursor position |
| 5H | select the active display page |
| 6H | scroll the active page up |
| 7H | scroll the active page down |
| 8H | read character/attribute at the current cursor position |
| 9H | write character/attribute at cursor position |
| AH | write only the character at current cursor position |
| BH | change the color palette |
| CH | write a point on the screen |

DH read a point on the screen

EH write in teletype style to the active page

FH return information about the active video state

Not all these functions are applicable to the DEB. The filter receives the Interrupt 10H function call, filters the functions that are applicable to the DEB and performs them. The functions that are not applicable to the DEB are passed on to the ROM BIOS INT 10H routine or to a previously installed filter or driver routine. The following section describes the functions which are processed by the DEB Interrupt 10H filter.

# Functions

**Set Mode**

The function establishes the mode for both the DEB and the VDC. If you select a non-DEB related mode, control is passed to the ROM resident Set Mode function. Set Mode initializes palette 0 as the active palette.

**Input**

| | | |
|---|---|---|
| (AH) | = 0H | function number for Set Mode |
| (AL) | = new mode | |
| (BL) | = optional overlay mode | |

Setting AL bit 7 = 0 puts you in either the DEB transparent mode or 16-color graphics mode:

| | |
|---|---|
| (AL) = 0H | 40 × 25 monochrome, text |
| (AL) = 1H | 40 × 25 color, text |
| (AL) = 2H | 80 × 25 monochrome, text |
| (AL) = 3H | 80 × 25 color, text |
| (AL) = 4H | 320 × 200 color |
| (AL) = 5H | 320 × 200 monochrome |
| (AL) = 6H | 640 × 200 color |
| (AL) = 40H | 640 × 400 with 2-position programmable palette, defaulting to black and white |
| (AL) = 41H | 640 × 200 16-color graphics with four palettes |
| (AL) = 42H | 640 × 400 16-color graphics with four palettes |
| (AL) = 44H | Disable mode (disables both DEB and VDC output) |

Setting AL bit 7 = 1 puts you in overlay mode. The following values are only used in overlay mode. AL contains the setting for the VDC; BL contains the mode setting for the DEB. In overlay modes, the active page defaults to zero.

| | | |
|---|---|---|
| **VDC Settings** | (AL) = 82H | 80 × 25 monochrome, text |
| | (AL) = 83H | 80 × 25 color, text |
| | (AL) = 86H | 640 × 200 color graphics |
| | (AL) = 0C0H | 640 × 400 color graphics |
| | (AL) = 0C4H | Disable mode. Disables only the VDC. |
| **DEB settings** | (BL) = 6H | 640 × 200 graphics with four 8-position palettes. |
| | (BL) = 40H | 640 × 400 graphics with four 8-position palettes. |
| | (BL) = 44H | Disable mode. Disables only the DEB. |

**Output** Contents of all registers are preserved.

**Example**

```
MOV AH,0      ; Select Set Mode
MOV AL,41H    ; Select 16 color graphics
INT 10H       ; Change the mode
```

**Set Cursor Position**

This function sets the cursor position for either the DEB, the VDC, or both.

**Input**

| | | |
|---|---|---|
| (AH) | = | 2H Function number for Set Cursor Position |
| (DH,DL) | = | row, column of new position |
| (BH) | = | page number<br>Valid page numbers for DEB modes are 0 for the VDC and 80H for the DEB in overlay mode. Row values are 0 thru 23, column values are 0 thru 79, in DEB modes. |

**Output**

Contents of all registers are preserved.

**Example**

```
MOV AH,2    ; SCP function
MOV DH,ROW
MOV DL,COL
MOV BH,PAGE
INT 10H     ; Moves cursor to position defined in
              above variables.
```

**Read Cursor Position**

This function returns the position of the cursor for the DEB, VDC, or both.

**Input**

(AH) = 3H Function number for Read Cursor Position

(BH) = page number
Valid page numbers for DEB modes are 0 for VDC and 80H for the DEB in overlay mode. Row values are 0 thru 23, column values are 0 thru 79, in DEB modes.

**Output**

(DH,DL) = row, column of current position.

Contents of all other registers are preserved.

**Example**

```
MOV AH,3
MOV BH,PAGE
INT  10H
MOV ROW,DH
MOV COL,DL
```

**Select Active Display**

This command allows selection of the DEB display in overlay mode.

**Input**

(AH) = 5H Function number for Select Action Display
(AL) = active page
Values for the active page in DEB modes are, 0 for the VDC and 80H for the DEB.

**Output**

Contents of all registers are preserved.

**Example**

```
MOV AH,5
MOV AL,PAGE
INT  10H
```

**Scroll Active Page Up**

This function defines a pattern that is to be displayed on the blank lines as the screen scrolls. The pattern consists of ones and zeros. Zeros are interpreted as the background color (palette position zero). Ones are interpreted as the foreground color, which is defined in BL. Care should be taken when scrolling in DEB modes, to insure that all applications set the additional argument in BH correctly.

**Input**

| | | |
|---|---|---|
| (AH) | = | 6H function number for Scroll Active Page Up |
| (AL) | = | number of lines to scroll |
| (CH,CL) | = | row, column of upper left corner to scroll |
| (DH,DL) | = | row, column of lower right corner to scroll |
| (BH) | = | pattern to be used on blank lines |
| (BL) | = | foreground color |

The range of lines to be scrolled is 0 thru 23 (where 0 specifies clear screen).
Row values are 0 thru 23, column values are 0 thru 79, in DEB modes.
Valid foreground colors are specified by palette position 0-FH for 16-color graphics, and 0-7H for 8-color graphics.

**Output**

Contents of all registers are preserved.

**Example**

```
MOV AH,6   ;Scroll Active Page Up
MOV AL,LINES
MOV CH,UPROW
MOV CL,UPCOL
MOV DH,LOWROW
MOV DL,LOWCOL
MOV BH,0
MOV BL,FGCOLOR
INT  10H
```

**Scroll Active Page Down**

This function permits you to define a pattern that is to be displayed on the blank lines as the screen scrolls downward. The pattern consists of ones and zeros. Zeros are interpreted as the background color (palette position zero). Ones are interpreted as the foreground color, which is defined in BL. Care should be taken when scrolling in DEB modes, to insure that all applications set the additional argument in BH correctly.

**Input**

| | | |
|---|---|---|
| (AH) | = | 7H function number for Scroll Active Page Down |
| (AL) | = | number of lines to scroll |
| (CH,CL) | = | row, column of upper left corner to scroll |
| (DH,DL) | = | row, column of lower right corner to scroll |
| (BH) | = | pattern to be used on blank lines |
| (BL) | = | foreground color |

The range of lines to be scrolled is 0 thru 23 (where 0 specifies clear screen).

Row values are 0 thru 23, column values are 0 thru 79, in DEB modes.

Valid foreground colors are specified by palette position 0-FH for 16-color graphics, and 0-7H for 8-color graphics.

**Output**

Contents of all registers are preserved.

**Example**

```
MOV AH,7    ; Scroll Active Page Down
MOV AL,LINES
MOV CH,UPROW
MOV CL,UPCOL
MOV DH,LOWROW
MOV DL,LOWCOL
MOV BH,0
MOV BL,FGCOLOR
INT   10H
```

**Read Character and Attribute at Current Cursor Position**

This function returns the value of the character at the current cursor position. The value of the character's foreground color is returned in AH.

**Input**

(AH) = 8 Function number for Read Character and Attribute at Current Cursor Position.

(BH) = Valid page numbers for DEB modes are 0 for the VDC and 80H for the DEB in overlay mode.

**Output**

(AL) = ASCII character code

(AH) = foreground palette position or VDC attribute

Contents of all other registers are preserved.

**Example**

```
MOV AH,8    ;Read CHR function
MOV BH,PAGE
INT  10H
MOV CHAR,AL ;Save CHAR/COLOR
MOV CURCOLOR,AH
```

**Write Character and Attribute at Current Cursor Position**

This function displays the character whose ASCII code is in register AL. The character is displayed according to the color values in BL.

**Input**

(AH) = 9H Write Character function
(AL) = ASCII character code
(BL) = foreground color
(BH) = page
(CX) = count of characters to write

If bit 7 of BL = 1, the color value is XOR'd with the current dots in that location. Valid page numbers for DEB modes are 0 for the VDC and 80H for the DEB in overlay mode.

Valid foreground colors are specified by palette position 0-FH for 16-color graphics, and 0-7H for 8-color graphics.

**Output**

Contents of all registers are preserved.

**Example**

```
MOV AH,9
MOV AL,CHAR
MOV BL,CURCOLOR
MOV BH,PAGE
MOV CX,1
INT  10H
```

**Write Character Only at Current Cursor Position**

In DEB modes, this function is the same as "Write Character and Attribute."

**Set Color Palette**

This function is used to set color values in one of the four palettes, to switch between palettes, or to reset palettes to their default values.

In the overlay modes, the Set Color Palette function works on the active page. If the active page is set to display to the VDC board, this function works the same as the standard ROM BIOS INT 10H (function 0BH).

If you specify a palette position greater than the value allowed for the mode in which you are working, the value you specify will be put in that palette's highest position. For example, if you attempted to set palette position 13 to red when working in overlay mode, which has 8-position palettes, the 8th palette position would be set to red.

Note:
The following discussion covers the use of the simple palette programming functions. You can also use "Set Color Palette" to program the LUT. (For more information, see **Chapter 5, "Programming the LUT"**).

**Input**

| | | |
|---|---|---|
| (AH) | = | 0BH Function number for Set Color Palette |
| (AL) | = | palette function selector |
| (BH) | = | positional pointer |
| (BL) | = | color value |

For simple palette programming functions, use the following

| | | | |
|---|---|---|---|
| (AL) | = | 0 | |
| (BH) | = | palette color ID | |
| | | BH = FFH | switches to the palette specified in BL, without changing to the default palettes unless there is a change in palette type (e.g., change from a 16-position palette to an 8-position palette). |
| | | BH = 80H | switches to the palette specified in BL and resets the palette to its default. |
| | | BH = 0-16 | sets this palette position to the color or attribute in BL. |
| (BL) | = | actual color value or code for blinking and dithering | |

The special settings for using a customized LUT in Set Color Palette are as follows:

**Input**

(AL) = non-zero (a zero here selects a standard palette)

AL bit 0 = 1 means use ES:SI to program the palette and registers BH and BL to indicate an offset and length into the LUT. ES:SI points to the LUT table (in the above example, LUT-STRING).

In this case, BH = offset into LUT and BL = length of portion of LUT to be changed. If you are loading an entire new table, set BH and BL to 0.

AL bit 1 = 1 means use BH and BL to program the LUT one location at a time.

In this case, BH = position in LUT and BL = the value to put in that position.

AL bit 2 = 1 means use the short LUT addressing mode. (Only uses the first 16 LUT entries).

The DEB driver lets you automatically load your customized LUT and use it in place of one of the standard palettes.

The steps for loading and using the customized LUT are:

1. Define the table with DB (Define Byte) statements.
2. Load the table in by using the Set Color Palette command.
3. Use the Read Dot and Write Dot commands to access the LUT (see **Chapter 4, "Interrupt 10H Functions"**).

The code for defining the table would be similar to this:

```
LUT-STRING    DB  4            ! Signifies active palette 4
              DB  1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,
              DB  1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,
              DB  1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,
              DB  1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,
              .
              .
              .
              DB  1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,
```

To load a new table of values into the LUT, where the table in your program is named LUT-STRING, you can use these statements:

```
PUSH DS          ! save the data segment address
POP ES
MOV   S1,LUT-STRING
MOV   AL,1
MOV   AH,11
XOR   BH,BX      ! Sets BH=BL=0
INT   10
```

The defaults for each of the four palettes are:

**Default Palettes**

**Palette Number 0**

| Position | Color |
|---|---|
| 0 | 0 = black |
| 1 | 2 = green |
| 2 | 4 = red |
| 3 | 6 = brown |
| 4 | 1 = blue |
| 5 | 3 = cyan |
| 6 | 5 = magenta |
| 7 | 7 = white |
| 8 | 8 = gray |
| 9 | 9 = light blue |
| 10 | 10 = light green |
| 11 | 11 = light cyan |
| 12 | 12 = light red |
| 13 | 13 = light magenta |
| 14 | 14 = yellow |
| 15 | 15 = high-intensity white |

**Palette Number 1**

| Position | Color |
|---|---|
| 0 | 0 = black |
| 1 | 3 = cyan |
| 2 | 5 = magenta |
| 3 | 7 = white |
| 4 | 1 = blue |
| 5 | 2 = green |
| 6 | 4 = red |
| 7 | 6 = brown |
| 8 | 8 = gray |
| 9 | 9 = light blue |
| 10 | 10 = light green |
| 11 | 11 = light cyan |
| 12 | 12 = light red |
| 13 | 13 = light magenta |
| 14 | 14 = yellow |
| 15 | 15 = high-intensity white |

Palettes 2 and 3 are the same, and they contain the standard colors in numerical order.

**Palette Number 2 and Palette Number 3**

| Position | Color |
|---|---|
| 0 | 0 = black |
| 1 | 1 = blue |
| 2 | 2 = green |
| 3 | 3 = cyan |
| 4 | 4 = red |
| 5 | 5 = magenta |
| 6 | 6 = brown |
| 7 | 7 = white |
| 8 | 8 = gray |
| 9 | 9 = light blue |
| 10 | 10 = light green |
| 11 | 11 = light cyan |
| 12 | 12 = light red |
| 13 | 13 = light magenta |
| 14 | 14 = yellow |
| 15 | 15 = high-intensity white |

# DITHER COMBINATIONS FOR DEB PALETTES 0-3

Color combinations 136-255 have been pre-assigned to allow you easy access to dithering effects while using the standard palettes. The following table describes the available combinations.

| A ↓ / B → | black | blue | green | cyan | red | magenta | brown | white | gray | light blue | light green | light cyan | light red | light magenta | yellow |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| black | | | | | | | | | | | | | | | |
| blue | 136 | | | | | | | | | | | | | | |
| green | 137 | 138 | | | | | | | | | | | | | |
| cyan | 139 | 140 | 141 | | | | | | | | | | | | |
| red | 142 | 143 | 144 | 145 | | | | | | | | | | | |
| magenta | 146 | 147 | 148 | 149 | 150 | | | | | | | | | | |
| brown | 151 | 152 | 153 | 154 | 155 | 156 | | | | | | | | | |
| white | 157 | 158 | 159 | 160 | 161 | 162 | 163 | | | | | | | | |
| gray | 164 | 165 | 166 | 167 | 168 | 169 | 170 | 171 | | | | | | | |
| light blue | 172 | 173 | 174 | 175 | 176 | 177 | 178 | 179 | 180 | | | | | | |
| light green | 181 | 182 | 183 | 184 | 185 | 186 | 187 | 188 | 189 | 190 | | | | | |
| light cyan | 191 | 192 | 193 | 194 | 195 | 196 | 197 | 198 | 199 | 200 | 201 | | | | |
| light red | 202 | 203 | 204 | 205 | 206 | 207 | 208 | 209 | 210 | 211 | 212 | 213 | | | |
| light magenta | 214 | 215 | 216 | 217 | 218 | 219 | 220 | 221 | 222 | 223 | 224 | 225 | 226 | | |
| yellow | 227 | 228 | 229 | 230 | 231 | 232 | 233 | 234 | 235 | 236 | 237 | 238 | 239 | 240 | |
| high-intensity white | 241 | 242 | 243 | 244 | 245 | 246 | 247 | 248 | 249 | 250 | 251 | 252 | 253 | 254 | 255 |

NOTE: To select a value that combines colors A and B to create a new color, find the number at the intersection of row A and column B.

# BLINKING COLOR EFFECTS FOR DEB PALETTES 0-3

Color combinations 16-135 have been pre-assigned to allow you easy access to blinking effects while using the standard palettes. The following table describes the available combinations.

| A ↓ / B → | blue | green | cyan | red | magenta | brown | white | gray | light blue | light green | light cyan | light red | light magenta | yellow | high-intensity white |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| black | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| blue | | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 |
| green | | | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 |
| cyan | | | | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 |
| red | | | | | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 |
| magenta | | | | | | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 |
| brown | | | | | | | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 |
| white | | | | | | | | 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 |
| gray | | | | | | | | | 108 | 109 | 110 | 111 | 112 | 113 | 114 |
| light blue | | | | | | | | | | 115 | 116 | 117 | 118 | 119 | 120 |
| light green | | | | | | | | | | | 121 | 122 | 123 | 124 | 125 |
| light cyan | | | | | | | | | | | | 126 | 127 | 128 | 129 |
| light red | | | | | | | | | | | | | 130 | 131 | 132 |
| light magenta | | | | | | | | | | | | | | 133 | 134 |
| yellow | | | | | | | | | | | | | | | 135 |

NOTE: To select a value that will cause blinking between colors A and B, find the number at the intersection of row A and column B.

**Write Dot**

The Write function writes a pixel to the location on the screen that you specify. If the screen is in the DEB mode, Write Dot may also write a pattern.

**Input**

| | | |
|---|---|---|
| (AH) | = | 0CH, Function number for Write Dot |
| (AL) | = | Palette position to be written. |
| (BH) | = | display page designator (bit 7 = 1 selects the DEB) |
| (CX) | = | column number |
| (DX) | = | row number |

**Output**

Contents of all registers are preserved.

**Example**

```
MOV AH,0CH
MOV AL,PALPOS
MOV BH,PAGE
MOV CX,COL
MOV DX,ROW
INT   ;Write the Dot
```

**Read Dot**

This function reads a dot from the screen. If the screen is in the DEB mode this function returns the value in the LUT that corresponds to this dot. (For more information, see **Chapter 5, "Programming the LUT."**)

**Input**

(AH) = 0DH Function number for Read Dot
(BH) = display page designator (bit 7 = 1 selects the DEB)
(CX) = column number
(DX) = row number

**Output**

(AH) = VDC value or DEB palette position
Contents of all other registers are preserved.

**Example**

```
MOV AH,0DH
MOV BH,PAGE
MOV CX,COL
MOV DX,ROW
INT
MOV DOTCOL,AH    ;Save the Dot
```

**Input** (AH) = 0EH, Function number for Write Teletype
(AL) = character to write
(BL) = foreground color (in graphics modes)
If bit 7 = 1, color is XOR'd to current contents.

**Output** Contents of all registers are preserved.

**Example**

```
MOV AH,0EH
MOV AL,CHAR
MOV BL,FGCOL
INT   10H
```

| | |
|---|---|
| **Read Current Video State** | This function returns the current video state. It indicates whether the DEB or VDC is active in the overlay mode and returns the number of the active palette. |
| **Input** | (AH) = 0FH Function number for Read Current Video State |
| **Output** | (BH) = display page designator<br>(AL) = mode currently set<br>(ES:DI) = pointer to a copy of the current LUT |
| **Example** | **MOV AH,0FH**<br>**INT 10H** |

# 5 Programming the LUT

- Overview
- 16-Color Graphics LUT Programming
- Overlay Modes LUT Programming
- Programming the Bit Planes

# Overview

This chapter describes programming the DEB Look-Up Table (LUT). By programming the LUT yourself, you can create color patterns that are not available when you use standard palettes. You need not read this chapter if you do not want to use this extended functionality.

The hardware uses the LUT to translate the contents of video memory patterns into graphics effects. In the standard palettes, INT 10H filter programs the LUT for you and thereby provides the preassigned color combinations and effects as described in previous chapters.

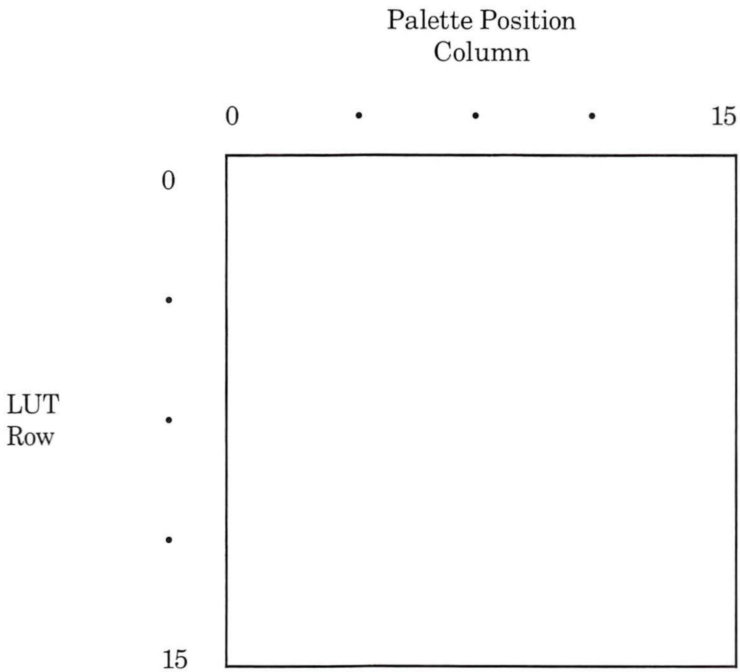
To program the LUT directly, you select Palette 4 in Set Color Palette function. Palette 4, also called the "LUT palette," has a minimum of 256 positions. Each palette position contains a value between 0 and 15. These values map into the LUT locations on the DEB. The 256 locations on the DEB collectively determine the color and special effects displayed when you specify a particular palette position. The color and special effect for each pixel on the screen are determined by:

- the palette position you specify
- the values in the LUT
- the active mode

There are some differences in the way the LUT is structured for 16-color graphics modes and overlay modes. This chapter describes LUT operation for 16-color graphics modes and overlay modes separately.

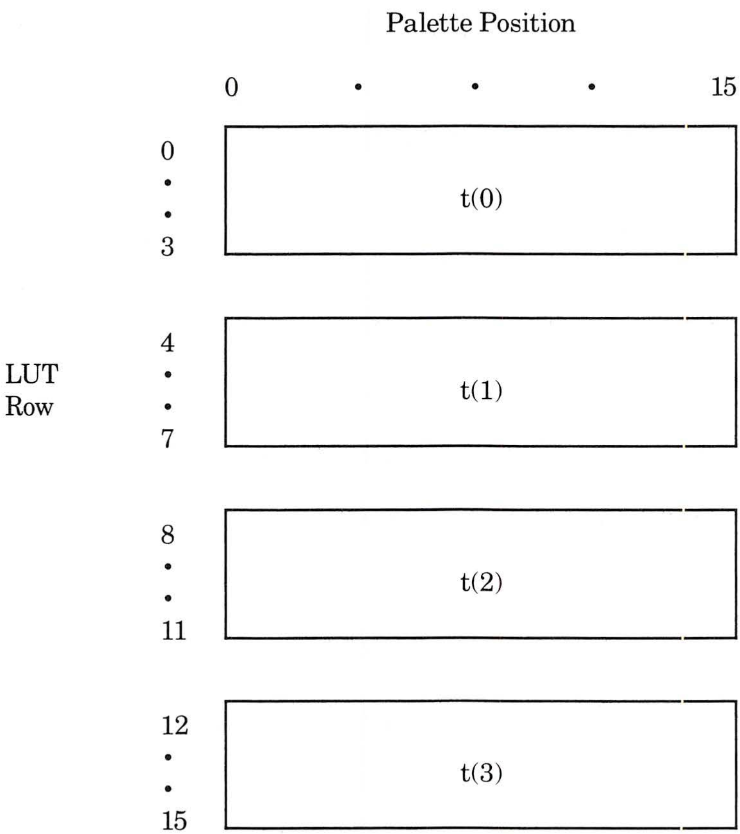
# 16-Color Graphics Lut Programming

In these modes, the LUT can be viewed as a two-dimensional array (16 × 16). Each location contains one of the standard 16 colors.



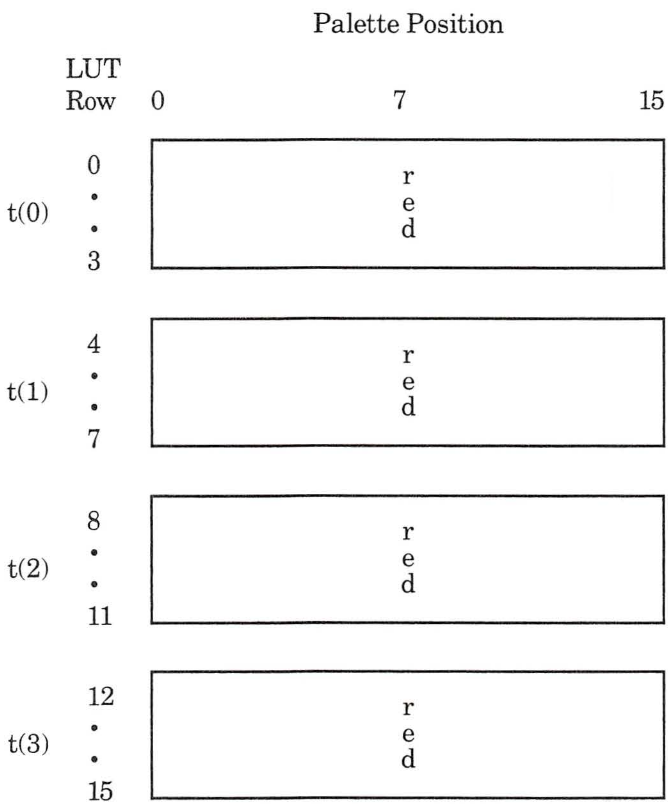

The locations in the LUT are numbered consecutively from left to right and top to bottom. Thus, location 17 corresponds to Row 1, palette position 1.

In the 16-color graphics mode, the LUT is divided into four "time states." At any one time, only one quarter of the LUT determines the display on the screen.



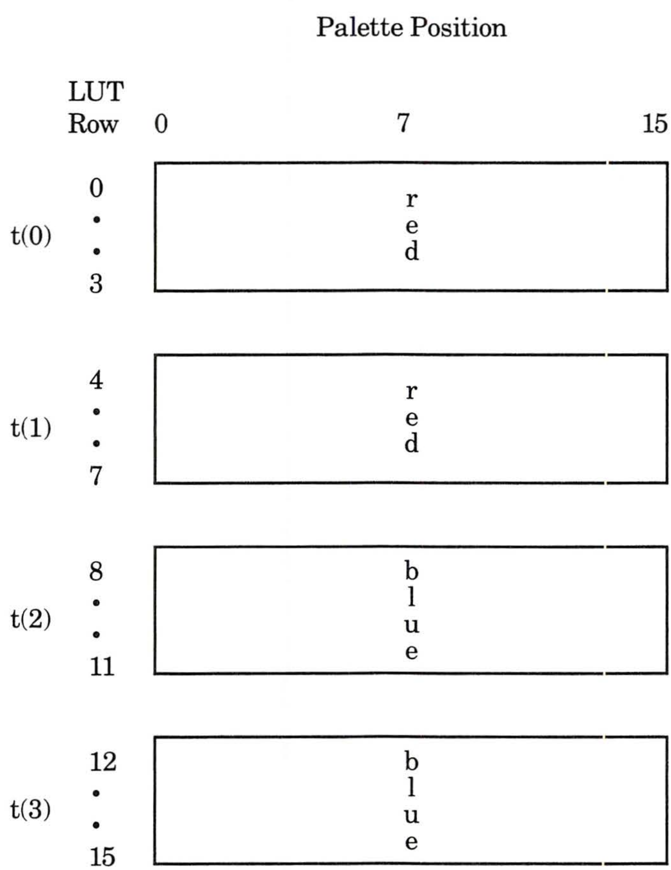

The hardware cycles through the LUT every second, so each quarter of the LUT is active for ¼ of each second. The cycling mechanism produces blinking. The following examples show the details of how you can produce several different blinking effects by setting different values in the LUT.

In this example, the Write Dot or Write Character functions specify palette position 7 and the LUT is set up as shown. Pixels are displayed as a solid red color. In the first ¼ second, the DEB displays the color in the first quarter of the LUT, which in this case is red. In the second, third, and fourth ¼ seconds, the DEB displays the color in the second, third, and fourth quarters of the LUT, respectively. In this example, the DEB keeps finding the color value for red, so what you see on the screen is a solid (non-blinking) red color.



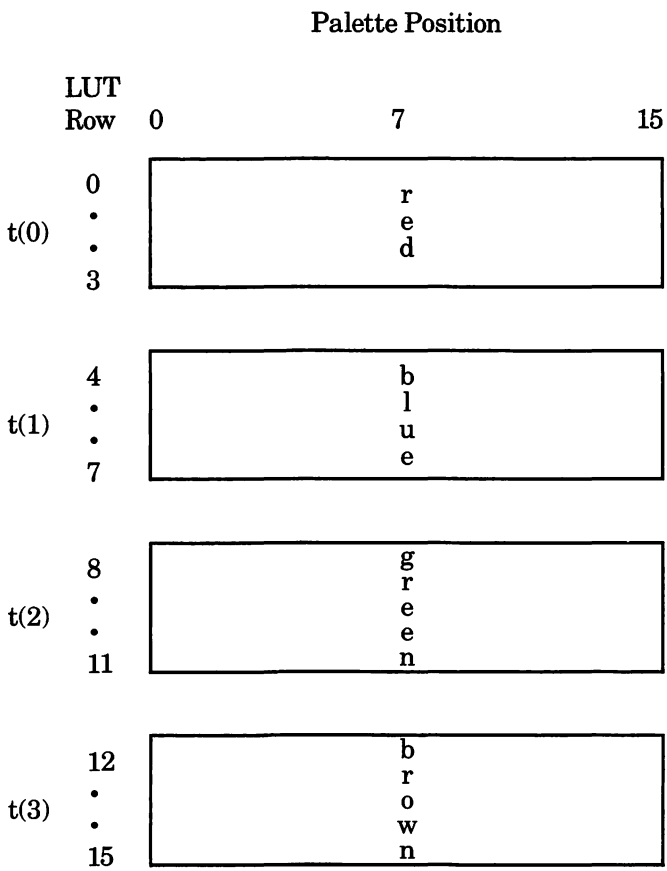

Non-Blinking Color

In this example, any item displayed on the screen with palette position 7 blinks between red and blue. For the first two ¼ seconds, the DEB picks up the color value for red from the first and second quarters of the LUT. For the second two ¼ seconds, the DEB obtains the color value of blue from the LUT. The net effect is a slow blink between red and blue.



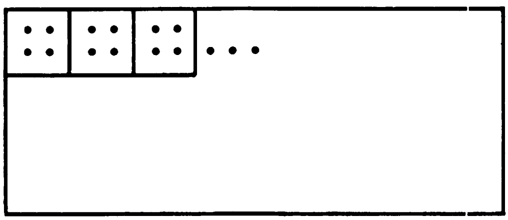

Slow Blink

In this example, any item displayed using palette position 7 blinks rapidly between red, blue, green, and brown.

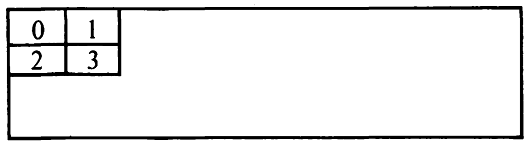


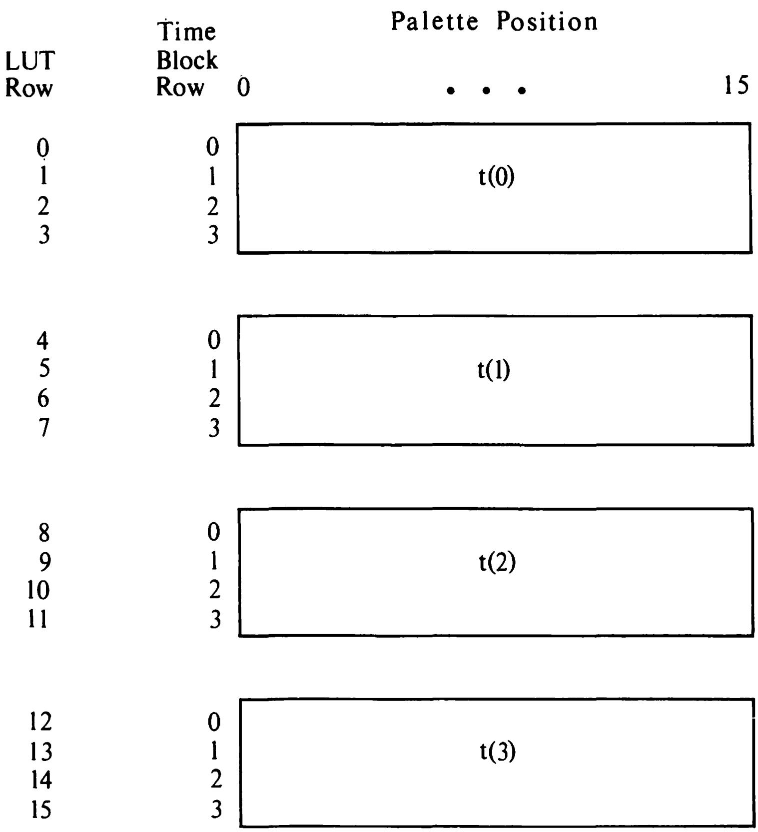

4-Color Fast Blink

For dithering colors, the DEB uses a scheme similar to the blinking scheme. Dithering is accomplished by manipulating groups of 4 adjacent pixels. The screen is divided into blocks of 4 pixels.

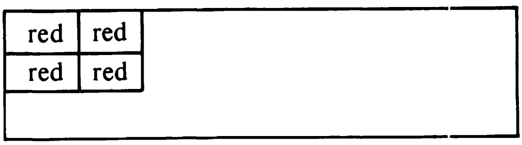


Each of the 4 time states is divided into four dither states that determine the dithering effect. The rows of the time state blocks correspond to the 4-pixel blocks on the screen in the following way:



The pixels in the pixel blocks are so close together that our eyes cannot perceive them as separate. If each of the pixels in a pixel block is a different color, our eyes perceive the pixel block as one color — a combination of the color of the individual pixels. If the adjacent pixels are the same color, our eyes see just that one color.

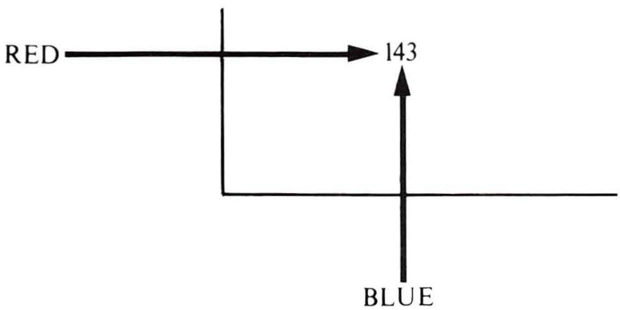



Palette Position

| Time Block | Row | 0      7      15 |
|---|---|---|
| | 0 | red |
| | 1 | red |
| t(0) | 2 | red |
| | 3 | red |
| | 0 | red |
| | 1 | red |
| t(1) | 2 | red |
| | 3 | red |
| | 0 | red |
| | 1 | red |
| t(2) | 2 | red |
| | 3 | red |
| | 0 | red |
| | 1 | red |
| t(3) | 2 | red |
| | 3 | red |

"Solid" Dither showing correspondence between pixel positions in a pixel block and time state rows

Remember the table of "pre-assigned" dithered colors. To combine colors, you check the table for the color number for a particular dither effect. For example, you would choose this number to produce a dither between red and blue.



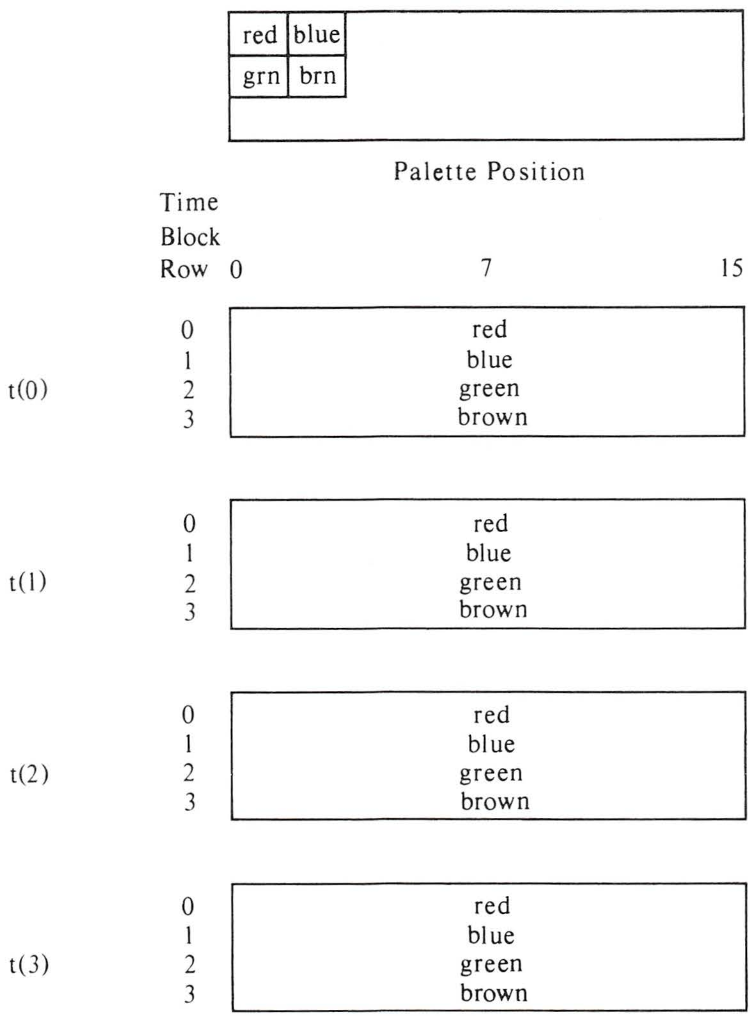
If you want to program the LUT directly to dither red and blue together, the LUT would look like this:

| | | | |
|---|---|---|---|
| blue | red | blue | red |
| blue | red | blue | red |

| Time | Block Row | Palette Position 0 … 7 … 15 |
|---|---|---|
| t(0) | 0 | blue |
| | 1 | red |
| | 2 | blue |
| | 3 | red |
| t(1) | 0 | blue |
| | 1 | red |
| | 2 | blue |
| | 3 | red |
| t(2) | 0 | blue |
| | 1 | red |
| | 2 | blue |
| | 3 | red |
| t(3) | 0 | blue |
| | 1 | red |
| | 2 | blue |
| | 3 | red |

2-Color Dither

You can set up the LUT to dither two, three, or four colors together.




4-Color Dither

The following examples show the actual LUT values for each of the previous cases of blinking and dithering.

Palette Position

| | LUT Row | 0 | 7 | 15 |
|---|---|---|---|---|
| | 0 | | 4 (red) | |
| | 1 | | 4 | |
| t(0) | 2 | | 4 | |
| | 3 | | 4 | |
| | 4 | | 4 | |
| | 5 | | 4 | |
| t(1) | 6 | | 4 | |
| | 7 | | 4 | |
| | 8 | | 4 | |
| | 9 | | 4 | |
| t(2) | 10 | | 4 | |
| | 11 | | 4 | |
| | 12 | | 4 | |
| | 13 | | 4 | |
| t(3) | 14 | | 4 | |
| | 15 | | 4 | |

Palette Position 7 programmed for Non-Blinking Red

Palette Position

| | LUT Row | 0 | 7 | 15 |
|---|---|---|---|---|
| | 0 | | 4 (red) | |
| | 1 | | 4 | |
| t(0) | 2 | | 4 | |
| | 3 | | 4 | |
| | 4 | | 4 | |
| | 5 | | 4 | |
| t(1) | 6 | | 4 | |
| | 7 | | 4 | |
| | 8 | | 1 (blue) | |
| | 9 | | 1 | |
| t(2) | 10 | | 1 | |
| | 11 | | 1 | |
| | 12 | | 1 | |
| | 13 | | 1 | |
| t(3) | 14 | | 1 | |
| | 15 | | 1 | |

Palette Position 7 programmed to blink slowly between red and blue.

Palette Position

| | LUT Row | 0 | 7 | 15 |
|---|---|---|---|---|
| | 0 | | 4 (red) | |
| | 1 | | 4 | |
| t(0) | 2 | | 4 | |
| | 3 | | 4 | |
| | 4 | | 1 (blue) | |
| | 5 | | 1 | |
| t(1) | 6 | | 1 | |
| | 7 | | 1 | |
| | 8 | | 2 (green) | |
| | 9 | | 2 | |
| t(2) | 10 | | 2 | |
| | 11 | | 2 | |
| | 12 | | 6 (brown) | |
| | 13 | | 6 | |
| t(3) | 14 | | 6 | |
| | 15 | | 6 | |

4-Color Fast Blink

| | LUT Row | Palette Position: 0 | 7 | 15 |
|---|---|---|---|---|
| | 0 | | 4 (red) | |
| | 1 | | 4 | |
| t(0) | 2 | | 4 | |
| | 3 | | 4 | |
| | 4 | | 4 | |
| | 5 | | 4 | |
| t(1) | 6 | | 4 | |
| | 7 | | 4 | |
| | 8 | | 4 | |
| | 9 | | 4 | |
| t(2) | 10 | | 4 | |
| | 11 | | 4 | |
| | 12 | | 4 | |
| | 13 | | 4 | |
| t(3) | 14 | | 4 | |
| | 15 | | 4 | |

Solid Red Dither

| | LUT Row | Palette Position 0 | 7 | 15 |
|---|---|---|---|---|
| | 0 | | 1 (blue) | |
| | 1 | | 4 (red) | |
| t(0) | 2 | | 1 (blue) | |
| | 3 | | 4 (red) | |
| | 4 | | 1 | |
| | 5 | | 4 | |
| t(1) | 6 | | 1 | |
| | 7 | | 4 | |
| | 8 | | 1 | |
| | 9 | | 4 | |
| t(2) | 10 | | 1 | |
| | 11 | | 4 | |
| | 12 | | 1 | |
| | 13 | | 4 | |
| t(3) | 14 | | 1 | |
| | 15 | | 4 | |

2-Color Dither: Red and Blue

Palette Position

| | LUT Row | 0 | 7 | 15 |
|---|---|---|---|---|
| | 0 | | 4 (red) | |
| | 1 | | 2 (green) | |
| t(0) | 2 | | 1 (blue) | |
| | 3 | | 6 (brown) | |
| | 4 | | 4 | |
| | 5 | | 2 | |
| t(1) | 6 | | 1 | |
| | 7 | | 6 | |
| | 8 | | 4 | |
| | 9 | | 2 | |
| t(2) | 10 | | 1 | |
| | 11 | | 6 | |
| | 12 | | 4 | |
| | 13 | | 2 | |
| t(3) | 14 | | 1 | |
| | 15 | | 6 | |

4-Color Dither Between Red, Green, Blue, and Brown

The following is an example that combines blinking and dithering:

Palette Position

| | LUT Row | 0 | 7 | 15 |
|---|---|---|---|---|
| | 0 | | 1 (blue) | |
| | 1 | | 4 (red) | |
| t(0) | 2 | | 1 | |
| | 3 | | 4 | |
| | 4 | | 1 | |
| | 5 | | 4 | |
| t(1) | 6 | | 1 | |
| | 7 | | 4 | |
| | 8 | | 2 (green) | |
| | 9 | | 6 (brown) | |
| t(2) | 10 | | 2 | |
| | 11 | | 6 | |
| | 12 | | 2 | |
| | 13 | | 6 | |
| t(3) | 14 | | 2 | |
| | 15 | | 6 | |

The following table of values can be used to program the LUT for normal 16-color graphics.

Palette Position

| | LUT Row | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 1 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(o) | 2 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 3 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 4 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 5 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(1) | 6 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 7 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 8 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 9 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(2) | 10 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 11 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 12 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 13 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(3) | 14 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 15 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | 7, | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |

Non-Blinking Standard Colors

Note that palette position 7 in the first two time states has been programmed to show white and in the second two time states to show red.

Palette Position

| | LUT Row | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 1 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(o) | 2 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 3 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 4 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 5 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(1) | 6 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 7 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **7,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 8 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 9 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(2) | 10 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 11 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 12 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 13 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| t(3) | 14 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |
| | 15 | 0, | 1, | 2, | 3, | 4, | 5, | 6, | **4,** | 8, | 9, | 10, | 11, | 12, | 13, | 14, | 15, |

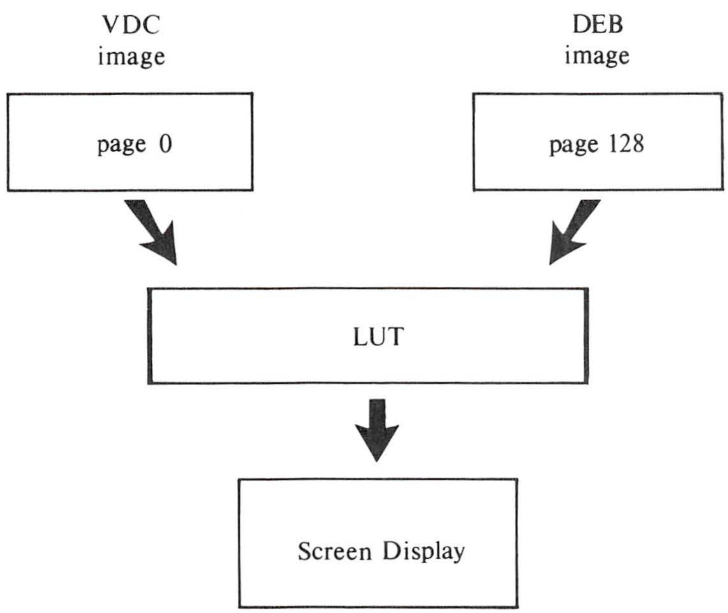
LUT for Blinking Between White and Red in Palette Position 7

# Overlay Modes LUT Programming

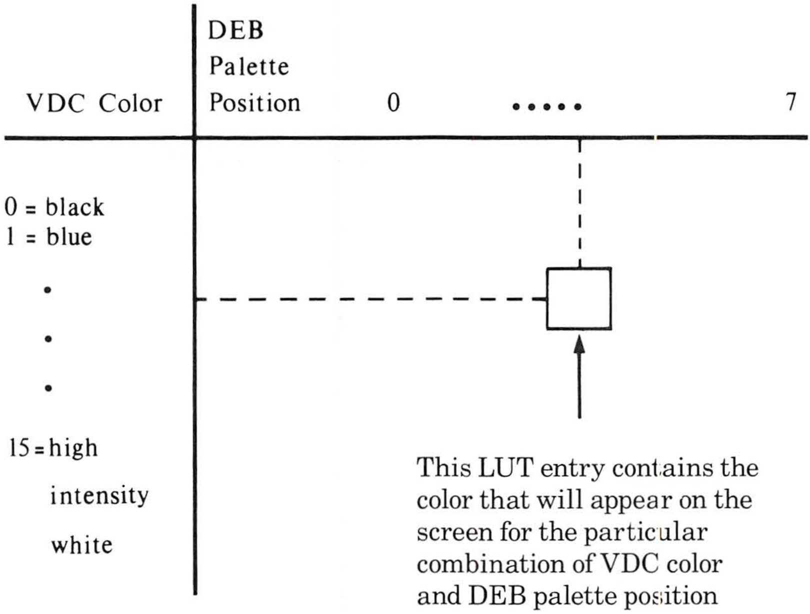
**Overlay Modes LUT Programming**

When the LUT is used in the overlay modes it can be viewed as a two-dimensional array with 8 columns and 32 rows. The column values are DEB palette positions. The row values are VDC color values.

In overlay modes, there are 2 separately controlled images: the VDC image and the DEB image. The 2 images are combined on the display screen. Each pixel on the screen has 2 values associated with it: the VDC color and the DEB palette position. The LUT is used to resolve contention between the 2 values associated with each pixel.



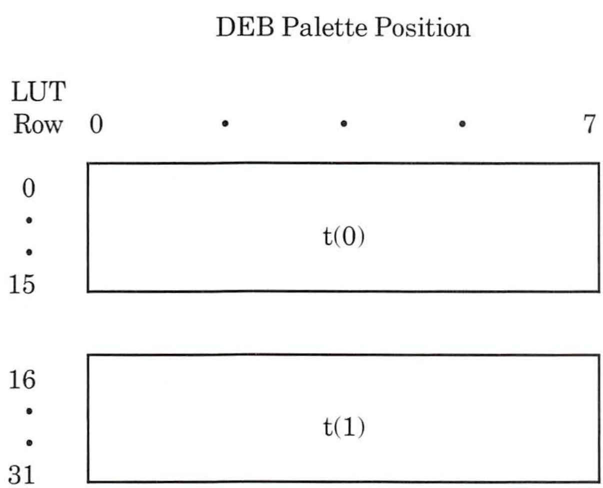
The LUT for overlay modes looks like this:




As in the 16-color graphics modes, the locations in the LUT are numbered consecutively from left to right and top to bottom. For example, location 17 corresponds to Row 2, Palette Position 0.

In the overlay modes, as in the 16-color graphics mode, the LUT is divided into time states that control blinking effects. However, in the overlay modes, the LUT is only divided into two time states. Half of the LUT determines what is being displayed at any time. The top half is used for the first ½ of each second and the bottom half is used for the second ½ of each second.

Using the overlay modes, you create blinking by making the values in the top half of the table different from the corresponding values in the bottom half of the table.



The following example shows the LUT values for standard Palette 2 of an overlay mode. The LUT is programmed so that the DEB image is displayed only if the VDC color is 0 (black). If the VDC requests any other color, then that color is displayed no matter what the DEB requests. This has the effect of overlaying the VDC image "on top" of the DEB image.

DEB Palette Position

| | VDC Color Values | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | **0,** | **1,** | **2,** | **3,** | **4,** | **5,** | **6,** | **7,** |
| | 1 | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, |
| | 2 | 2, | 2, | 2, | 2, | 2, | 2, | 2, | 2, |
| | 3 | 3, | 3, | 3, | 3, | 3, | 3, | 3, | 3, |
| t(0) | 4 | 4, | 4, | 4, | 4, | 4, | 4, | 4, | 4, |
| | 5 | 5, | 5, | 5, | 5, | 5, | 5, | 5, | 5, |
| | 6 | 6, | 6, | 6, | 6, | 6, | 6, | 6, | 6, |
| | 7 | 7, | 7, | 7, | 7, | 7, | 7, | 7, | 7, |
| | 8 | 8, | 8, | 8, | 8, | 8, | 8, | 8, | 8, |
| | 9 | 9, | 9, | 9, | 9, | 9, | 9, | 9, | 9, |
| | 10 | 10, | 10, | 10, | 10, | 10, | 10, | 10, | 10, |
| | 11 | 11, | 11, | 11, | 11, | 11, | 11, | 11, | 11, |
| | 12 | 12, | 12, | 12, | 12, | 12, | 12, | 12, | 12, |
| | 13 | 13, | 13, | 13, | 13, | 13, | 13, | 13, | 13, |
| | 14 | 14, | 14, | 14, | 14, | 14, | 14, | 14, | 14, |
| | 15 | 15, | 15, | 15, | 15, | 15, | 15, | 15, | 15, |

DEB Palette Position

| | VDC Color Values | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| | **0** | **0,** | **1,** | **2,** | **3,** | **4,** | **5,** | **6,** | **7,** |
| | 1 | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, |
| | 2 | 2, | 2, | 2, | 2, | 2, | 2, | 2, | 2, |
| | 3 | 3, | 3, | 3, | 3, | 3, | 3, | 3, | 3, |
| t(1) | 4 | 4, | 4, | 4, | 4, | 4, | 4, | 4, | 4, |
| | 5 | 5, | 5, | 5, | 5, | 5, | 5, | 5, | 5, |
| | 6 | 6, | 6, | 6, | 6, | 6, | 6, | 6, | 6, |
| | 7 | 7, | 7, | 7, | 7, | 7, | 7, | 7, | 7, |
| | 8 | 8, | 8, | 8, | 8, | 8, | 8, | 8, | 8, |
| | 9 | 9, | 9, | 9, | 9, | 9, | 9, | 9, | 9, |
| | 10 | 10, | 10, | 10, | 10, | 10, | 10, | 10, | 10, |
| | 11 | 11, | 11, | 11, | 11, | 11, | 11, | 11, | 11, |
| | 12 | 12, | 12, | 12, | 12, | 12, | 12, | 12, | 12, |
| | 13 | 13, | 13, | 13, | 13, | 13, | 13, | 13, | 13, |
| | 14 | 14, | 14, | 14, | 14, | 14, | 14, | 14, | 14, |
| | 15 | 15, | 15, | 15, | 15, | 15, | 15, | 15, | 15, |

In this example, the standard Palette 2 is modified so that position 2 is a blinking between blue (color 1) and red (color 4).

DEB Palette Position

| | VDC Color Values | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0, | 1, | **1,** | 3, | 4, | 5, | 6, | 7, |
| | 1 | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, |
| | 2 | 2, | 2, | 2, | 2, | 2, | 2, | 2, | 2, |
| | 3 | 3, | 3, | 3, | 3, | 3, | 3, | 3, | 3, |
| t(0) | 4 | 4, | 4, | 4, | 4, | 4, | 4, | 4, | 4, |
| | 5 | 5, | 5, | 5, | 5, | 5, | 5, | 5, | 5, |
| | 6 | 6, | 6, | 6, | 6, | 6, | 6, | 6, | 6, |
| | 7 | 7, | 7, | 7, | 7, | 7, | 7, | 7, | 7, |
| | 8 | 8, | 8, | 8, | 8, | 8, | 8, | 8, | 8, |
| | 9 | 9, | 9, | 9, | 9, | 9, | 9, | 9, | 9, |
| | 10 | 10, | 10, | 10, | 10, | 10, | 10, | 10, | 10, |
| | 11 | 11, | 11, | 11, | 11, | 11, | 11, | 11, | 11, |
| | 12 | 12, | 12, | 12, | 12, | 12, | 12, | 12, | 12, |
| | 13 | 13, | 13, | 13, | 13, | 13, | 13, | 13, | 13, |
| | 14 | 14, | 14, | 14, | 14, | 14, | 14, | 14, | 14, |
| | 15 | 15, | 15, | 15, | 15, | 15, | 15, | 15, | 15, |

DEB Palette Position

| | VDC Color Values | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0, | 1, | 4, | 3, | 4, | 5, | 6, | 7, |
| | 1 | 1, | 1, | 1, | 1, | 1, | 1, | 1, | 1, |
| | 2 | 2, | 2, | 2, | 2, | 2, | 2, | 2, | 2, |
| | 3 | 3, | 3, | 3, | 3, | 3, | 3, | 3, | 3, |
| t(1) | 4 | 4, | 4, | 4, | 4, | 4, | 4, | 4, | 4, |
| | 5 | 5, | 5, | 5, | 5, | 5, | 5, | 5, | 5, |
| | 6 | 6, | 6, | 6, | 6, | 6, | 6, | 6, | 6, |
| | 7 | 7, | 7, | 7, | 7, | 7, | 7, | 7, | 7, |
| | 8 | 8, | 8, | 8, | 8, | 8, | 8, | 8, | 8, |
| | 9 | 9, | 9, | 9, | 9, | 9, | 9, | 9, | 9, |
| | 10 | 10, | 10, | 10, | 10, | 10, | 10, | 10, | 10, |
| | 11 | 11, | 11, | 11, | 11, | 11, | 11, | 11, | 11, |
| | 12 | 12, | 12, | 12, | 12, | 12, | 12, | 12, | 12, |
| | 13 | 13, | 13, | 13, | 13, | 13, | 13, | 13, | 13, |
| | 14 | 14, | 14, | 14, | 14, | 14, | 14, | 14, | 14, |
| | 15 | 15, | 15, | 15, | 15, | 15, | 15, | 15, | 15, |

In this example, values in the LUT cause the DEB's output to take precedence over the VDC's output. The VDC's output is only displayed when you specify DEB palette position 0 in a graphics statement.

| | VDC Color Values | DEB Palette Positions 0 1 2 3 4 5 6 7 |
|---|---|---|
| | 0 | **0,** 1, 2, 3, 4, 5, 6, 7, |
| | 1 | **1,** 1, 2, 3, 4, 5, 6, 7, |
| | 2 | **2,** 1, 2, 3, 4, 5, 6, 7, |
| | 3 | **3,** 1, 2, 3, 4, 5, 6, 7, |
| | 4 | **4,** 1, 2, 3, 4, 5, 6, 7, |
| | 5 | **5,** 1, 2, 3, 4, 5, 6, 7, |
| | 6 | **6,** 1, 2, 3, 4, 5, 6, 7, |
| t(0) | 7 | **7,** 1, 2, 3, 4, 5, 6, 7, |
| | 8 | **8,** 1, 2, 3, 4, 5, 6, 7, |
| | 9 | **9,** 1, 2, 3, 4, 5, 6, 7, |
| | 10 | **10,** 1, 2, 3, 4, 5, 6, 7, |
| | 11 | **11,** 1, 2, 3, 4, 5, 6, 7, |
| | 12 | **12,** 1, 2, 3, 4, 5, 6, 7, |
| | 13 | **13,** 1, 2, 3, 4, 5, 6, 7, |
| | 14 | **14,** 1, 2, 3, 4, 5, 6, 7, |
| | 15 | **15,** 1, 2, 3, 4, 5, 6, 7, |

DEB Palette Positions

| | VDC Color Values | 0 1 2 3 4 5 6 7 |
|---|---|---|
| | 1 | **0,** 1, 2, 3, 4, 5, 6, 7, |
| | 1 | **0,** 1, 2, 3, 4, 5, 6, 7, |
| | 2 | **2,** 1, 2, 3, 4, 5, 6, 7, |
| | 3 | **3,** 1, 2, 3, 4, 5, 6, 7, |
| | 4 | **4,** 1, 2, 3, 4, 5, 6, 7, |
| t(1) | 5 | **5,** 1, 2, 3, 4, 5, 6, 7, |
| | 6 | **6,** 1, 2, 3, 4, 5, 6, 7, |
| | 7 | **7,** 1, 2, 3, 4, 5, 6, 7, |
| | 8 | **8,** 1, 2, 3, 4, 5, 6, 7, |
| | 9 | **9,** 1, 2, 3, 4, 5, 6, 7, |
| | 10 | **10,** 1, 2, 3, 4, 5, 6, 7, |
| | 11 | **11,** 1, 2, 3, 4, 5, 6, 7, |
| | 12 | **12,** 1, 2, 3, 4, 5, 6, 7, |
| | 13 | **13,** 1, 2, 3, 4, 5, 6, 7, |
| | 14 | **14,** 1, 2, 3, 4, 5, 6, 7, |
| | 15 | **15,** 1, 2, 3, 4, 5, 6, 7, |

The following LUT entirely blocks out VDC output:

| | VDC Color Values | DEB Palette Positions 0 1 2 3 4 5 6 7 |
|---|---|---|
| | 0 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 1 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 2 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 3 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 4 | 0, 1, 2, 3, 4, 5, 6, 7, |
| t(0) | 5 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 6 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 7 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 8 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 9 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 10 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 11 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 12 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 13 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 14 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 15 | 0, 1, 2, 3, 4, 5, 6, 7, |

DEB Palette Positions

| | VDC Color Values | 0 1 2 3 4 5 6 7 |
|---|---|---|
| | 0 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 1 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 2 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 3 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 4 | 0, 1, 2, 3, 4, 5, 6, 7, |
| t(1) | 5 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 6 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 7 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 8 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 9 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 10 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 11 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 12 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 13 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 14 | 0, 1, 2, 3, 4, 5, 6, 7, |
| | 15 | 0, 1, 2, 3, 4, 5, 6, 7, |

# Programming the Bit Planes

**Introduction**

Once you have learned to program the LUT directly using the Set Color Palette command, you can make further use of the LUT's capabilities by programming the VDC and DEB video memory directly.

By directly programming the video memory of the VDC and DEB boards, you can increase the graphics display speed. The values you load into the video memory planes determine how the LUT is accessed. This section assumes that you have read and understood how to program the LUT directly.

In the 16-color graphics modes, the device driver combines the 3 bit planes of the DEB with one bit plane from the VDC to create the four bit planes necessary for 16-color graphics.

In the overlay modes, the device driver uses the 3 DEB bit planes for 8-color graphics output and uses the VDC board separately for either text or graphics output.

| | |
|---|---|
| A0000:0 | DEB Bitplane 0 |
| A8000:0 | DEB Bitplane 1 |
| B000:0 | DEB Bitplane 2 |
| B8000:0 | VDC Bitplane |

Memory Map

## LUT Addressing

A LUT address is an 8-bit value that points to one of the 256 locations within the LUT. The method of address formation depends on the current video mode.

For transparent and disabled modes, LUT addressing is irrelevant. In the transparent mode, VDC color values bypass the LUT processing and go directly to the monitor output. In the disabled mode, all output from the LUT is forced to the value of zero.

For the 16-color graphics and overlay modes, the LUT address is composed of bits from the DEB video bit planes, the VDC's video output, and DEB timing bits.

## Timing Bits

The timing bits are called BLINK1, BLINK2, PAT1, and PAT2. BLINK1 and BLINK2 effect blinking; PAT1 and PAT2 effect patterning (dithering).

All of the timing bits are applicable in the 16-color graphics mode; only BLINK1 is part of the address formation in the overlay mode. Therefore, you have fewer options for blinking and no ability to dither in the overlay mode.

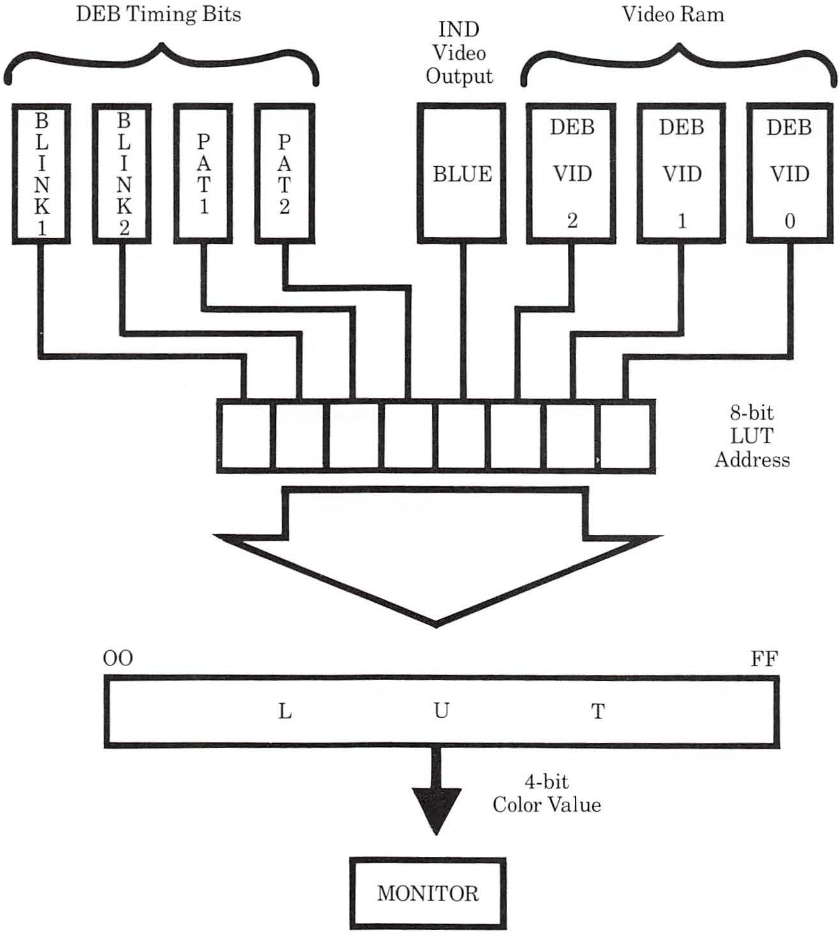
The operation of the timing bits is very fundamental to creation of special effects. The bits always cycle on and off, each at a different rate. BLINK1 cycles on and off each 1/4 second. BLINK2 cycles on and off each 1/8 second.

PAT1 and PAT2 cycle on and off so fast that the eye cannot perceive a blink (PAT1 is the fastest). A dithered color is really 2-4 separate colors that are changing so rapidly that the eye perceives them as one solid color.

PAT1 changes value at the same rate that the monitor's cathode ray moves from one pixel to the next. PAT1's effect on LUT addressing is that it switches the address by 16 LUT entries — in the previous table, between pairs of rows. PAT2 changes value at the same rate that the cathode ray changes scanlines — in the previous table, between one pair of rows and the next pair of rows.

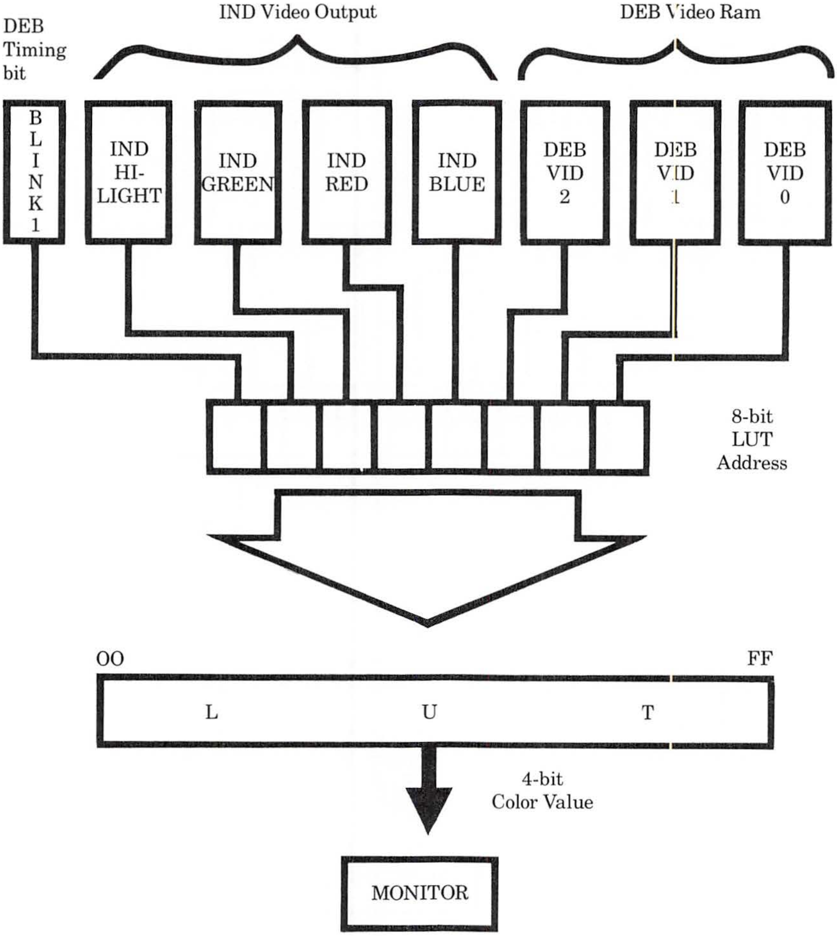
| PAT2 | PAT1 | Portion of LUT |
|---|---|---|
| 0 | 0 | 1st 16 entries of each quarter |
| 0 | 1 | 2nd 16 entries of each quarter |
| 1 | 0 | 3rd 16 entries of each quarter |
| 1 | 1 | 4th 16 entries of each quarter |

1. 16-color Graphics Mode




To output a color to the monitor, the DEB concatenates the DEB timing bits BLINK1, BLINK2, PAT1, PAT2, the BLUE output bit from the VDC, and a bit from corresponding locations on each of the three DEB bit planes.

2. Overlay Mode

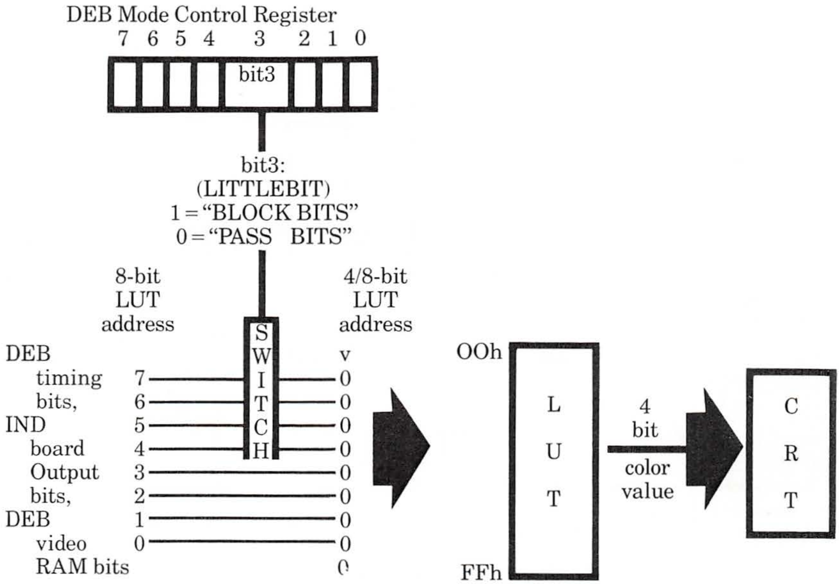



To output a color to the monitor, the DEB concatenates the following bits: BLINK1, the HILIGHT, GREEN, RED, and BLUE output bits from the VDC, and a bit from corresponding locations on each of the three DEB bit planes.

**Short LUT Addresses**

The DEB supports a method for you to access only the first sixteen LUT locations. This lets you use normal 16-color graphics without needing to manage all of the 256 LUT locations. You invoke this short addressing mode by a setting bit 2 in AL in the "Set Color Palette" command.

3. Short LUT Addresses



4. Modes, Address Formation, & DEB Mode Control Register