

A GUIDE TO OMNI

JANUARY 13, 1984

This document and project is dedicated to Donald Teiser, without whose judgement, guidance and protection this effort would have resulted in nothing, whatsoever.

I extend thanks to the following people for their help:

Penny Burton, my wife, who lent her name and support and who forgave the pleasures of marriage on many occasions for this endeavor,

Vivian Filipak, my mother, who lent her name and her son to this,

Arlen Olive who provided technical assistance and lent his daughter's name, Heather,

Akio Tanaka and Eric Breeze for their good work,

John Seghers for the MAPping algorithm and

President Ronald Raygun without whom America would have surely fallen to the heathen communists.

TO THE READER

I apologize for any omissions I have made or any confusion resulting from my terminology, explanations or organization. If you are reading this document for the first time, my heart goes out to you. Good luck!

Believe me, it was as hard to write as it is to read. I trust that

by your tenth reading, most of the features of OMNI will come clear.

If you have any suggestions or criticisms regarding this document or the product, you will find me an enthusiastic listener.

Mark Filipak

OMNI, A THREE DIMENSIONAL VIDEO GRAPHICS SYSTEM

TABLE OF CONTENTS:

2	SYSTEM FEATURES	
5	SYSTEM LIMITATIONS/DRAWBACKS	
	SYSTEM DESCRIPTION	5
7	SYSTEM MEMORY MAP	
8	SYSTEM BLOCK DIAGRAM	
10	SPRITE DEFINED	
9	SPRITE POSITIONING SPACE	
10	PERCEPT DEFINED	
11	FUNCTIONAL PRIORITIZATION	
12	GRAPHICS MEMORY MAP	
	SPRITE PARAMETER DEFINITIONS	
13	S	
13	E	
13	I & R	
13	HGT	
13	CHA	
14	MAP	
14	XOFF, YOFF & ZOFF	
14	XPOS, YPOS & ZPOS	
15	FORMAT	
15	FORMAT SELECTION GUIDE	
16	DAZZLER SPRITE	
17	MEDIUM CARTOON SPRITE	
18	LARGE CARTOON SPRITE	

19	DETAIL SPRITE
20	DETAIL REVERSE SPRITE
21	MEDIUM SHADE SPRITE
22	MEDIUM REVERSE SPRITE
23	LARGE SHADE SPRITE
24	LARGE REVERSE SPRITE
25	AIR BRUSH SPRITE
26	COLOR-OR SPRITE
27	TEXTURE SPRITE
28	EDGE ENHANCEMENT SPRITE
	PROGRAMMING THE COLOR PALETTE
28	PALETTE MEMORY MAP
28	TWO METHODS OF GENERATING COLORS
	APPENDIX
A-1	WHAT IS STENCILING ?
A-2	WHAT IS TILING ?
A-3	NTSC COLOR CHART
A-4	FIRST QUADRANT
A-5	SECOND QUADRANT
A-6	THIRD QUADRANT
A-7	FOURTH QUADRANT
A-8	AN EXAMPLE OF '15-SWITCH' COLOR
A-9	AN EXAMPLE OF '15-SWITCH' BRIGHTNESS
A-10	AN EXAMPLE OF '15-SWITCH' INTENSITY
A-11	AN EXAMPLE OF COMBINED '15-SWITCH' BRIGHTNESS & INTENSITY
	AN EXAMPLE OF '15-SWITCH' COLOR, BRIGHTNESS & INTENSITY

SYSTEM FEATURES

THE DISPLAY

High resolution 648 pixel/line by 488 line screen (NTSC) where
 one pixel is equal to 0.03" on a 25" television,
 Square pixels,

Single pixel position resolution resulting in 648 positions
 across the visible portion of the screen (for a 25" television, this
 translates to .03" per increment of position),

Two pixel resolution in color resulting in 324 color changes
 across the visible portion of the screen,

Single pixel resolution in intensity resulting in 648 intensity
 changes across the visible portion of the screen,

THE PROGRAMMING ENVIRONMENT

True X,Y,Z three dimensional coordinate system allowing the
 program to 'view' the space and manipulate objects in true 'third person'
 perspective,

256 levels of depth into the screen (Z-coordinate),

Automatic display prioritization to generate the 'first person'
 view of the three dimensional space for presentation on the TV,

The 648x488x256 display is within a 2048x1024x512 virtual space
 to simplify scrolling along and movement in X, Y & Z,

THE OBJECTS

True sprite type graphics objects defined by three-dimensional
 position (X,Y,Z) and object height,

Display priority can be changed by merely moving an object in Z
 with a single CPU store as opposed to a fixed priority (which means that
 all

objects to be reshuffled by the program) or link list priority
(which means that the link list must be maintained by the program),

Up to 18,432 independent (visible) sprites (49,152 virtual
sprites ready for scrolling into the visible screen) which can be used as
either motion sprites or playfield sprites without
differentiation on
the hardware level allowing for maximum flexibility in
programming,

Two classes of sprites (color & intensity),

Nine types of sprites with the data densities and bandwidth for
each
optimized for broadcast television systems (NTSC, PAL & SECAM),

Pixel transparency control for all sprite types,
 Sprites can be grouped together to form large 3D objects which
 can then be repositioned with only three CPU stores,
 Sprites can be laminated one upon another to add detailed
 sections to otherwise low detail areas,
 Playfield sprites can easily be used to create a 3D playfield
 with up to 256 levels of foreground/background objects,
 Sprites are generated and regenerated without CPU involvement,
 without matrix transforms and without peripheral math packs,
 Anti-aliasing designed into objects by the graphic artist in a
 stright forward, easily understood and predictable manner,
 Eighty character text with a dynamically redefinable character
 set,

THE OUTPUT

Programmable pixel clock to bring text into registration with the
 shadow mask to produce optimal characters on most televisions,
 Programmable palette gives the graphics designer full control
 over all aspects of pixel color (hue, degree of saturation and shade),
 2794 hue/saturation/shade combinations in the palette (ie, 203
 hues with an average of 1.6 degrees of saturation each and an average
 of 8-1/2 shades each) for use by color sprites plus an additional 16
 intensity levels per pixel for use by intensity sprites for a
 total of 44,704 hue/saturation/shade/intensity combinations,

	NUMBER OF HUES (% COLOR OF TOTAL)	NUMBER OF HUE/SAT COMBINATIONS	NUMBER OF HUE/SAT/ SHADE COMBINATIONS (% OF TOTAL)
grey	1 (0.5)	1	24 (0.9)
red	42 (20.7)	78	689 (24.7)
yellow	38 (18.7)	59	524 (18.7)

green	47 (23.2)	72	595 (21.3)
cyan	26 (12.8)	45	390 (13.9)
blue	26 (12.8)	41	323 (11.6)
magenta	23 (11.3)	30	249 (8.9)
	-----	---	-----
	203 (100.0)	326	2794 (100.0)

Enhanced color resolution from 90 degrees to 270 degrees of the chrominance phase spectrum so that twice as many reds, yellows, greens, and luminance levels can be created than would otherwise be possible,

Fully interlaced repeat field displays for compatability with videodisc and other electronic media,

The composite video is generated synthetically and in baseband so that the signal is ready to be injected into the channel 2/3 modulator without color sub-carrier quadrature modulators, ratioing circuits or color phase delay lines thereby reducing parts count and cost, component complexity, quality assurance overhead, frequency alignment overhead, failure rates & color drift between samples and over time,

Automatically detects the presence of an external video input (ie, videodisc, etc.) synchronizes to its signal and displays it as background,

A single system clock frequency adjustment at the end of the assembly line simultaneously aligns the color burst frequency, the color phase circuitry, the color sub-carrier frequency & the scan, line and field counters (in essence, everything except the channel 2/3 modulator and the audio sub-carrier),

Four outputs available:

- baseband NTSC (or PAL or SECAM),
- modulated NTSC (or PAL or SECAM),
- RGB and
- R-Y,B-Y,Y (or U,V,Y),

GENERAL FEATURES

Distributed processing system architecture with the graphics subsystem separate from the main system allowing the CPU to run at full speed without wait states or halts,

The use of separate sprite types for color and intensity results in a 50 percent increase in memory utilization allowing the use of slow and inexpensive graphics memory,

The system is modular and expandable,

The system is generalized so that it can be used in a wide spectrum of products,

Custom chips utilize hardwired logic (not microcoded) allowing relatively low clock rate permitting larger chip geometry resulting in increased yield and

Custom graphics chip designed using standard cell technology with spares on chip which can be used as needed to further increase yield.

SYSTEM LIMITATIONS/DRAWBACKS

Rotations must be accomplished by the CPU (by rotating the graphics),

Zooming (or shinking) of sprites moving toward (or away from) the screen must be accomplished by the CPU (by zooming or shrinking the graphics),

True perspective positioning must be done by the CPU and

The hardwiring of logic in the custom chips (as opposed to micro-coded logic) will make any modifications to these chips difficult, time consuming and costly.

SYSTEM DESCRIPTION

THE MAIN SYSTEM

1, a sixteen bit CPU.

2, 966,656 words of system memory space consisting of:

16K words of Operating System ROM for system operation, interrupt processing, input/output management (contoller routines, sound routines, videodisc handlers, playcable loaders, etc.), graphics routines & software signiture,

16K words of Operating System EEPROM (electrically eraseable programmable read only memory) for game parameter store which can remember high scores, skill level, game progress, etc. so that games can be resumed after power has been turned off for periods of up to ten years,

16K words of memory mapped I/O,

885K words of mixed media/system RAM (media can be ROM as in our present products or media subsystem consisting of videodisc reader, playcable loader, etc.) and

52K words of biphased video subsystem ram (see Video Subsystem description on the next page) which can be read or written to
by the CPU at any time;

3, 15,810,560 words of spare addressing space for expansion and

4, VIVIAN, a custom chip for memory management and DMA functions
(it can be dynamically configured using micro-code stores from the
CPU to allow one basic architecture to handle a broad mix of
memory and I/O configurations and speeds).

THE VIDEO SUBSYSTEM

1, 48K words of graphics RAM containing sprite graphics and
 param- meters from the CPU;

2, from one to three PENNYs, a custom chip, each of which
 performs the following functions:

simultaneous generation of 32 sprites using the parameters and
 data stored in graphics RAM,

reuse of each sprite generator up to 82 times per screen
 (assuming all are single line sprites),

programmable grouping of sprites to form larger pseudo 3D
 objects in X,Y,Z,

automatic visual prioritization (in Z) for all sprites within
 each chip with transparency pixels allowing any sprites 'behind' to
 show through,

support of a virtual space over eight times larger than
 display space to ease program maintenance of sprite positions,

support of a display space larger than the actual screen to
 simplify simultaneous X,Y scrolling,

output of a four bit color (index) per pixel on the fly, with
 a color index of zero designating transparency, for use by the
 palette RAM and

output of a four bit intensity per pixel on the fly, with a
 intensity of zero designating full stored intensity for use by
 the HEATHER chip;

3, 4K words of color palette RAM containing chrominance and
 luminance selection data from the CPU to the HEATHER chip with an inter-
 mediate pixel by pixel lookup supplied on the address lines
 from the PENNY chips color outputs and

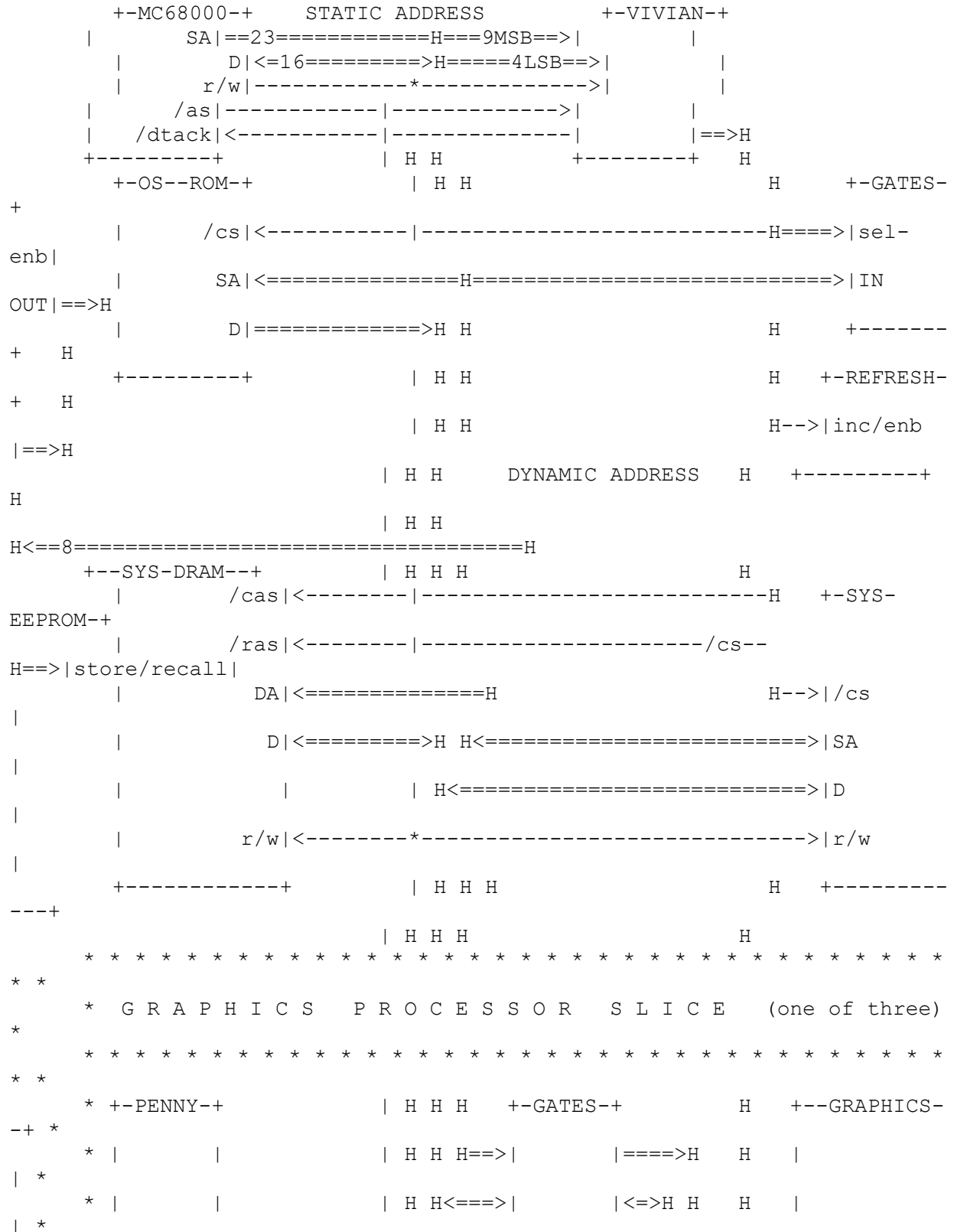
4, HEATHER, a custom chip, which performs the following functions:

Generates the baseband composite video signal,

Synchronizes to and displays an external video signal (when all sprites are showing transparent) and

Generates the various system clocks.

SYSTEM BLOCK DIAGRAM



```

* |      r/w|<-----*----->|dir      |  H H  H  |
| *
* |      ctl|-----|----->|enb      |  H H  H  |
| *
* |      |      | H H H  +-----+  H H  H  |
| *
* |      GA|=====>H=====>|GA
| *
* |      GD|<=====H<=====>|GD
| *
* |  gr/gw|-----|----->|r/w
| *
* |  /gcas|-----|----->|/cas
| *
* |  /sel|<-----|----->*----->|/ras
| *
* |  COLOR|====4==>H      | H H H      |      H  |
| *
* |  INTEN|==4==>H H      | H H H      |      +- /cs--H  +-----
----+ *
* +-----+      H H      | H H H      |      H
*
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* *
                                H H      | H H      +--MUX--+      H
                                H H      | H H==12==>|  2x13  |      H      +--
PALETTE--+
        external  H H      *----->|      |=====>|A
|
        video    H H=====12==>|      |----->|r/w
|
        |      H      | H      +5V-->|  sel|<--*--H      |
|
        crystal |      H      | H      +-----+      |      |
|
        |      |      H      | H      +-GATES--+      |      |
|
        v      v      H      +----->|dir enb|<--+      |
|
+-HEATHER--+      H      H<=====>|      |<==>H<==>|D
|
|      INTEN|<==12=H      +-----+      H      +-----
-+
|      COLOR|<==16=====H
|      videout|-----> COMPOSITE
VIDEO
|      CLOCKS|=====> SYSTEM CLOCKS
+-----+

```

SYSTEM MEMORY MAP

```

FFFFFFFF: : .....:
          : UNASSIGNED :
          :           :
          :           :
          : 15.7M TOTAL :
          :           :
          :           :
0F0000: : _____:
        | UNUSED      | S
        |             | G
0ED000: : _____! U
0EC000: : 4K_PALETTE  ! R
        | 16K PENNY 2 | B
        | GRAPHICS RAM | A
        |             | S
0E8000: : _____! P
        | 16K PENNY 1 | Y
        | GRAPHICS RAM | H
        |             | S
0E4000: : _____! I
        | 16K PENNY 0 | T
        | GRAPHICS RAM | C
        |             | E
0E0000: : _____! S
        : SYSTEM RAM : M
        :   |       :
        :   v       :
        :           :
        : 869K TOTAL :
        :           :
        :   ^       :
        :   |       :
        : SYSTEM MEDIA :
00C000: : .....:
        : 16K I/O MAP :
        :           :
        :           :
        :           : WRITE ONLY
008000: : .....:
        : 16K EEPROM : ' : VIVIAN REGS :
        :           : ' :
        :           : ' :
        :           : ' :
004000: : .....:
        : 16K BASEPAGE : ' :
        :           : ' :
        :           : ' :
        :           : ' : READ ONLY
000000: : .....:
        : OS BOOT ROM : Power-up routines
        :           : System configuration
        :           : Interrupt vectors
        :           : NMI routine

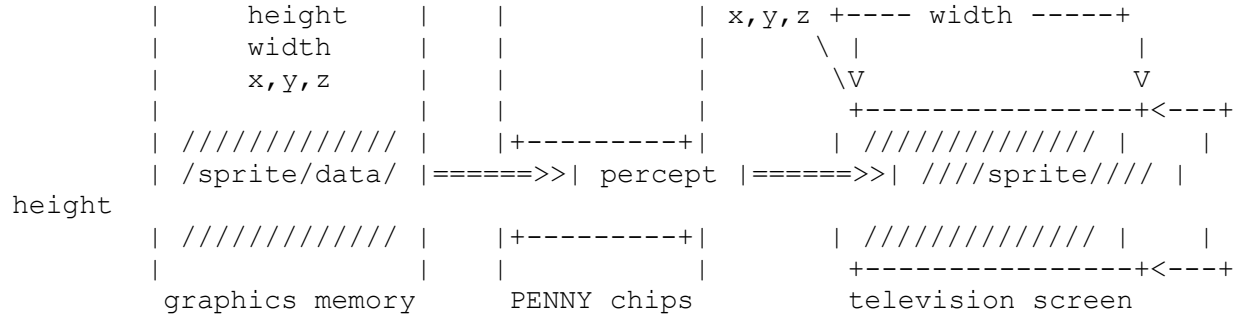
```


SPRITE DEFINED:

A sprite is an object which is generated, and positioned on the television screen without requiring the cpu to handle its data.

It can

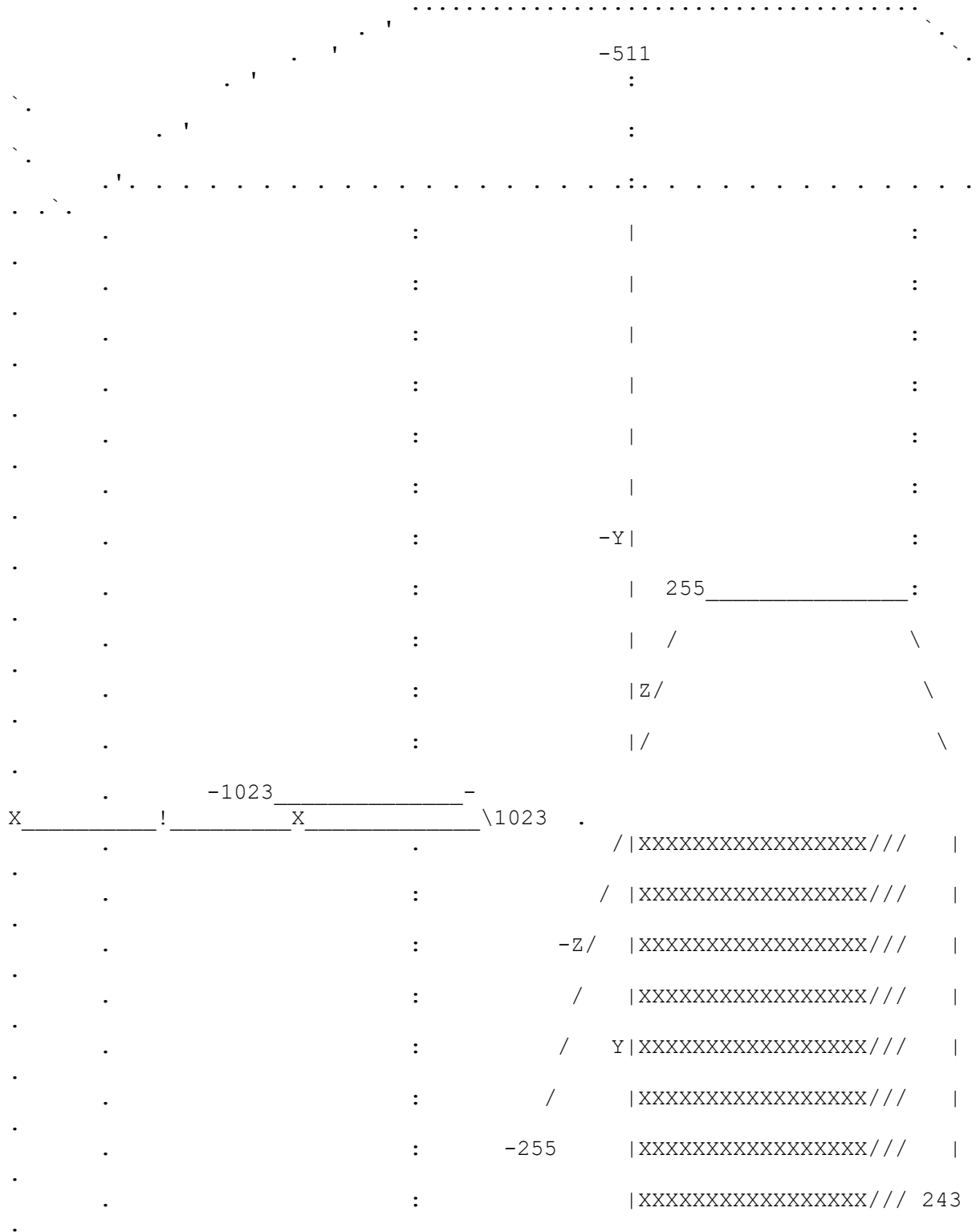
consist of mixed transparent/non-transparent pixels. There is no restriction on pixel transparency.



PERCEPT DEFINED:

A percept is a hardware sprite generator which can be used to generate one or more sprite incarnations. There are 32 percepts per PENNY numbered from 0 to 31. Each percept has an initial link. The initial link points to the attribute list for the first sprite incarnation. Each sprite thereafter contains a link to the next sprite incarnation for that percept in link list fashion. Sprites cannot be reincarnated on the same line and the link list must proceed in the order of television scanning (top to bottom) without overlap between sprites from the same percept.

SPRITE POSITIONING SPACE



GENERAL SPRITE USAGE

```

+-----+
| P E N N Y 2 |
+-----+
| | P E N N Y 1 | + |
+-----+
+---| |---+===== 5 =====>| P E N N Y 0 | +
|+ |
| | +-----+ | |== DIFFERENT ==>| | +
|+ |
| | |stamped| |===== FORMATS ===>| +-PERCEPT-00-+ | +
|+ |
+---| |color |=====>| +-PERCEPT-01-+ | +
|+ |
+-|sprite |---+ | +-PERCEPT-02-+ | +
|+ |
+-----+ | +-PERCEPT-03-+ | + | + |
ARBITRARILY ADDRESSED | +-PERCEPT-04-+ | + | + |
ARBITRARILY SIZED | +-PERCEPT-05-+ | + | + |
CONTIGUOUS STAMP | +-PERCEPT-06-+ | +
|+ |
+-----+ . | +-PERCEPT-07-+ | + | + |
|+-----+ . | +-PERCEPT-08-+
|+ | + |
+-----+ | . | +-PERCEPT-09-+ | +
|+ |
| | |===== 8 =====>| +-PERCEPT-10-+ | + | + |
| +-----+ | |=== DIFFERENT ==>| +-PERCEPT-11-+ | +
|+ |
| |stamped | |===== FORMATS ===>| +-PERCEPT-12-+ | +
|+ |
| |intensity|=====>| +-PERCEPT-13-+ | +
|+ |
+-|sprite |---+ | | +-PERCEPT-14-+ | +
|+ |
+-----+-----+ | +-PERCEPT-15-+ | + | + |
ARBITRARILY ADDRESSED | +-PERCEPT-16-+ | + | + |
ARBITRARILY SIZED | +-PERCEPT-17-+ | +
|+ |
CONTIGUOUS STAMP | +-PERCEPT-18-+ | + | + |
. _____ .- . _____ . I . | +-PERCEPT-19-+ | +
|+ |
| . _____ | | _____ | ____ . N . | +-PERCEPT-20-+ | +
|+ |
. ____ | _ | _____ v _ | ____ . | D . PARALLEL | +-PERCEPT-21-+
|+ | + | | | | |
| wrap : |==== E == PLANE ==>| +-PERCEPT-22-+ | + | + |
| around : | | P M OR | +-PERCEPT-23-+ | + | + |
| color : | | = E A SERIAL =>| +-PERCEPT-24-+ | + | + |
.-->| bit maps+-----+ |<-- . N P PIXEL | +-PERCEPT-25-+ | +
|+ |
! ____ | ..... | mapped | ..... | ____ ! D S DATA | +-PERCEPT-26-+ | +
|+ |

```

```

|      |color |===== E == PER =====>| +-PERCEPT-27-+ |+
|+ |
|      |sprite|      |__! N MAP      | +-PERCEPT-27-+ |+
|| |
|      +-----+      | T      | +-PERCEPT-29-+ |+
|| | | |
|      !_____:_____|      | +-PERCEPT-30-+ ||
|| |
STORE BY ROWS ^ | ARBITRARILY ADDRESSED      | +-PERCEPT-31-+ ||
|| |
OR COLUMNS | | PROGRAMMABLY SIZED      | | position | ||
|| |
INTERCHANGE R&C `-' CONTIGUOUS MAP      | | x y & z | ||
|| |
._____.-.-._____. I .      | | offset | ||
|| | | | | | | |
|      | | |_____. N .      | | x y & z | ||
|| |
._|_____.v_|_____|_____. | D . PARALLEL      | | DATA ADR |
|| || | | | | | |
| wrap :      |===== E=== PLANE ==>| | FORMATS: | || || |
| around :      | | P M OR| | data | || |+ |
| intensity :      |===== E A SERIAL =>| | BIT MAP | ||
|---+
.-->| bit maps+-----+ |<--. N P PIXEL | | display | |+ |
!____|.....|mapped|.....|____! D S DATA | | LINK ADR | |---+
|      |intens|===== E == PER =====>| +-----+ |
|      |sprite|      | | N MAP      +-----+
|      +-----+      |__! T
|      !_____:_____|      In CAPS are
parameters
STORE BY ROWS ^ | ARBITRARILY ADDRESSED      used in sprite fetch.
OR COLUMNS | | PROGRAMMABLY SIZED
INTERCHANGE R&C `-' CONTIGUOUS MAP

```

```

+-----+      ===== SYMBOLIC COLOR SPRITE FORMATS =====
| P E N N Y 2 |
+-----+      @@@#....$$//&&*      DAZZLER
| P E N N Y 1 | + |=====>
+-----+      + |=====>      8 fixed-len elements, 2
pix/element,
| P E N N Y 0 | + |=====>==>      4 bits of color/element,
|               + |=====>==>      4 bits of overall brightness
| +-PERCEPT-00-+ |=====>==>==>
| +-PERCEPT-01-+ |=====>==>==>      @@@@#@###..../////      MED
CARTOON
| +-PERCEPT-02-+ |=====>==>==>
| +-PERCEPT-03-+ |=====>==>==>      4 run-len elements, 2*(1-16)
pix/run,
| +-PERCEPT-04-+ |=====>==>==>      4 bits of color/element,
| +-PERCEPT-05-+ |=====>==>==>      4 bits of overall brightness,
automatic
| +-PERCEPT-06-+ |=====>==>==>      edge enhancement on color change.
| +-PERCEPT-07-+ |=====>==>==>
| +-PERCEPT-08-+ |=====>==>==>      @@@@@@@@@@.....#####      LRG
CARTOON
| +-PERCEPT-09-+ |=====>==>==>
| +-PERCEPT-10-+ |=====>==>==>      4 run-len elements, 32*(1-16)
pix/run,
| +-PERCEPT-11-+ |=====>==>==>      4 bits of color/element,
| +-PERCEPT-12-+ |=====>==>==>      4 bits of overall brightness,
automatic
| +-PERCEPT-13-+ | = 96 >==>==>      edge enhancement on color change.
| +-PERCEPT-14-+ | = SPRITES =>
| +-PERCEPT-15-+ | = PER LINE >      //...//...//...//...//...//...//      AIR
BRUSH
| +-PERCEPT-16-+ |=====>==>==>      %..%..%..%..%..%..%..%..%..%..%..%
COLOR-OR
| +-PERCEPT-17-+ |=====>==>==>
| +-PERCEPT-18-+ |=====>==>==>      32 fixed-len elements, 2
pix/element,
| +-PERCEPT-19-+ |=====>==>==>      4 bits of overall color
| +-PERCEPT-20-+ |=====>==>==>
| +-PERCEPT-21-+ |=====>==>==>      == SYMBOLIC INTENSITY SPRITE
FORMATS ==
| +-PERCEPT-22-+ |=====>==>==>
| +-PERCEPT-23-+ |=====>==>==>      xX_ XxxxX      DETAIL
| +-PERCEPT-24-+ |=====>==>==>      Xx_ xXXXx      DETAIL REV
| +-PERCEPT-25-+ |=====>==>==>
| +-PERCEPT-26-+ |=====>==>==>      8 fixed-len elements, 1
pix/element,
| +-PERCEPT-27-+ |=====>==>==>      4 bits of intensity/element
| +-PERCEPT-28-+ |=====>==>
| +-PERCEPT-29-+ |=====>==>      XXX_ XXXXXXxxxxxxxx      MED SHADE
| +-PERCEPT-30-+ |=====>      xxx_ xxxxxxxXXX      MED REVERSE
| +-PERCEPT-31-+ |=====>
| | position | | | | |      4 run-len elements, 1-16 pix/run,

```

x y & z	4 bits of intensity/element
offset	
x y & z	XXXXXXXXXXXXXXXXXXXXXXXXX LRG SHADE
data adr	XXXXXXXXXXXXXXXXXXXXXXXXX LRG REVERSE
FORMATS:	
DATA +	4 run-len elements, 16*(1-16) pix/run,
bit map --+	4 bits of intensity/element
display +	
link adr --+	X X X XXX X X XX X TEXTURE
+-----+	
+-----+	32 fixed-len elements, 1 pix/element,
	4 bits of overall intensity

In CAPS are parameters used in sprite formatting. Xx__xX EDGE

ALL SPRITES CAN BE	4 fixed-len elements reflected to form
INVERTED, REFLECTED	8 elements plus a 1-256 pixel offset,
& INVERT-REFLECTED	1 pix/element, 4 bits of intensity/ele.

DAZZLER
MED CARTOON
LRG CARTOON
AIR BRUSH
COLOR-OR
DETAIL
DETAIL REV
MED SHADE
MED REVERSE
LRG SHADE
LRG REVERSE
TEXTURE
EDGE

```
+--PERCEPT--+
| POSITION      |
|  X Y & Z    |
| OFFSET      |
|  X Y & Z    |
| DATA ADR   |
| FORMATS:    |
|  DATA      |
|  BIT MAP    |
|  DISPLAY    |
|  LINK ADR   |
+-----+
```

FUNCTIONAL PRIORITIZATION:

There are 8192 levels of functional prioritization in each PENNY.

The functional priority is divided between software controllable and hard-ware fixed priority fields. Soft priority has precedence over hard priority. When portraying a three-dimensional space, soft priority is merely the z-coordinate of a sprite, so the terms 'soft priority', 'z-coordinate' and 'z-level' are interchangeable. Final system display priority between PENNYs is determined by color palette mapping.

```

      .------.
      |          Z = 255          |          SOFT PRIORITY
      |          lowest           |  -----
--
      .------. | Soft priority is
determined      .------.| | by z-coordinate (256
levels).
      oooooooooooooooooooooooooo|| |
      o          Z = 0          o|| |
      o          highest        o|| |
      o          FRONT OF      o||_|
      o          TV SCREEN     o||.
      o                      o||.
      o                      o||
      o                      o|
      oooooooooooooooooooooooooo

      oooooooooooooooooooooooooo
      o|          PERCEPT 31  o|          HARD PRIORITY
      o |          lowest      o |  -----
      oooooooooooooooooooooooooo | Within a soft priority level
      oooooooooooooooooooooooooo| | hard priority is determined
      oooooooooooooooooooooooooo|| | by percept number (32 levels).
      *          PERCEPT 0    *|| |
      *          highest        *|| |
      *                      *|| |
      *                      *||_|
      * pixel(x,y,z,percept) *||.
      * @                      *||.
      * /                      *||
determined
      * /                      *|
      */*****

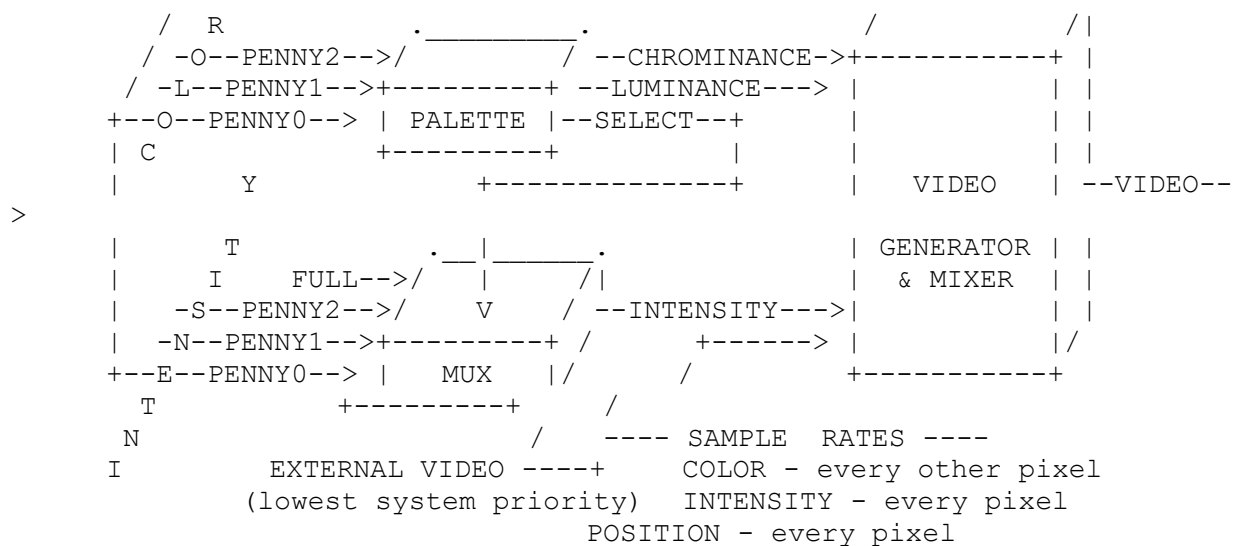
```

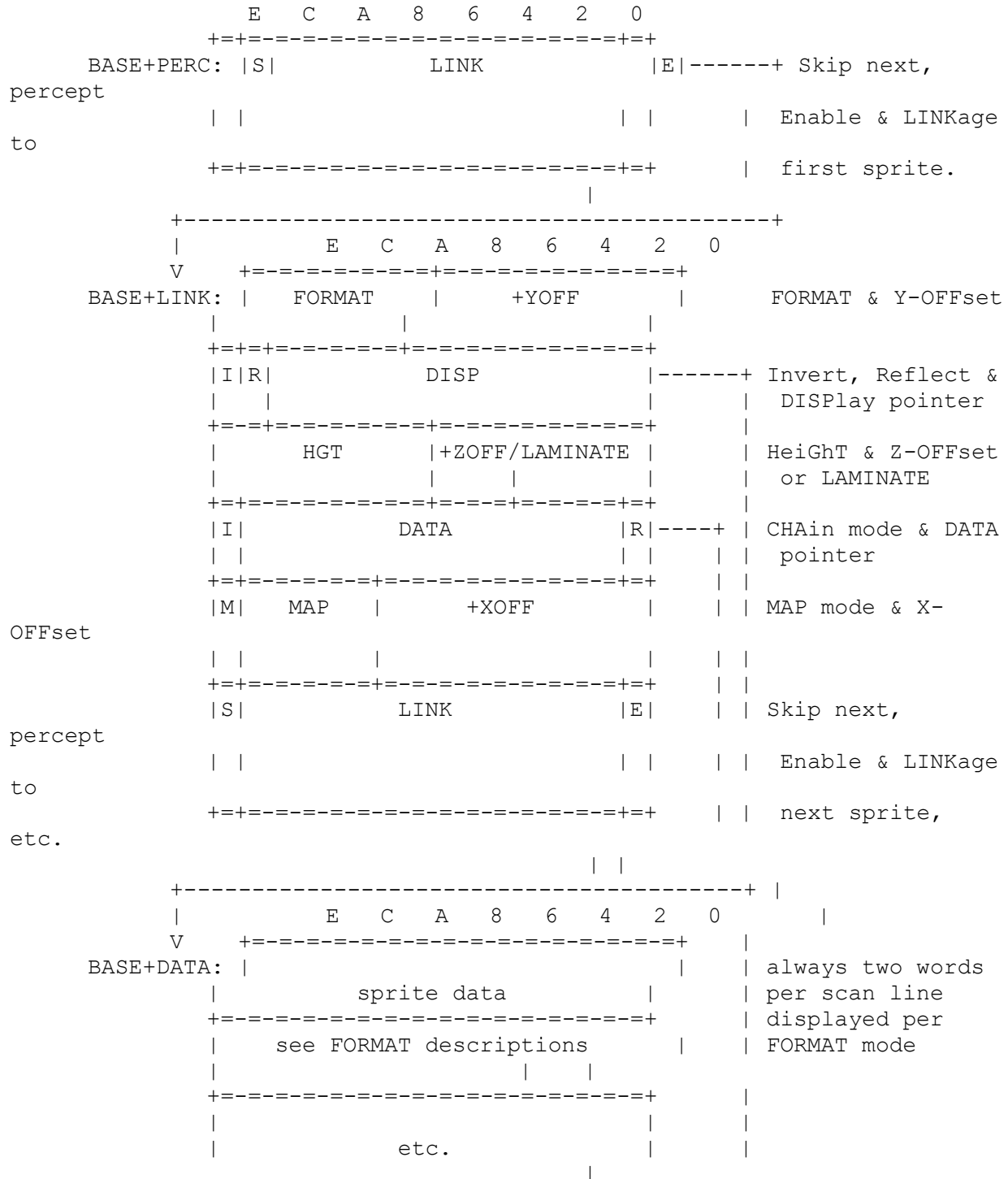
PALETTE PRIORITY

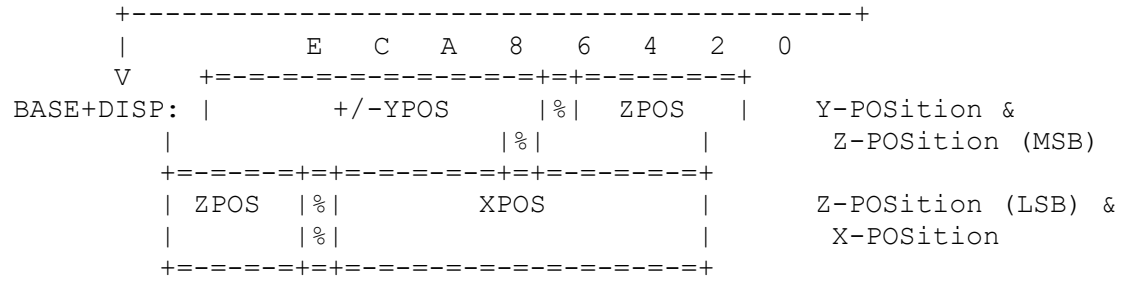
System priority is

by color palette mapping.

_____.







.....
: .FORMAT. . . .+YOFF. . .
:DISP.
: . . .HGT.+ZOFF. . .
: I:DATA.
: M: .MAP. . . .+XOFF. . . .
: S:LINK.

.....
0E0000: : 0:002A. : 0E0002: : 0:0182.

.....
0E002A: :: : 0E0182: ::
:: ::
:: ::
:: ::
:: ::

.....
:: ::
:: ::
:: ::
:: ::
:: ::
:: ::

.....
:: ::
:: ::
:: ::
:: ::
:: ::
:: ::

.....
::::::
::::::
::::::
::::::
::::::
::::::
::::::

.....
::
::
::
::
::
::

.....

:.....:.....:
:.....:.....:
:.....:.....:
:.....:.....:
:.....:.....:
:.....:.....:

S (Skip next switch) DEFINED:

The skip next switch causes the next sprite to be skipped and resumes processing with the succeeding sprite which is linked from the skipped sprite as though the skipped sprite were processed normally.

E (percept Enable) DEFINED:

The percept enable allows the next link to be processed. If not enabled, the percept is terminated and no further sprites will be generated by it until the next screen.

I & R (Invert & Reflect switches) DEFINED:

Invert and reflect work on sprite data and do just as the names imply.

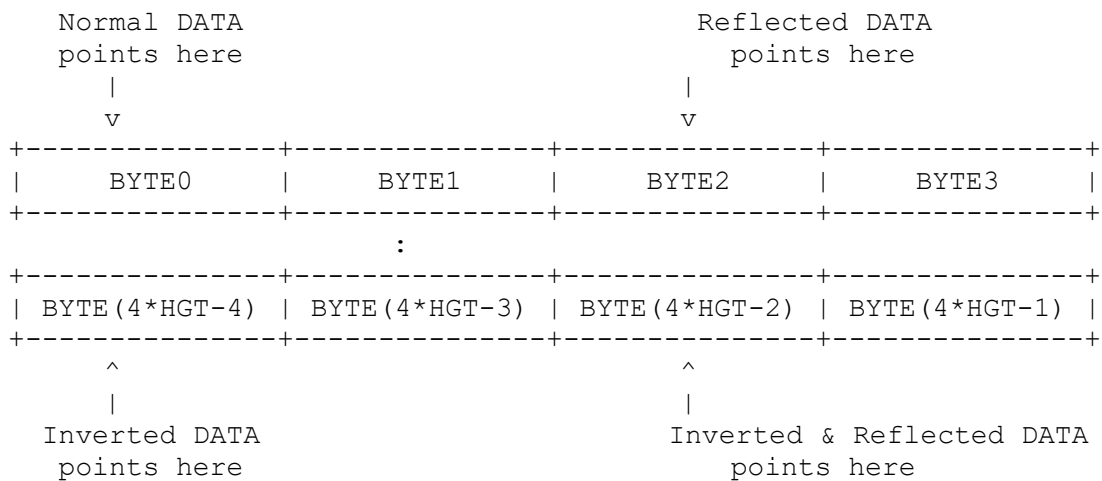
If the invert switch is on, DATA must point to the last word of sprite data as the pointer is decremented with each line instead of

incremented. If the reflect switch is on, the CHAINin/out function is reversed (see CHAIN mode).

```

                                +-----+-----+-----+-----+
DATA: =.                        | BYTE0 | BYTE1 | BYTE2 | BYTE3 |
RDATA:    =DATA+3                +-----+-----+-----+-----+
IDATA:    =DATA+4*(HGT-1)
IRDATA:   =RDATA+4*(HGT-1)

```



HGT (HeiGhT) DEFINED:

Height defines the vertical height of the sprite in scan lines.

Remember that a line of sprite data is always two words.

CHA (CHAI mode) DEFINED:

Chaining allows two percepts to interact. The chain input (chainin) to a percept is an enable for the generation of the sprite for that percept. The chain output (chainout) from a percept is the enable (chainin) to the next higher numbered percept (or lower numbered if the reflect switch is on ... see Invert & Reflect switches). An unchained percept is always enabled.

If not reflected

```

+-----+
| PERCEPT n |
from PERCEPT n-1 ---->|chainin chainout|-----> to PERCEPT n+1
+-----+
```

If reflected

```

+-----+
| PERCEPT n |
to PERCEPT n-1 <----|chainout chainin|<----- from PERCEPT n+1
+-----+
```

CHA	Mode	line of sprite starts on	DATA out is enabled	Chainout equals
00	unchained	X = XPOS+XOFF	always	DATA > 0
01	stencil	X = XPOS+XOFF	by chainin	chainin
10	startile	X = XPOS+XOFF	always	line done this sprite
11	tile	chainin	always	line done this sprite

Stenciling and tiling is discussed in detail in the APPENDIX.

MAP (MAP mode) DEFINED:

There are 32 map functions for each view as follows:

	-- M a p W i d t h --					
	n u m b e r o f s p r i t e s *					
	4	8	1 6	3 2		
	+-----+-----+-----+-----+					
n	4	1 F	1 E	1 D	1 C	* The number of
pixels that		+-----+-----+-----+-----+				each sprite
u						
produces is						
M m	8	1 B	1 A	1 9	1 8	determined by its
FORMAT.						
a b		+-----+-----+-----+-----+				
p e	1 6	1 7	1 6	1 5	1 4	Map height & width
r		+-----+-----+-----+-----+				
refer to the						
H	3 2	1 3	1 2	1 1	1 0	arrangement of data
in memory,						
e o		+-----+-----+-----+-----+				only ... not to the
size of						
i f	6 4	0 F	0 E	0 D	0 C	screen graphics.
They are used						
g		+-----+-----+-----+-----+				to control memory
addresses and						
h l	1 2 8	0 B	0 A	0 9	0 8	for vertical &
horizontal wrap-						
t i		+-----+-----+-----+-----+				around only. Reflect
reverses						
n	2 5 6	0 7	0 6	0 5	0 4	
e		+-----+-----+-----+-----+				horizontal wrap-
around only.						
s	5 1 2	0 3	0 2	0 1		
		+-----+-----+-----+				
		+-----+				
		0 0 MAP mode off				
		+-----+ Sprite is graphics stamp				

XOFF, YOFF & ZOFF (X-OFFset, Y-OFFset & Z-OFFset) DEFINED:

These unsigned binaries specify the offsets to the upper left-hand corner of the sprite relative to the sprite position as defined by XPOS, YPOS & ZPOS.

XPOS, YPOS & ZPOS (X-POSition, Y-POSition & Z-POSition) DEFINED:

These signed binaries are the x,y & z values of course or group

positioning to which XOFF, YOFF & ZOFF are added to arrive at the sprite's true x, y & z positions on the screen. Both these parameters and their associated offsets are full resolution and full screen so that they can be individually used to perform complete positioning. The presence of the offsetting ability, though, makes group motion and/or screen scrolling much easier.

FORMAT (FORMAT mode) DEFINED:

FORMAT SELECTION GUIDE

```

FORMAT ('bbbb' is overall brightness,
|      'cccc' is overall color and
|      'iiii' is overall intensity)
|  AUTOMATIC EDGE SMOOTHING
|  |  NUMBER OF BITS OF GRAPHICS DATA PER ELEMENT
|  |  |  COLOR DETERMINED BY (Format, Graphics)
|  |  |  |  INTENSITY DETERMINED BY (Format, Graphics)
|  |  |  |  |  GRAPHICS ELEMENT TYPE (Fixed-length, Run-length)
|  |  |  |  |  |  ELEMENT SIZE IN PIXELS
|  |  |  |  |  |  |  NUMBER OF ELEMENTS PER SPRITE
|  |  |  |  |  |  |  |  MAX SPRITE SIZE IN PIXELS
|  |  |  |  |  |  |  |  |  + indicates offset(s), also
V      V V V V V      V      V      V      NAME (page)
===== = = = = = ===== == ===
=====

```

===== Z-PRIORITIZED COLOR SPRITES

```

001bbbb - 4 G F F      2      8      16  DAZZLER (16)
010bbbb X 8 G F R      2 to 32    4    128  MEDIUM CARTOON (17)
011bbbb X 8 G F R     32 to 512    4   2048  LARGE CARTOON (18)
110cccc - 1 F - F      2      32     64  AIR BRUSH (25)

```

===== NON-PRIORITIZED COLOR SPRITE

```

111cccc - 1 F - F      2      32     64  COLOR-OR ( )

```

===== INTENSITY SPRITES

```

Self laminate bit
| 0 => Laminate on color sprite specified by LAMINATE parameter
| 1 => Self-laminate (always on)
v
000S000 - 4 - G F      1      8      8+ EDGE ENHANCEMENT (27)
" S010 - 4 - G F      1      8      8  DETAIL (19)
" S011 - 4 - G F      1      8      8  DETAIL REVERSE (20)
" S100 - 8 - G R      1 to 16    4    64  MEDIUM SHADE (21)
" S101 - 8 - G R      1 to 16    4    64  MEDIUM REVERSE (22)
" S110 - 8 - G R     16 to 256    4   1024  LARGE SHADE (23)
" S111 - 8 - G R     16 to 256    4   1024  LARGE REVERSE (24)

```

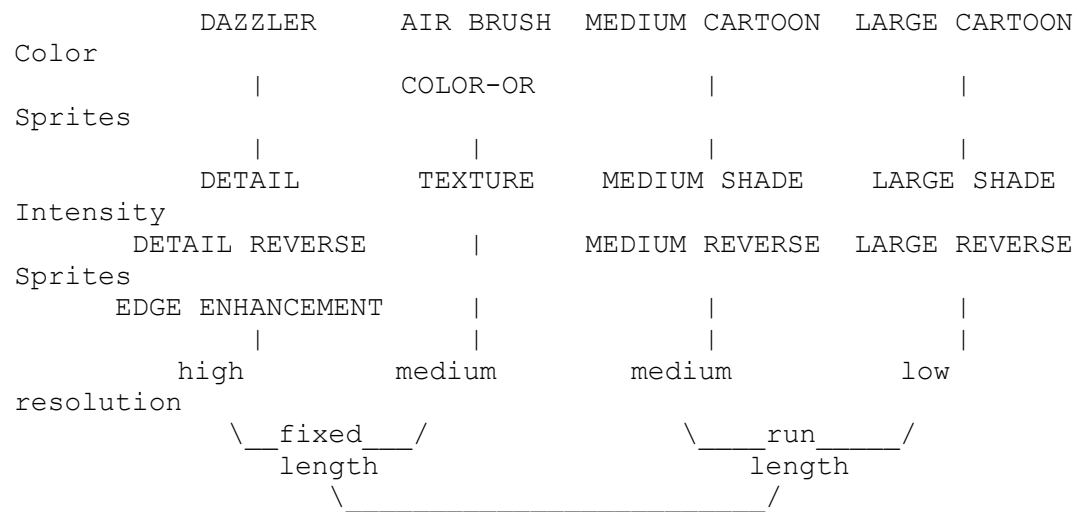
```

Self laminate bit
| 0 => Laminate on color sprite specified by LAMINATE parameter
| 1 => Self-laminate (always on)
v

```

10Siiii - 1 - F F 1 32 32 TEXTURE (26)

SPRITE FAMILY TREE



DAZZLER SPRITE; FORMAT = 001bbbb

Optimized for high resolution poly-chromatic detail, this sprite
has 16 pixels of graphics. Pixel color is determined by the graphics
data and overall brightness is determined by the FORMAT.

```

<----- Sprite data word 1 -----> <----- Sprite data word 2 ----->
F      B      7      3      0 F      B      7      3      0
+-----+-----+-----+-----+-----+-----+-----+-----+
| C1  | C2  | C3  | C4  | C5  | C6  | C7  | C8  |
+-----+-----+-----+-----+-----+-----+-----+
left ----- 2 pixels per nibble -----> right

```

```

===== PROGRAMMING =====
===== ELEMENT COLOR =====  ===== OVERALL BRIGHTNESS =====

```

Cn	color	comments	FORMAT	brightness	comments
====	====	=====	=====	=====	=====
0000	0000	transparent	0110000	0000	full LUM
--	--	--	--	--	--
1110	1110	color-14	1110	1110	2/16ths of LUM
1111	-----> 15-switch		1111	-----> 15-switch	
	v			v	
	Color is a			Brightness is a	
ZPOS+ZOFF	<--- function of /Z		ZPOS+ZOFF	<--- function of Z	
=====	=====		=====	=====	
+0000%%%	1111	closest to	+0000%%%	0000	closest to
		screen, color-15			screen, full LUM
--	--	--	--	--	--
+1111%%%	0000	furthest from	+1111%%%	1111	furthest from
		screen, t'parent			screen, 1/16ths LUM

SCREEN GRAPHICS (one typical element out of eight total elements)

```

+--+
|  |
+--+
-->| |<-- 2 pixels

```

Does not automatically generate edge smoothing.

Can be laminated by intensity sprites.

Note the similarities & differences between this sprite & its kin, the DETAIL & DETAIL REVERSE sprites.

NOTES:

MEDIUM CARTOON SPRITE; FORMAT = 010bbbb

```

<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->
F      B      7      3      0 F      B      7      3      0
+-----+-----+-----+-----+-----+-----+-----+-----+
|L1/2-1 : C1 |L2/2-1 : C2 |L3/2-1 : C3 |L4/2-1 : C4 |
+-----+-----+-----+-----+-----+-----+-----+-----+
left ----- L pixels per byte -----> right

===== PROGRAMMING =====
===== ELEMENT COLOR =====
===== OVERALL BRIGHTNESS =====

Cn  color      comments  FORMAT brightness  comments
====
=====
0000      0000      transparent  0110000  0000  full LUM
--  --
1110      1110      color-14      1110  1110  2/16ths of LUM

1111 -----> 15-switch      1111 -----> 15-switch
          |
          v
          Color is a          Brightness is a
ZPOS+ZOFF <--- function of /Z  ZPOS+ZOFF <--- function of Z
=====
+0000%%% 1111  closest to  +0000%%% 0000  closest to
          screen, color-15  screen, full LUM
--  --
+1111%%% 0000  furthest from  +1111%%% 1111  furthest from
          screen, t'parent    screen, 1/16ths LUM

```

SCREEN GRAPHICS (one typical element out of four total elements)

```

+--+--+--+--+--+--+--+--+--+  Automatically generates left & right edge
|                                |  smoothing for each element.
+--+--+--+--+--+--+--+--+--+
|<--- L pixels -->|  Can be laminated by intensity sprites.

```

```

For L =  StoreNote the similarities and differences between
-----this sprite and its kin, the MEDIUM SHADE &
2       0       MEDIUM REVERSE sprites.
4       1
6       2
etc.

```

NOTES:

LARGE CARTOON SPRITE; FORMAT = 011bbbb

```

<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->
F      C B      8 7      4 3      0 F      C B      8 7      4 3      0
+-----+-----+-----+-----+-----+-----+-----+-----+
|L1/32-1: C1 |L2/32-1: C2 |L3/32-1: C3 |L4/32-1: C4 |
+-----+-----+-----+-----+-----+-----+-----+
left ----- L pixels per byte -----> right

===== PROGRAMMING =====
===== ELEMENT COLOR =====      ===== OVERALL BRIGHTNESS =====

Cn  color      comments      FORMAT brightness      comments
====
=====
0000      0000      transparent      0110000      0000      full LUM
--      --      --      --
1110      1110      color-14      1110      1110      2/16ths of LUM

1111 -----> 15-switch      1111 -----> 15-switch
          |
          v
          Color is a          Brightness is a
ZPOS+ZOFF <--- function of /Z      ZPOS+ZOFF <--- function of Z
=====
+0000%%% 1111      closest to      +0000%%% 0000      closest to
          screen, color-15          screen, full LUM
--      --      --      --
+1111%%% 0000      furthest from      +1111%%% 1111      furthest from
          screen, t'parent          screen, 1/16ths LUM

```

SCREEN GRAPHICS (one typical element out of four total elements)

```

+--+--+--+--+--+--+--+--+--+  Automatically generates left & right edge
|                                |  smoothing for each element.
+--+--+--+--+--+--+--+--+--+
|<--- L pixels -->|  Can be laminated by intensity sprites.

```

```

For L =      StoreNote the similarities and differences between
-----      -----this sprite and its kin, the LARGE SHADE &
32           0           LARGE REVERSE sprites.
64           1
96           2
etc.

```

NOTES:

```
DETAIL SPRITE; FORMAT = 000S010
```

```

      ^
      |
Self laminate bit -----+
0 => Laminate on color sprite specified by LAMINATE parameter
1 => Self-laminate (always on)

```

Optimized for high resolution multi-intensity detail, this sprite has 8 pixels. Pixel intensity is determined by the graphics data. It is very useful for creating text and for edge enhancement of color sprites.

```

<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->
F      B      7      3      0 F      B      7      3      0
+-----+-----+-----+-----+-----+-----+-----+-----+
| I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 |
+-----+-----+-----+-----+-----+-----+-----+
left ----- 1 pixel per nibble -----> right

```

```
===== PROGRAMMING ELEMENT INTENSITY =====
```

In	intensity	comments
====	====	=====
0000	0000	full LUM as stored in palette (no contrast)
--	--	
1110	1110	2/16ths of LUM stored in palette (max contrast)

```

1111 -----> 15-switch -----+
      |
ZPOS+ZOFF <----- intensity is a function of /Z <+
=====
+0000%%% 1111 closest to screen, 1/16th LUM (max contrast)
--      --
+1111%%% 0000 furthest from screen, full LUM (no contrast)

```

SCREEN GRAPHICS (one typical element out of eight total elements)

```

+-+      If the 'S' bit in the FORMAT is off, this
| |      sprite must be laminated to a color sprite.
+-+
-->| |<-- 1 pixel      Note the similarities & differences between
                        this sprite & its kin, the DAZZLER & DETAIL
                        REVERSE sprites.

```

NOTES:

DETAIL REVERSE SPRITE; FORMAT = 000S011

^
|

Self laminate bit -----+

0 => Laminate on color sprite specified by LAMINATE parameter

1 => Self-laminate (always on)

that This sprite is functionally identical to the DETAIL sprite except
be the values of INTENSITY stored are internally complimented before
a written on the screen resulting in an inverse video (but not
complimentary color) display. It is very useful for creation of
cursor. Any font character can be transformed into that same
character displayed over the cursor with a single bit set in its
FORMAT field (ie, change 0000010 to 0000011).

```

<----- Sprite data word 1 -----> <----- Sprite data word 2 ----->
F      B      7      3      0 F      B      7      3      0
+-----+-----+-----+-----+-----+-----+-----+-----+
| I1  | I2  | I3  | I4  | I5  | I6  | I7  | I8  |
+-----+-----+-----+-----+-----+-----+-----+
left ----- 1 pixel per nibble -----> right

```

===== PROGRAMMING ELEMENT INTENSITY =====

In	intensity	comments
====	====	=====
0000	1111	1/16ths of LUM stored in palette (max contrast)
--	--	
1110	0001	14/16ths of LUM stored in palette (min contrast)
1111		-----> 15-switch -----+
ZPOS+ZOFF	<-----	intensity is a function of /Z <+ =====
+0000%%%	0000	closest to screen, full LUM (no contrast)
--	--	
+1111%%%	1111	furthest from screen, 1/16th LUM (max contrast)

IMPORTANT: Note that though the intensities generated are reversed from the DETAIL sprite, the 15-switch (In = 1111) is the same.

SCREEN GRAPHICS (one typical element out of eight total elements)

```

+--+
| |

```

If the 'S' bit in the FORMAT is off, this sprite must be laminated to a color sprite.

++
-->| |<-- 1 pixel Note the similarities & differences between
 this sprite & its kin, the DAZZLER & DETAIL
 sprites.

NOTES:

MEDIUM SHADE SPRITE; FORMAT = 000S100

^
|

Self laminate bit -----+

0 => Laminate on color sprite specified by LAMINATE parameter

1 => Self-laminate (always on)

<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->

F B 7 3 0 F B 7 3 0

+-----+-----+-----+-----+-----+-----+-----+-----+-----+

| L1-1 : I1 | L2-1 : I2 | L3-1 : I3 | L4-1 : I4 |

+-----+-----+-----+-----+-----+-----+-----+-----+-----+

left ----- L pixels per byte -----> right

===== PROGRAMMING ELEMENT INTENSITY =====

In	intensity	comments
=====	=====	=====
0000	0000	full LUM as stored in palette (no contrast)
--	--	
1110	1110	2/16ths of LUM stored in palette (max contrast)

1111 -----> 15-switch -----+

|

ZPOS+ZOFF <----- intensity is a function of /Z <--+

=====

+0000%% 1111 closest to screen, 1/16th LUM (max contrast)

-- --

+1111%% 0000 furthest from screen, full LUM (no contrast)

SCREEN GRAPHICS (one typical element out of four total elements)

+--+--+--+--+--+--+--+--+

| |

+--+--+--+--+--+--+--+--+

If the 'S' bit in the FORMAT is off, this sprite must be laminated to a color sprite.

|<--- L pixels -->|

For L = StoreNote the similarities and differences between

-----this sprite and its kin, the MEDIUM CARTOON &

1 0 MEDIUM REVERSE sprites.

2 1

3 2

etc.

NOTES:

MEDIUM REVERSE SPRITE; FORMAT = 000S101

^

|

Self laminate bit -----+

0 => Laminate on color sprite specified by LAMINATE parameter

1 => Self-laminate (always on)

<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->

F B 7 3 0 F B 7 3 0

+-----+-----+-----+-----+-----+-----+-----+-----+-----+

| L1-1 : I1 | L2-1 : I2 | L3-1 : I3 | L4-1 : I4 |

+-----+-----+-----+-----+-----+-----+-----+-----+-----+

left ----- L pixels per byte -----> right

===== PROGRAMMING ELEMENT INTENSITY =====

In	intensity	comments
=====	=====	=====
0000	1111	1/16ths of LUM stored in palette (max contrast)
--	--	
1110	0001	14/16ths of LUM stored in palette (min contrast)

1111 -----> 15-switch -----+

|

ZPOS+ZOFF <----- intensity is a function of /Z <--+

=====

+0000%% 0000 closest to screen, full LUM (no contrast)

-- --

+1111%% 1111 furthest from screen, 1/16th LUM (max contrast)

IMPORTANT: Note that though the intensities generated are reversed from the MEDIUM SHADE sprite, the 15-switch (In = 1111) is the same.

SCREEN GRAPHICS (one typical element out of eight total elements)

+--+--+--+--+--+--+--+--+

| |

+--+--+--+--+--+--+--+--+

|<--- L pixels -->| If the 'S' bit in the FORMAT is off, this sprite must be laminated to a color sprite.

For L = StoreNote the similarities and differences between

----- this sprite and its kin, the MEDIUM CARTOON &

1 0 MEDIUM SHADE sprites.

2 1

3 2

etc.

NOTES:


```
LARGE SHADE SPRITE; FORMAT = 000S110
```

$$\begin{matrix} \wedge \\ | \end{matrix}$$

Self laminate bit -----+

```
0 => Laminate on color sprite specified by LAMINATE parameter
```

1 => Self-laminate (always on)

```
<----- Sprite data word 1 -----> <----- Sprite data word 2 ----->
```

F B 7 3 0 F B 7 3 0

+-----+-----+-----+

L1/16-1:	I1	L2/16-1:	I2	L3/16-1:	I3	L4/16-1:	I4	
----------	----	----------	----	----------	----	----------	----	--

left ----- L pixels per byte -----> right

```
===== PROGRAMMING ELEMENT INTENSITY =====
```

In intensity comments

=====

0000 0000 full LUM as stored in palette (no contrast)

— — — — —

```
1110          1110  2/16ths of LUM stored in palette (max
```

```
contrast)
```

```
1111 -----> 15-switch -----+
```

1

```
ZPOS+ZOFF <----- intensity is a function of /Z <+
```

=====

+0000%% 1111 closest to screen, 1/16th LUM (max contrast)

— — — — —

```
+1111%%%% 0000 furthest from screen, full LUM (no contrast)
```

SCREEN GRAPHICS (one typical element out of four total elements)

$$+ - + - + - + - + - + - + - +$$

Downloaded from <http://ajphaphysiol.org/> on September 11, 2012

```
+---+---+---+---+---+---+   If the 'S' bit in the FORMAT is off, this
|<--- L pixels -->|         sprite must be laminated to a color sprite.
```

For L = StoreNote the similarities and differences between

```
-----  ----this sprite and its kin, the LARGE CARTOON &
```

```
16      0      LARGE REVERSE sprites.
```

32 1

48 2

etc.

NOTES :

LARGE REVERSE SPRITE; FORMAT = 000S111

^

|

Self laminate bit -----+

0 => Laminate on color sprite specified by LAMINATE parameter

1 => Self-laminate (always on)

<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->

F B 7 3 0 F B 7 3 0

+-----+-----+-----+-----+-----+-----+-----+-----+-----+

|L1/16-1: I1 |L2/16-1: I2 |L3/16-1: I3 |L4/16-1: I4 |

+-----+-----+-----+-----+-----+-----+-----+-----+-----+

left ----- L pixels per byte -----> right

===== PROGRAMMING ELEMENT INTENSITY =====

In	intensity	comments
=====	=====	=====
0000	1111	1/16ths of LUM stored in palette (max contrast)
--	--	
1110	0001	14/16ths of LUM stored in palette (min contrast)
1111		
		-----> 15-switch -----+
ZPOS+ZOFF	<-----	intensity is a function of /Z <+-----
=====		
+0000%%%	0000	closest to screen, full LUM (no contrast)
--	--	
+1111%%%	1111	furthest from screen, 1/16th LUM (max contrast)

IMPORTANT: Note that though the intensities generated are reversed from the LARGE SHADE sprite, the 15-switch (In = 1111) is the same.

SCREEN GRAPHICS (one typical element out of eight total elements)

+--+--+--+--+--+--+--+

|

+--+--+--+--+--+--+--+

|<--- L pixels -->| If the 'S' bit in the FORMAT is off, this sprite must be laminated to a color sprite.

For L = StoreNote the similarities and differences between
 ----- this sprite and its kin, the LARGE CARTOON &
 16 0 LARGE SHADE sprites.
 32 1
 48 2
 etc.

NOTES:


```
AIR BRUSH SPRITE; FORMAT = 110cccc
```

Optimized for high resolution mono-chromatic detail, this sprite has 64 pixels of color graphics. Pixel color is determined by the FORMAT.

[illegible]

```
===== PROGRAMMING ELEMENT COLOR =====
```

C	FORMAT	color	comments
0	110%%%	0000	transparent
1	" 0000	0000	dissolve (see Appendix for its use)
1	" --	--	
1	" 1110	1110	color-14
1	" 1111	----->	15-switch
		v	
	ZPOS+ZOFF	<-----	Color is a function of /Z
1	0000%%%	1111	closest to screen, color-15
1	--	--	
1	1111%%%	0000	furthest from screen, dissolve (see Appendix for its use)

SCREEN GRAPHICS (one element out of 32 total elements)

```

      +-+--+      Does not automatically generate edge smoothing.
      |  |
      +-+--+      Note the similarities and differences between
--->|  |<--- 2 pixels this sprite and its kin, the COLOR-OR &
TEXTURE          sprites.

```

NOTES :

COLOR-OR; FORMAT = 111cccc

Optimized for high resolution mono-chromatic detail, this sprite has 64 pixels of color graphics. Pixel color is determined by the FORMAT. It differs from the AIR BRUSH sprite in only two respects. First, it does not participate in display prioritization; it outputs its color word whenever it has a pixel defined as non-transparent. And second, that color word output is logically 'or'ed into whatever color word exists from any other sprites in a manner analogous to the way that intensities are logically 'or'ed on the intensity bus. This is the only color sprite which is not prioritized and which 'or's its color on to the color bus. It is especially useful for creating color planed bit map displays (see Appendix for examples).

[illegible]

```
===== PROGRAMMING ELEMENT COLOR =====
```

C	FORMAT	color	comments
====	=====	=====	=====
0	11111111	0000	transparent
1	" 0000	0000	transparent
1	" --	--	
1	" 1110	1110	color-14
1	" 1111	----->	15-switch
		v	
	ZPOS+ZOFF	<----	Color is a function of /Z
	=====		
1	00001111	1111	closest to screen, color-15
1	--	--	
1	11111111	0000	furthest from screen, transparent

SCREEN GRAPHICS (one element out of 32 total elements)

```

+--+--+ Does not automatically generate edge smoothing.
|  |  |
+--+--+ Note the similarities and differences between

```

--->| |<--- 2 pixels this sprite and its kin, the AIR BRUSH &
TEXTURE sprites.

NOTES:

```
TEXTURE SPRITE; FORMAT = 10SiIII
```

```
      ^
```

```
      |
```

```
Self laminate bit -----+
```

```
0 => Laminate on color sprite specified by LAMINATE parameter
```

```
1 => Self-laminate (always on)
```

Optimized for high resolution mono-intensity detail, this sprite
has 32 pixels of intensity graphics. Pixel intensity determined by the
FORMAT parameter.

```
<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->
F E D C B A 9 8 7 6 5 4 3 2 1 0 F E D C B A 9 8 7 6 5 4 3 2 1 0
+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+
|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|I|
+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+==+
left ----- 1 pixel per bit -----> right
```

```
===== PROGRAMMING ELEMENT INTENSITY =====
```

I	FORMAT	intensity	comments
0	101%%%	%%%	full LUM (no contrast)
1	" 0000	0000	full LUM (no contrast)
1	-- --		
1	" 1110	1110	2/16ths LUM (max contrast)
1	" 1111 -----> 15-switch -----+		
	ZPOS+ZOFF <-----		intensity is a function of /Z <+=====
1	+0000%%%	1111	closest to screen, 1/16th LUM (max contrast)
1	-- --		
1	+1111%%%	0000	furthest from screen, full LUM (no contrast)

```
SCREEN GRAPHICS (one element out of 32 total elements)
```

```

+--+
| |
+--+
--->| |<--- 1 pixel
between
      If the 'S' bit in the FORMAT is off, this
      sprite must be laminated to a color sprite.
      Note the similarities and differences
      this sprite and its kin, the AIR BRUSH &
      COLOR-OR sprites.
NOTES:
```


EDGE ENHANCEMENT SPRITE; FORMAT = 000S000

^

|

Self laminate bit -----+

0 => Laminate on color sprite specified by LAMINATE parameter

1 => Self-laminate (always on)

not
edge
This sprite is useful to add edge smoothing to sprites which do
automatically generate edge smoothing or to further enhance an
which has been automatically generated for further anti-aliasing.

```

<----- Sprite data word 1 ----> <----- Sprite data word 2 ---->
F          7          3          0 F          B          7          0
+-----+-----+-----+-----+-----+-----+-----+
| LEFTOFFSET | I1  | I2  | I3  | I4  | RIGHTOFFSET |
+-----+-----+-----+-----+-----+-----+
left ----- 1 pixel -----> edge
right <--- per nibble ---- edge

```

===== PROGRAMMING ELEMENT INTENSITY =====

In	intensity	comments
=====	=====	=====
0000	0000	full LUM as stored in palette (no contrast)
--	--	
1110	1110	2/16ths of LUM stored in palette (max contrast)

```

1111 -----> 15-switch -----+
                               |
ZPOS+ZOFF <----- intensity is a function of /Z <--+
=====
+0000%%% 1111 closest to screen, 1/16th LUM (max contrast)
--      --
+1111%%% 0000 furthest from screen, full LUM (no contrast)

```

DETAIL OF EDGE

```

+--+--+--+--+--+--+--+
  |1|2|3|4|4|3|2|1|
+--+--+--+--+--+--+--+
    1 pixel --->| |<--- |
<----- LEFTOFFSET ----->|<----- RIGHTOFFSET ----->

```

RIGHTOFFSET is only needed if the sprite is to be reflectable in
which
case LEFTOFFSET plus RIGHTOFFSET must be a constant for all lines.

If the 'S' bit in the FORMAT is off, this sprite must be laminated
to a color sprite.

NOTES:

PALETTE MEMORY MAP:

```

+----- COLOR from PENNY2
|+----- COLOR from PENNY1
||+----- COLOR from PENNY0
|||
||| F A 5 1 0 <--- bit
VVV +-----+-----+-----+-----+
0EC000: | LUM | P1 | P0 |ISS| This is the background
color switch.
+-----+-----+-----+-----+
0EC00k: | LUM | P1 | P0 |ISS| This is PENNY0 color-k.
+-----+-----+-----+-----+
0EC0j0: | LUM | P1 | P0 |ISS| This is PENNY1 color-j.
+-----+-----+-----+-----+
0ECi00: | LUM | P1 | P0 |ISS| This is PENNY2 color-i.
+-----+-----+-----+-----+
0ECijk: | LUM | P1 | P0 |ISS| Combinations of colors
can visual
+-----+-----+-----+-----+ be used to determine
^ ^ ^ ^ priority between
PENNY's or
| | | | for color mixing,
shading,
LUMinance Phaser1 Phaser0 | special effects, etc.
|
Intensity Source Select -----+

00 ==> intensity determined by intensity sprite from PENNY0
01 ==> intensity determined by intensity sprite from PENNY1
10 ==> intensity determined by intensity sprite from PENNY2
11 ==> intensity (if any) commensurate with stored luminance &
phasers

```

TWO METHODS OF GENERATING COLORS:

1, Find values for LUM, P1 & P0 from the color charts in the APPENDIX or

2, Calculate values for LUM, P1 & P0 as follows:

```
;; select desired amount of intensity of primary colors
off = 0 =< R,G,B =< 23.9 = brightest

```

```
;; store LUMinance

```

```
LUM = INTEGER[.30*R+.59*G+.11*B+.49]

;; store Phaser1
TEMP = .877*(.70*R-.59*G-.11*B)
P1 = INTEGER[TEMP-.49] ; if TEMP < 0
P1 = INTEGER[TEMP+.49] ; if TEMP >= 0

;; & store Phaser0
TEMP = .493*(.89*B-.30*R-.59*G)
P0 = INTEGER[TEMP-.49]MODULO16 ; if TEMP < 0
P0 = INTEGER[TEMP/2+.49]MODULO16 ; if TEMP >= 0
```

WHAT IS STENCILING ?

With stenciling on, the resultant display will be the topological intersection of the stenciled sprite and the stenciling sprite.

```
An example:      stenciled sprite --+      +---- sprite
                  from percept n      |      | from percept n-1
```

	resulting screen			another sprite	
	V	V	V	V	
	.	prioritization	\	.	.
	.	<===	\\	:\	.
	.	<===	\\\	:\	.
Note that the ' ' sprite has the lowest priority so it shows	.	<=<===	\\\\\	:	\ .
	\	<=<===	\\\\\	:	0 .
	\	<=<=<===	\\\\\	:	00
.					
only where the others	\	<=<=<===	\\\\\	:	000
.					
are transparent. The	\0	<=<=<=<===	\\\\\	:	\
0000 .					
'--' sprite has the	\00	<=<=<=<===	\\\\\	:	\
00000 .					
highest priority but	\000	<=<=<=<=<===	\\\\\	:	
000000 .					
is stenciled by the	\000	<=<=<=<===	\\\\\	:	
000000					
'O' sprite so that its	\000	<=<=<=<===	\\\\\	:	\ 000000
left half is transparent	\\000	<=<=<=<===	\\\\\	:	\000000
allowing the ' ' sprite	\\000	<=<=<=<===	\\\\\	:	00000
to show through there.	\000	<=<=<=<===	\\\\\	:	0000
	000	<=<=<=<===	\\\\\	:	000
	00	<=<=<=<===	\\\\\	:	00
	0	<=<=<=<===	\\\\\	:	0

Stenciling is performed irrespective of z-position. Stenciling is performed in x & y only.

Stenciling is different from masking. A mask defines the area where a sprite IS NOT to appear, however, a stencil defines the area where a sprite IS to appear. Masks are data intensive and they are useless

area
used
except as masks. Stencils are data conservative since only the
of interest is reproduced in the data and a stencil can also be
as a graphic sprite for display purposes.

priority.
the
Stenciling does not affect priority and is not affected by
The stenciling sprite does not have to have priority to perform
stenciling function.

like
color
All intensity sprites are automatically laminated. Laminating is
stenciling except that the laminating sprite, which must be a
sprite, must also have priority for the current pixel.

WHAT IS TILING ?

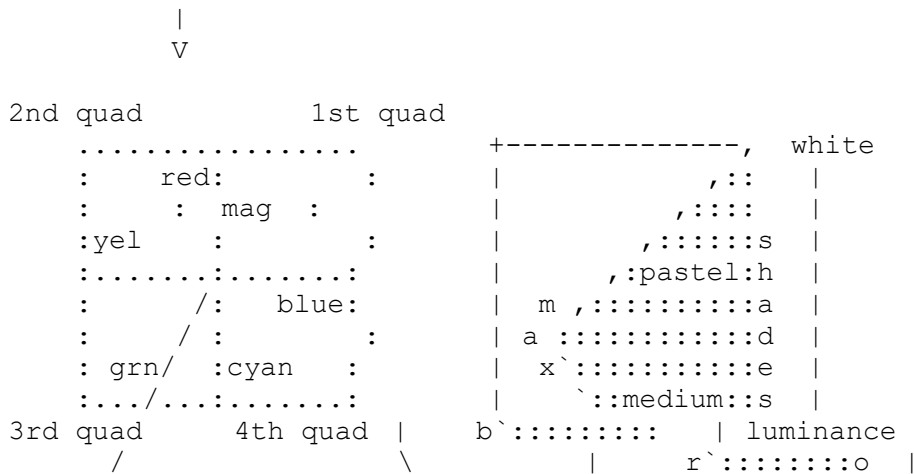
edge It is often desired to cover an area with percepts laid edge to
 with to form a large surface. That is tiling. A tiling chain begins
 with the lowest numbered percept to the left and proceeds to the right
 with incrementally higher numbered percepts, thus:

startile	tile	tile	...
from	from	from	etc.
percept n	percept n+1	percept n+2	...

same Each sprite must have the same number of lines of data and the
 types Y-POSition and Y-OFFset. But, they can be of different FORMAT
 must and they can even be on different z-levels. The left-most sprite
 the be a Startile with X-POSition & X-OFFset defined. The rest of
 sprites in the tiling chain ignore X because each begins when its
 left-hand neighbor finishes.

mix A tiling chain can be of any length and it can be made up of any
 of reflected and unreflected sprites.

Tiling does not affect priority and is not affected by priority.



GREEN'		\	SIDE =>	i`:::f	
SLICE	VIEW =>		l`:dark:		All points in
	OF =>		l`:::g		this slice are
	GREEN =>		i`:::r		identically the
	SLICE =>		a`:::e		same shade of
			n`::y		green.
Detailed top views of the			c`:		
four quadrants are found	+-----		e`		black
on the next four pages.			max-----min		
			amount of color		
			saturation		

point legal NTSC ----> *m <--- minimum allowable LUM for this color
 point color point n <--- maximum allowable LUM for this color

NTSC COLOR CHART

V A L U E S O F P 0

0	1	2	3	4	5		
+	+	+	+	+	+	16	
		F I R S T					
		Q U A D R A N T					
+	+	+	+	+	+	15	
*9	+	+	+	+	+	14	
9							
*8	*9	*10	+	+	+	13	
9	9	10					
*7	*8	*9	*10		+	12	
10	10	10	11	PURE			
			+----+ /	MAGENTA			
*7	*8	*8	*9 /		+	11	V
12	12	11	11				A
			+----+				L
*6	*7	*8	*9	+	+	10	U
12	13	13	12				E
							S
*6	*6	*7	*8	+	+	9	
14	13	14	12				O
							F
*5	*6	*7	*7	*8	+	8	
15	15	15	12	8			P
							1
*5	*5	*6	*7	*8	+	7	
16	16	16	12	8			
*4	*5	*5	*6	*7	+	6	
17	17	15	12	8			
*3	*4	*5	*6	*6	+	5	
18	18	16	12	7			
*3	*4	*4	*5	*6	+	4	
20	20	16	12	8			
*2	*3	*4	*5	*5	+	3	
20	20	16	12	8			

*2	*2	*3	*4	*5	+	2
22	20	16	12	8		
*1	*2	*3	*3	*4	+	1
23	20	16	11	8		
*0	*1	*2	*3	*4	+	0
24	20	16	12	8		

CROSS POINTS (+) ARE ILLEGAL NTSC COLORS...BUT THEY
CAN BE USED FOR MONITORS. MAXIMUM AND MINIMUM
LUM VALUES DO NOT APPLY TO MONITORS EITHER.

V A L U E S O F P 0												
	6	7	8	9	10	11	12	13	14	15	0	
V A L U E S O F P 1	16	+	+	+	+	+	+	+	+	+	+	+
					S E C O N D							
					Q U A D R A N T							
	15	+	+	+	+	+	+	+	+	+	+	+
	14	+	+	+	+	+	PURE	*9	*9	*9	*9	
							RED	9	9	9	9	
						\						
	13	+	+	+	+	+	*9	\ *8	*8	*8	*8	
							10	\ 10	8	9	9	
								+---+				
	12	+	+	+	+	*11	*9	*7	*7	*7	*7	
						11	11	11	9	9	10	
								+---+				
	11	+	+	+	+	*11	*9	*7	*6	*6	*7	
						12	12	12	10	10	12	
	10	+	+	+	+	*13	*11	*9	*7	*6	*6	*6
						13	13	13	13	12	12	12
	9	+	+	+	+	*13	*11	*9	*7	*5	*5	*6
						14	14	14	14	14	13	14
	8	+	+	+	*15	*13	*11	*9	*7	*5	*5	*5
					15	15	15	15	15	15	14	15
	7	+	+	+	*15	*13	*11	*9	*7	*5	*4	*5
					16	16	16	16	16	16	15	16
	6	+	+	*17	*15	*13	*11	*9	*7	*5	*4	*4
				18	18	18	18	18	18	18	17	17
	5	+	+	*17	*15	*13	*11	*9	*7	*5	*3	*3
				19	19	19	19	19	19	19	18	18
	4	+	*19	*17	*15	*13	*11	*9	*7	*5	*3	*3
			20	20	20	20	20	20	20	20	20	20
	3	+	*19	*17	*15	*13	*11	*9	*7	*5	*3	*2
			21	21	21	21	21	21	21	21	21	20
			+---+									
	2		*19	*17	*15	*13	*11	*9	*7	*5	*3	*2

PURE	22	22	22	22	22	22	22	22	22	22	22
YELLOW	----+----+										
1	*19	*17	*15	*13	*11	*9	*7	*5	*3	*1	
	21	22	22	23	23	23	23	23	23	23	23
0	+-----*19	-*17	-*15	-*13	-*11	-*9	--*7	--*5	--*3	--*0	
	21	21	22	22	22	23	23	23	23	24	

V A L U E S O F P 1	0	+	-----	*19 21	-*17 21	-*15 22	-*13 22	-*11 22	-*9 23	-*7 23	-*5 23	-*3 23	-*0 24
	31	+		*19 20	*17 21	*15 21	*13 21	*11 22	*9 22	*7 23	*5 23	*3 23	*2 23
	30	+		*19 20	*17 20	*15 20	*13 21	*11 21	*9 22	*7 22	*5 23	*3 23	*3 23
	29	+		*19 19	*17 20	*15 20	*13 20	*11 21	*9 21	*7 22	*5 22	*4 22	*4 22
	28	+	+		*17 19	*15 19	*13 20	*11 20	*9 21	*7 21	*5 21	*5 21	*5 22
	27	+	+		*17 18	*15 19	*13 19	*11 20	*9 20	*7 20	*6 20	*6 21	*6 21
	26	+	+		*17 18	*15 18	*13 19	*11 19	*9 19	*7 19	*7 19	*7 20	*7 20
	25	+	+		*17 17	*15 18	*13 18	*11 18	*8 18	*8 18	*8 19	*8 19	*8 19
	24	+	+		*17 17	*15 17	*13 17	*11 18	*10 18	*10 18	*10 19	*10 19	*10 19
	23	+	+	+		*15 16	*13 17	*11 17	*11 18	*11 18	*11 18	*11 19	*11 19
	22	+	+	+		*15 16	*13 16	*12 16	*12 17	*12 17	*12 18	*12 18	*12 18
	21	+	+	+	PURE GREEN	*15 15	*13 15	*13 16	*13 16	*13 16	*13 17	*13 17	*13 18
	20	+	+	+		*15 15	*14 15	*14 15	*14 15	*14 15	*14 16	*14 16	*14 16
	19	+	+	+	+	+	+	+	*15 15	*15 15	*15 15	*15 16	*15 16
18	+	+	+	+	+	+	+	+	+	+	+	+	
17	+	+	+	+	+	+	+	+	+	+	+	+	
	6	7	8	9	10	11	12	13	14	15	0		

V A L U E S O F P 0

*0	-----*1	-----*2	-----*3	-----*4	-----+	0	
24	20	16	12	8			
*2	*2	*2	*2	*3	*5	31	
23	20	16	12	8	5		
				+-++			
*3	*3	*3	*3	*3	*5	30	
23	20	16	12	8	5		
				+-++			
*4	*4	*4	*4	*4 \	*5	29	
22	20	16	12	8	PURE BLUE 5		
*5	*5	*5	*5	*5	+	28	V
22	20	16	12	8			A
							L
*6	*6	*6	*6	*6	+	27	U
21	20	16	12	8			E
							S
*7	*7	*7	*7	*7	+	26	
20	20	16	11	7			O
							F
*8	*8	*8	*8	+	+	25	
19	19	15	11				P
	+-++						1
*10	*10	*10	*10	+	+	24	
19	20	16	12				
	+-++						
*11	*11 \	*11	*11	+	+	23	
19	20	16	12				
		PURE CYAN					
*12	*12	*12	*12	+	+	22	
18	19	16	12				
*13	*13	*13	+	+	+	21	
18	18	16					
*14	*14	*14	+	+	+	20	
17	18	16					
*15	*15	*15	+	+	+	19	
16	17	16					
+	+	+	+	+	+	18	
		F O U R T H Q U A D R A N T					
+	+	+	+	+	+	17	
0	1	2	3	4	5		

```

VALUES OF P0 +-----+
              | IF THIS |
              |          |
              | IS      |
              |          |
              | DRAWING |
PREPARED BY MARK FILIPAK | CAN BE |
              | SCALED  |
              +-----+

```

AN EXAMPLE OF '15-SWITCH' COLOR

Suppose that a color sprite has FORMAT = 0010000 ++-- '15-switch's

|| (pixels 3, 4,
& that a partial line of color sprite vv 5 & 6)
data looks like this (in hex) -----> 3FF3....

& the color palette looks like this -----> 000:

001:
(Only the colors of interest in the following 002:
discussion are listed, though, actually, the 003: COLOR 'R'
palette would be filled.) 004:

005: COLOR 'M' 006:
Let each of the eight pixels be represented by an 'R' 007:

Further, suppose that 008:
the LUM stored in the +-> RRRRRRRR (bright) ^ 009:
palette for color 'R' | RRRRRRRR | 00A:

COLOR 'N'
is a constant as rep- | RRRRRRRR LUM stored in 00B:
resented by this -----> RRRRRRRR (dim) | palette 00C:
00D:

Sprite scanned left to right -----> 00E:
00F: COLOR 'O'

ZPOS+ZOFF '15-switch' RR `RR
` color `RR `RR TRANSPARENT
` RR RR (background sprites
show
line) | \ _____ +1111% % % _____ \ ...0.... `RR RR .. through on this
| | \ _____ +1110% % % _____ \ ...1
| | | \ _____ +1101% % % _____ \ ...2 `RRMMMMRR
| | | | \ _____ +1100% % % _____ \ ...3 `RRMMMMRR COLOR 'M'
(note, its
| | | | | \ _____ +1011% % % _____ \ ...4 `RRMMMMRR palette LUM was
the
| | | | | | \ _____ +1010% % % _____ \ ...5.... `RRMMMMRR ..same as COLOR
'R's)
| | | | | | | \ _____ +1001% % % _____ \ ...6 NNNN
| | | | | | | \ _____ +1000% % % _____ \ ...7 RRNNNNRR
| | | | | | | | \ _____ +0111% % % _____ \ ...8 RRNNNNRR COLOR 'N'
(note,
| | | | | | | | | \ _____ +0110% % % _____ \ ...9 `RRNNNNRR its
palette entry
| | | | | | | | | | \ _____ +0101% % % _____ \ ...A.... `RRNNNNRR ..had higher
LUM)
| | | | | | | | | | | \ _____ +0100% % % _____ \ ...B
| | | | | | | | | | | \ _____ +0011% % % _____ \ ...C RR `RR
| | | | | | | | | | | | \ _____ +0010% % % _____ \ ...D RROOOORR COLOR
'O'

```

lower LUM      \|||||+0001%%%\...E      `RROOOORR    (a
found)         \|||||+0000%%%\...F....`RROOOORR ..value
values of Z    \|||||\`                |`\`
               \|||||\`                -%%%%%%%%.....\`..... negative
displayed.     \|||||\`                |`\`                are never
               \|||||\`                |
               \|||||\`                |
               \|||||\` television screen | The '15-switch' color
               \|||||\`                | index = the
compliment     \|||||\`                |
significant    \|||||\`                | of the most
ie,            \|||||\`                | nibble of ZPOS+ZOFF,
              \|||\`                  |
              \|||\`                  | cccc = /Z[7,4]
              \|_____!\`
              ,\`
              ,\`

```

AN EXAMPLE OF '15-SWITCH' BRIGHTNESS

```

++++-- '15-switch'
||||
vvvv

```

Now, suppose that the FORMAT is changed to 0011111

& that the '15-switch' is removed from the partial line of sprite data so that it looks like this -----> 3333....

```

RRRRRRRR ^
RRRRRRRR |
RRRRRRRR LUM stored in palette
RRRRRRRR | for color 'R'

```

Sprite scanned left to right ----->

Overall brightness

ZPOS+ZOFF overall brightness increases as sprite moves toward

screen.

```

.
| \ _____ +11111111 \ ...1/16.... rrrrrrrr -- LUM/16
| | \ _____ +11101111 \ ...2/16
| | | \ _____ +11011111 \ ...3/16
| | | | \ _____ +11001111 \ ...4/16
| | | | | \ _____ +10111111 \ ...5/16 rrrrrrrr -- 6*LUM/16
| | | | | | \ _____ +10101111 \ ...6/16.... RRRRRRRR
| | | | | | | \ _____ +10011111 \ ...7/16
| | | | | | | | \ _____ +10001111 \ ...8/16
| | | | | | | | | \ _____ +01111111 \ ...9/16 rrrrrrrr --
11*LUM/16
| | | | | | | | | | \ _____ +01101111 \ ..10/16 RRRRRRRR
| | | | | | | | | | \ _____ +01011111 \ ..11/16.... RRRRRRRR
| | | | | | | | | | \ _____ +01001111 \ ..12/16
| | | | | | | | | | | \ _____ +00111111 \ ..13/16 RRRRRRRR
| | | | | | | | | | | | \ _____ +00101111 \ ..14/16 RRRRRRRR
| | | | | | | | | | | | | \ _____ +00011111 \ ..15/16 RRRRRRRR
| | | | | | | | | | | | | | \ _____ +00001111 \ ...full.... RRRRRRRR

```

bright-		television screen		The '15-switch'
significent				ness = the compliment
plus				of the most
ie,				nibble of ZPOS+ZOFF
				one divided by 16,

$$\frac{\begin{array}{l} \backslash | | \\ \backslash | \end{array}}{bbbb} = \frac{\begin{array}{l} | \\ ! \end{array}}{16} \quad /Z[7,4]+1$$

AN EXAMPLE OF '15-SWITCH' INTENSITY

Suppose that the FORMAT is returned to 0010000

& that the sprite data is left unchanged ----> 3333....

```
'15-switch' --+
```

```
& that it is laminated by an intensity sprite          v
that has a '15-switch' in the fifth pixel -----> 0000F000
```

```

RRRRRRRR ^
RRRRRRRR |
RRRRRRRR LUM stored in palette
RRRRRRRR | for color 'R'

```

Sprites scanned left to right ----->

```

      ZPOS+ZOFF      \      RRRRRRRR      Contrast increases
                        \      RRRRRRRR      as sprite is
moved      ._____ \      RRRRRRRR      toward the
screen.

```

```
| \          +1111%   \...full....`RRRRRRRR
| | \        +1110%   \..15/16      `
| | | \       +1101%   \..14/16      ` RRRr`rRR
| | | | \      +1100%   \..13/16     ` RRRRrRRR -- 11*LUM/16
| | | | | \     +1011%   \..12/16    ` RRRRRRRR
| | | | | | \    +1010%   \..11/16...`RRRRRRRR
| | | | | | | \   +1001%   \..10/16   `
| | | | | | | | \  +1000%   \...9/16   ` RRR ` RR
| | | | | | | | | \ +0111%   \...8/16   ` RRRR RRR
| | | | | | | | | | \ +0110%   \...7/16 ` RRRRrRRR --
```

```

6*LUM/16
|||||+0101%%%\...6/16....`RRRRRRRR
|||||+0100%%%\...5/16
\||||+0011%%%\...4/16`RRR`RR
\||||+0010%%%\...3/16`RRR`RR
\||||+0001%%%\...2/16`RRRR`RRR
\||||+0000%%%\...1/16....`RRRRrRRR --

```

LUM/16			
	\		\ \
	\		\ \
	\		Automatic
smoothing of			
	\		leading & lagging
pixels			
	\		done by the
hardware			

```

\||||||| television screen |
 \||||||| | The '15-switch'
  \||||||| | intensity = the most

```

\		significant nibble of
\		ZPOS+ZOFF plus one
\		divided by 16, ie,
\ _____	!	
		Z[7,4]+1
bbbb =	-----	
	16	

AN EXAMPLE OF COMBINED '15-SWITCH' BRIGHTNESS & INTENSITY

```

++++-- '15-switch'
||||
vvvv

```

Suppose that the FORMAT is returned to 0011111

& that the sprite data is left unchanged -----> 3333....

```

'15-switch' --+
|

```

& that it is laminated by the intensity sprite v
with the '15-switch' in the fifth pixel -----> 0000F000

```

RRRRRRRR ^
RRRRRRRR |
RRRRRRRR LUM stored in

```

palette

```

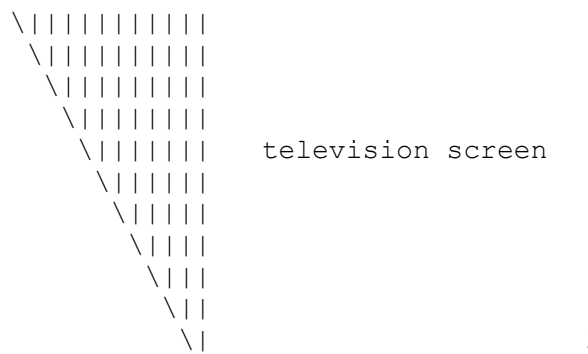
RRRRRRRR | for color sprite

```

'R'

Sprites scanned left to right ----->

	ZPOS+ZOFF	overall brightness	intensity	Both overall brightness
&	_____	_____	_____	_____
in-	\ _____ +1111%%%	_____ \ ...1/16....full....	`rrrrrrrr	contrast
the	\ _____ +1110%%%	_____ \ ...2/16...15/16		crease as
moves	\ _____ +1101%%%	_____ \ ...3/16...14/16		sprite
screen.	\ _____ +1100%%%	_____ \ ...4/16...13/16		toward
	\ _____ +1011%%%	_____ \ ...5/16...12/16	`rrrr rrr	
	\ _____ +1010%%%	_____ \ ...6/16...11/16....	`RRRRrRRR	
	\ _____ +1001%%%	_____ \ ...7/16...10/16		
	\ _____ +1000%%%	_____ \ ...8/16...9/16		
	\ _____ +0111%%%	_____ \ ...9/16...8/16	`rrr `rr	
	\ _____ +0110%%%	_____ \ ..10/16...7/16	`RRRR RRR	
	\ _____ +0101%%%	_____ \ ..11/16...6/16....	`RRRRrRRR	
	\ _____ +0100%%%	_____ \ ..12/16...5/16		
RR	\ \ _____ +0011%%%	_____ \ ..13/16...4/16	`RRR	
`RR	\ \ _____ +0010%%%	_____ \ ..14/16...3/16	`RRR	
RRR	\ \ _____ +0001%%%	_____ \ ..15/16...2/16	`RRRR	
	\ \ _____ +0000%%%	_____ \ ...full....1/16....	`RRRRrRRR	



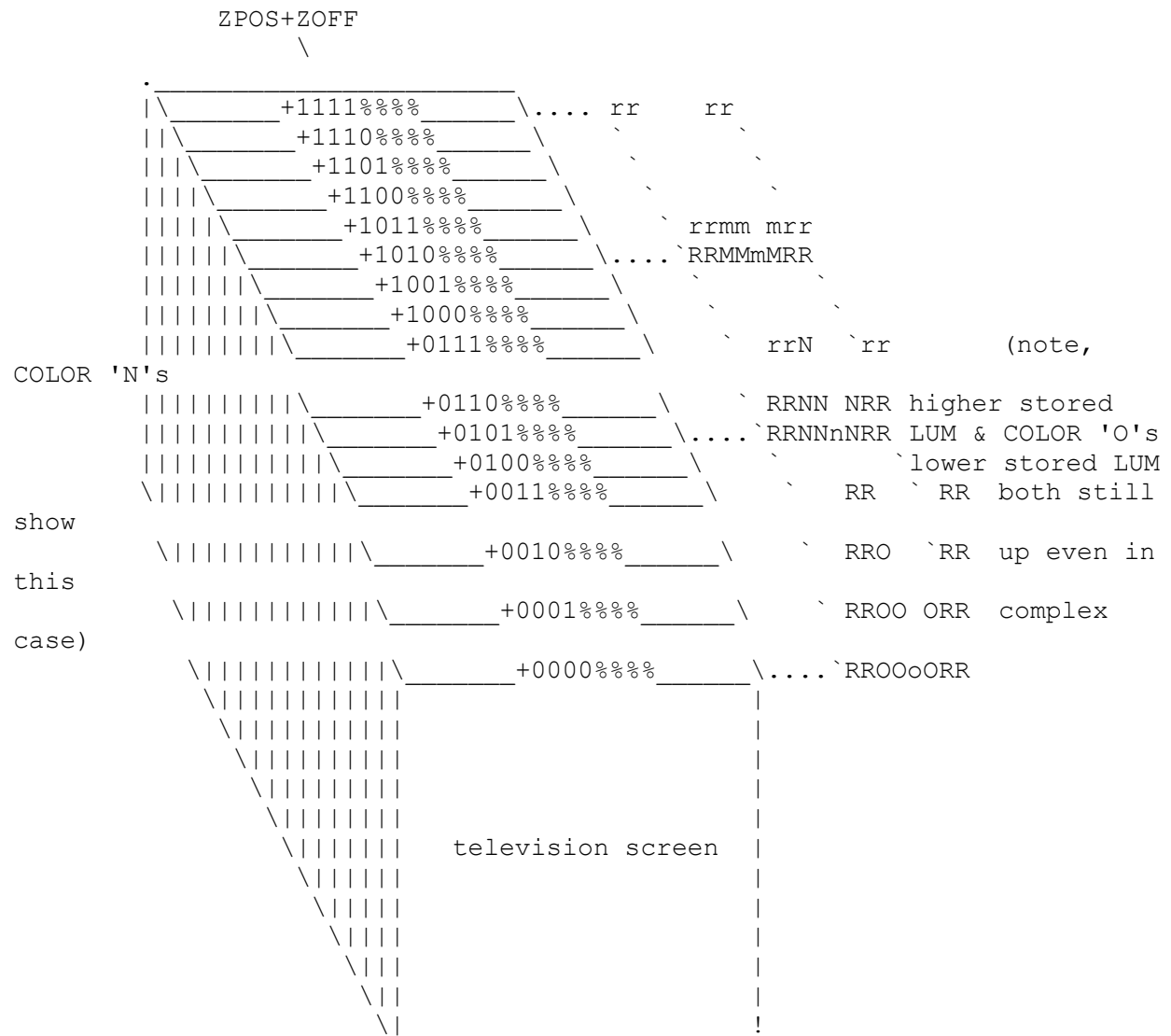
AN EXAMPLE OF '15-SWITCH' COLOR, BRIGHTNESS & INTENSITY

Finally, suppose that all the '15-switch's are set simultaneously

so that the FORMAT is set to

& the color sprite data is

& it is laminated by the intensity sprite with the '15-switch' in the fifth pixel

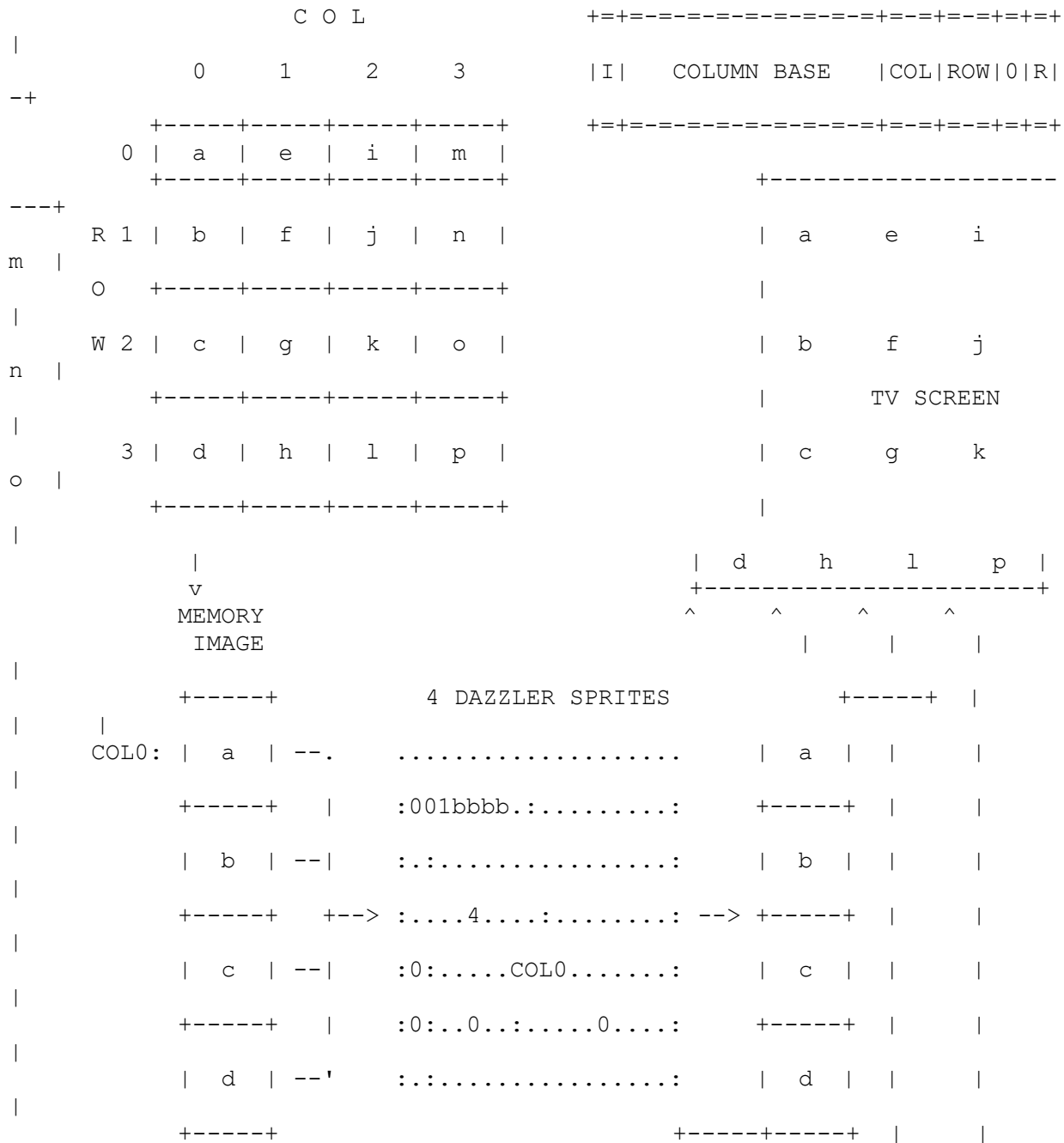


COLUMN BY ROW
BIT MAPPING TECHNIQUES

```

.....
:..FORMAT:....+YOFF...
:.....DISP.....
:....HGT.....+ZOFF..
:I:.....DATA.....: <-----+
:M:..MAP:....+XOFF...: |
:S:.....LINK.....: |

```



	COL1:	e		e
		+-----+	:001bbbb.:.....:	+-----+
		f	:::.....:	f
		+-----+	:....4....:.....: -->	+-----+
		g	:0:.....COL1.....:	g
		+-----+	:0:...0...:....16....: --16->+-----+	
		h	:::.....:	h
		+-----+		+-----+-----+
	COL2:	i		i
		+-----+	:001bbbb.:.....:	+-----+
		j	:::.....:	j
		+-----+	:....4....:.....: -->	+-----+
		k	:0:.....COL2.....:	k
		+-----+	:0:...0...:....32....: -----32----->+-----+	
		l	:::.....:	l
		+-----+		+-----+-----+
	COL3:	m		
m		+-----+	:001bbbb.:.....:	+-----+
-----+		n	:::.....:	
n		+-----+	:....4....:.....: -->	+-----+
-----+		o	:0:.....COL3.....:	
o		+-----+	:0:...0...:....48....: -----48----->+-----+	
-----+		p	:::.....:	
p		+-----+		+-----+

COLUMN BY ROW
BIT MAPPING TECHNIQUES

```

.....
:..FORMAT:....+YOFF...
:.....DISP.....
:....HGT.....+ZOFF..
:I:.....DATA.....: <-----+
:M:..MAP:....+XOFF...: |
:S:.....LINK.....: |

```

```

          C O L      +=+-----+=+-----+=+-----+=+-----+ |
          0      1      2      3      | I |   COLUMN BASE   | ROW | COL | 0 | R | -
+
      +-----+-----+-----+-----+      +=+-----+=+-----+=+-----+=+-----+
      0 | a | b | c | d |
      +-----+-----+-----+-----+
----+
R 1 | e | f | g | h |
d |
O  +-----+-----+-----+-----+
|
W 2 | i | j | k | l |
h |
    +-----+-----+-----+-----+
    | m | n | o | p |
1 |
3 +-----+-----+-----+-----+
|
    |
    v
    MEMORY
    IMAGE
|
    +-----+
    |
    COL0: | a | --. ....
|
    +-----+ | :001bbbb.:.....:
|
    COL1: | b | | :.:.....:
|
    +-----+ +--> :....4....:.....: --> +-----+ |
|
    COL2: | c | | :0:.....COL0.....:
|
    +-----+ | :0:.1F.:.....0....:
|
    COL3: | d | | :.:.....:
|
    +-----+ |
    | e | --| .....
|
          | m      n      o      p |
          +-----+-----+-----+
          ^      ^      ^      ^
          |      |      |      |
          +-----+
          4 DAZZLER SPRITES
          +-----+ |
          |
          COL0: | a | --. ....
          |
          +-----+ | :001bbbb.:.....:
          |
          COL1: | b | | :.:.....:
          |
          +-----+ +--> :....4....:.....: --> +-----+ |
          |
          COL2: | c | | :0:.....COL0.....:
          |
          +-----+ | :0:.1F.:.....0....:
          |
          COL3: | d | | :.:.....:
          |
          +-----+ |
          | e | --| .....
          |
          +-----+-----+-----+ |
          | b | |

```

	+-----+		:001bbbb.:.....:		+-----+	
	f		:.....:		f	
	+-----+		:....4.....:.....: -->		+-----+	
	g		:0:.....COL1.....:		j	
	+-----+		:0:.1F.....16.....: --16->+-----+			
	h		:.....:		n	
	+-----+				+-----+-----+	
	i --				c	
	+-----+		:001bbbb.:.....:		+-----+	
	j		:.....:		g	
	+-----+		:....4.....:.....: -->		+-----+	
	k		:0:.....COL2.....:		k	
	+-----+		:0:.1F.....32.....: -----32----->+-----+			
	l		:.....:		o	
	+-----+				+-----+-----+	
d	m --'					
-----+	+-----+		:001bbbb.:.....:		+-----+	
h	n		:.....:			
-----+	+-----+		:....4.....:.....: -->		+-----+	
l	o		:0:.....COL3.....:			
-----+	+-----+		:0:.1F.....48.....: -----48----->+-----+			
p	p		:.....:			
	+-----+				+-----+	

