# Transformation Engine and Cursor Reference Manual

Curtis Priem

SUN Microsystems Inc.

2550 Garcia Avenue

Mountain View, CA 94030

Version 3.0

January 21, 1988

CONTENTS

| Address | Name | Description |
|---|---|---|
| | 31 0 | INDEXED ADDRESS REGISTERS |
| tec+0xac0 | BQUADABSX | X for Fixed Point Abs Quad |
| tec+0xac4 | BQUADABSY | Y for Fixed Point Abs Quad |
| tec+0xac8 | BQUADABSZ | Z for Fixed Point Abs Quad |
| tec+0xacc | BQUADABSW | W for Fixed Point Abs Quad |
| tec+0xad0 | BQUADRELX | X for Fixed Point Rel Quad |
| tec+0xad4 | BQUADRELY | Y for Fixed Point Rel Quad |
| tec+0xad8 | BQUADRELZ | Z for Fixed Point Rel Quad |
| tec+0xadc | BQUADRELW | W for Fixed Point Rel Quad |
| tec+0xb00 | BRECTABSX | X for Fixed Point Abs Rectangle |
| tec+0xb04 | BRECTABSY | Y for Fixed Point Abs Rectangle |
| tec+0xb08 | BRECTABSZ | Z for Fixed Point Abs Rectangle |
| tec+0xb0c | BRECTABSW | W for Fixed Point Abs Rectangle |
| tec+0xb10 | BRECTRELX | X for Fixed Point Rel Rectangle |
| tec+0xb14 | BRECTRELY | Y for Fixed Point Rel Rectangle |
| tec+0xb18 | BRECTRELZ | Z for Fixed Point Rel Rectangle |
| tec+0xb1c | BRECTRELW | W for Fixed Point Rel Rectangle |

| Address | Name | Description |
|---|---|---|
| | 31 0 | INDEXED ADDRESS REGISTERS |
| tec+0xcc0 | FQUADABSX | X for Floating Point Abs Quad |
| tec+0xcc4 | FQUADABSY | Y for Floating Point Abs Quad |
| tec+0xcc8 | FQUADABSZ | Z for Floating Point Abs Quad |
| tec+0xccc | FQUADABSW | W for Floating Point Abs Quad |
| tec+0xcd0 | FQUADRELX | X for Floating Point Rel Quad |
| tec+0xcd4 | FQUADRELY | Y for Floating Point Rel Quad |
| tec+0xcd8 | FQUADRELZ | Z for Floating Point Rel Quad |
| tec+0xcdc | FQUADRELW | W for Floating Point Rel Quad |
| tec+0xd00 | FRECTABSX | X for Floating Point Abs Rect |
| tec+0xd04 | FRECTABSY | Y for Floating Point Abs Rect |
| tec+0xd08 | FRECTABSZ | Z for Floating Point Abs Rect |
| tec+0xd0c | FRECTABSW | W for Floating Point Abs Rect |
| tec+0xd10 | FRECTRELX | X for Floating Point Rel Rect |
| tec+0xd14 | FRECTRELY | Y for Floating Point Rel Rect |
| tec+0xd18 | FRECTRELZ | Z for Floating Point Rel Rect |
| tec+0xd1c | FRECTRELW | W for Floating Point Rel Rect |

# 3 - TRANSFORMATION REGISTERS

The TEC allows the software to off load all 2D and 3D signed integer, fixed point, and floating point transformations including scaling, rotation, translation, etc. The TEC will also has a general command register that allows it to do dot products such as matrix concatenation, implicitization, and subdivision.

The TEC consists of 64 36 bit data registers, a multiplier, and an adder. A coordinate can be transformed and loaded into the FBC's indexed address registers by writing the coordinate to the TEC's indexed address registers. The CPU can also do general matrix multiplications by writing to the COMMAND register.

When accessing the indexed address registers, the TEC uses the general transformation pipeline shown in Figure 3-1.

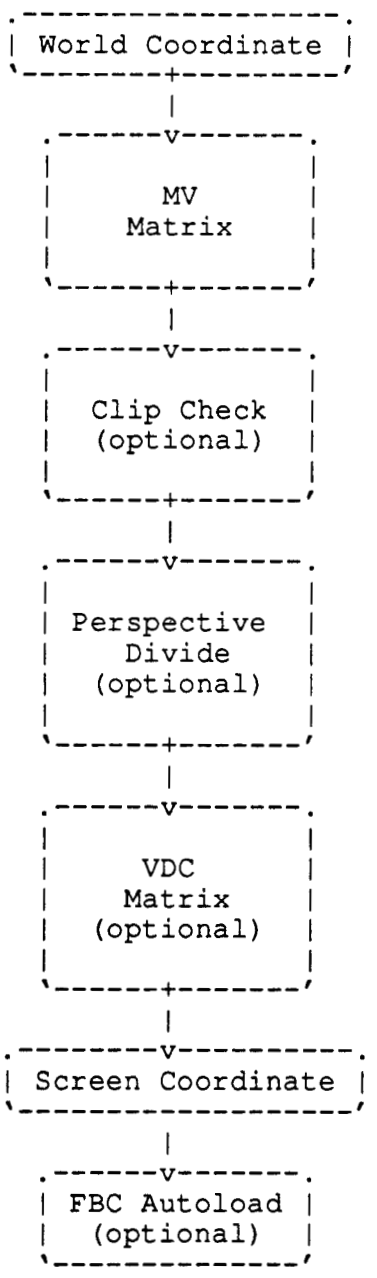
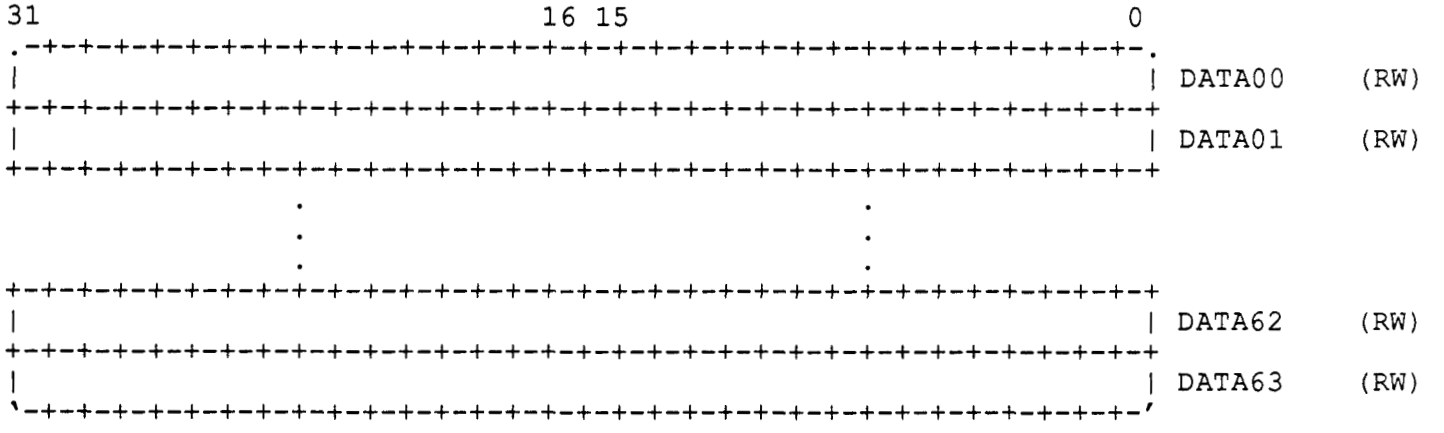



Figure 3-1 TEC Transformation Pipeline

## 3.1 DATA REGISTERS

The TEC contains 64 general purpose 36 bit data registers. The data is stored in a format that is not accessable by the user. There are three different formats supported for accessing the data registers: signed integer, signed fixed point (16.16), and IEEE floating point (single precision). The formats can be selected by setting the data type in the VDC_MATRIX register. NOTE: The internal format and operations are not IEEE floating point even though the interface is. If results are read back, they may differ from what the IEEE standard may produce. The TEC should be used only for "end of pipeline operations, and not used for iterative operations since accumulation of errors differ from IEEE.

All of the transformation data registers can be read and written for context switching by setting the data type in the MV_MATRIX register to internal formats 0 and 1 (INTERNAL0, INTERNAL1). Matrix multiplies should not be done while the data type is set to an internal format.




| Signed Integer: | | Signed Fixed Point: | | IEEE Floating Point: | |
|---|---|---|---|---|---|
| Bits | Description | Bits | Description | Bits | Description |
| 31 | Sign Bit | 31 | Sign Bit | 31 | Sign Bit |
| 30-0 | Integer | 30-16 | Integer | 30-23 | Exponent |
| | | 15-0 | Fraction | 22-0 | Fraction |

Figure 3-2 Transform Data Registers

## 3.2 MV MATRIX FORMAT REGISTER

The first matrix (MV) transformation which is done is a concatenation of the modeling and viewing transformation.

The type of the point that the MV matrix is to be applied to can be either be integer, fixed point, or floating point. This is determined by which of the index registers the X coordinate is written to. The X, Y, Z, and W coordinates of the point are loaded into data registers DATA00 thru DATA03 respectively. The size of the point can be 2D (X Y 0 1), 3D (X Y Z 1), 2H (X Y 0 W), or 3H (X Y Z W). The TEC automatically keeps track of the size of the point by recording if the Z or W values have been entered. The TEC also keeps track of the sign of W. The size and the sign are cleared whenever an X value is entered. The X value must always be entered.

The MV_MATRIX register contains a pointer that points to the first element of the MV matrix. The elements in the MV matrix are consecutive data in the register file organized by column. A 3x2 2D matrix (i=0, j=1) would take up 6 consecutive locations in the register file with its order being A, B, C, D, E, and F. See Figure 3-3.

```
A 0 0 0   A D 0 0   A D G 0   A D G J
B 0 0 0   B E 0 0   B E H 0   B E H K
0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
C 0 0 0   C F 0 0   C F I 0   C F I L

i=0 j=0   i=0 j=1   i=0 j=2   i=0 j=3


A 0 0 0   A E 0 0   A E I 0   A E I M
B 0 0 0   B F 0 0   B F J 0   B F J N
C 0 0 0   C G 0 0   C G K 0   C G K O
D 0 0 0   D H 0 0   D H L 0   D H L P

i=1 j=0   i=1 j=1   i=1 j=2   i=1 j=3
```

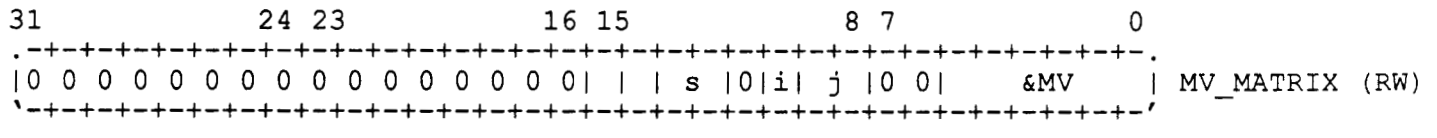
Figure 3-3 Elements used in MV matrix

The x, y, z, w outputs will be placed in data registers DATA04 thru DATA07 respectively.

If the input point is relative, the result of the previous MV matrix multiply (DATA04 thru DATA07) is used in the current MV matrix multiply instead of the translation elements (row 3) of the MV matrix.

The TEC can automatically load the FBC with the results of the MV matrix transformation by setting the AUTOLOAD_EN bit and not setting any clip check, divide, or VDC enable bits. The TEC will autoload the z coordinate if j >= 2, followed by the y coordinate if j >= 1, and then the x coordinate. The MV output type will be integer for autoload.

The MV_MATRIX register can be read and written for context switching.




| Bits | Description | |
|---|---|---|
| 15 | DIVIDE_EN | (1=Enable Perspective Divide) |
| 14 | AUTOLOAD_EN | (0=Don't load FBC, 1=Load Results into FBC) |
| 13-12 | Point Size | (0=2D, 1=3D, 2=2H, 3=3H) |
| 10 | i | (0=3, 1=4) |
| 9-8 | j | (0=1, 1=2, 2=3, 3=4) |
| 5-0 | Pointer to MV | (0=DATA00, 1=DATA01, ... 63=DATA63) |

Figure 3-4 MV Matrix Format Register

## 3.3 CLIP CHECKING FORMAT REGISTER

The TEC does clip checking (not 3D clipping) against the perspective viewing pyramid. If the output of MV matrix is homogeneous (j = 3) and the W coordinate of the point entered is positive, the following comparisons are done:

$$-w <= x <= w, \quad -w <= y <= w, \quad 0 <= z <= w \qquad (\text{if } W >= 0)$$

If the W coordinate of the point entered is negative, the following comparisons are done:

$$-w => x => w, \quad -w => y => w, \quad 0 => z => w \qquad (\text{if } W < 0)$$

where x, y, z, and w are the output of the MV matrix multiplication.

If the output of MV matrix is not homogeneous (j != 3) then the equations will be different. If the input is homogeneous the comparisons will be:

$$-W <= x <= W, \quad -W <= y <= W, \quad 0 <= z <= W \qquad (\text{if } W >= 0)$$

$$-W => x => W, \quad -W => y => W, \quad 0 => z => W \qquad (\text{if } W < 0)$$

If the input is not homogeneous, the comparisons will be:

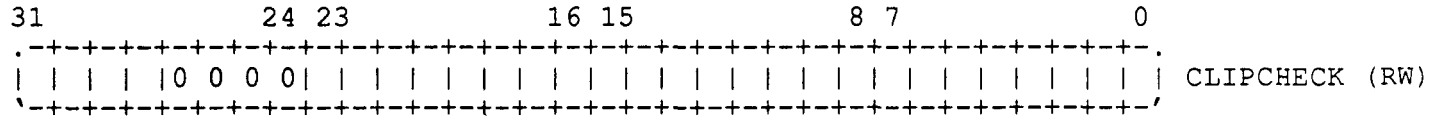
$$-1 <= x <= 1, \quad -1 <= y <= 1, \quad 0 <= z <= 1 \qquad (\text{if } 1 >= 0)$$

Each of the checks can be individually enabled or disabled. The TEC does the x comparison if j >= 0, the y comparison if j >= 1, and the z comparison if j >= 2. If the clipping is to be done to something other than the unit perspective viewing pyramid, the MV matrix must normalize to a unit pyramid and the VDC matrix must do the inverse. All clipping can be enabled or disabled by the CLIP_ENABLE bit.

The CLIPCHECK register accumulates if the points that have been entered (since the last time the CLIPCHECK register was cleared) are less than (LT), greater than (GT), inside the front (INSIDE) or back (BACKSIDE) viewing pyramid. This status is combined to tell if the object is totally hidden, intersected, or visible.

The TEC overflows from multiplies, adds, divide_by_zeros, input and output conversions are accumulated in the TEC_EXCEPTION bit of the CLIPCHECK register.

The CLIPCHECK register can be read and written for context switching.




| Bits | Description | | |
|---|---|---|---|
| 31 | TEC_EXCEPTION | (1=TEC has an exception) | |
| 30 | HIDDEN | (1=Object is Hidden) | (RO) |
| 29 | INTERSECT | (1=Object is Intersecting the Pyramid) | (RO) |
| 28 | VISIBLE | (1=Object is Visible) | (RO) |
| 24 | CLIP_ENABLE | (0=Disable, 1=Enable) | |
| 23 | ACC_Z_BACKSIDE | (\|= Z_LT_FRONT && Z_GT_BACK) | |
| 22 | ACC_Z_LT_FRONT | (\|= Z_LT_FRONT && !Z_GT_BACK) | |
| 21 | ACC_Z_INSIDE | (\|= !Z_LT_FRONT && !Z_GT_BACK) | |
| 20 | ACC_Z_GT_BACK | (\|= !Z_LT_FRONT && Z_GT_BACK) | |
| 19 | Z_LT_FRONT | (w>=0 && (z<0 && CLIP_FRONT) \|\| w<0 && (z<=0 \|\| !CLIP_FRONT)) | |
| 18 | Z_GT_BACK | (w>=0 && (z>w && CLIP_BACK) \|\| w<0 && (z>=w \|\| !CLIP_BACK)) | |
| 17 | CLIP_FRONT | (0=Disabled, 1=Enabled) | |
| 16 | CLIP_BACK | (0=Disabled, 1=Enabled) | |
| 15 | ACC_Y_BACKSIDE | (\|= Y_LT_BOTTOM && Y_GT_TOP) | |
| 14 | ACC_Y_LT_BOTTOM | (\|= Y_LT_BOTTOM && !Y_GT_TOP) | |
| 13 | ACC_Y_INSIDE | (\|= !Y_LT_BOTTOM && !Y_GT_TOP) | |
| 12 | ACC_Y_GT_TOP | (\|= !Y_LT_BOTTOM && Y_GT_TOP) | |
| 11 | Y_LT_BOTTOM | (w>=0 && (y<-w && CLIP_BOTTOM) \|\| w<0 && (y<=-w \|\| !CLIP_BOTTOM)) | |
| 10 | Y_GT_TOP | (w>=0 && (y>w && CLIP_TOP) \|\| w<0 && (y>=w \|\| !CLIP_TOP)) | |
| 9 | CLIP_BOTTOM | (0=Disabled, 1=Enabled) | |
| 8 | CLIP_TOP | (0=Disabled, 1=Enabled) | |
| 7 | ACC_X_BACKSIDE | (\|= X_LT_LEFT && X_GT_RIGHT) | |
| 6 | ACC_X_LT_LEFT | (\|= X_LT_LEFT && !X_GT_RIGHT) | |
| 5 | ACC_X_INSIDE | (\|= !X_LT_LEFT && !X_GT_RIGHT) | |
| 4 | ACC_X_GT_RIGHT | (\|= !X_LT_LEFT && X_GT_RIGHT) | |
| 3 | X_LT_LEFT | (w>=0 && (x<-w && CLIP_LEFT) \|\| w<0 && (x<=-w \|\| !CLIP_LEFT)) | |
| 2 | X_GT_RIGHT | (w>=0 && (x>w && CLIP_RIGHT) \|\| w<0 && (x>=w \|\| !CLIP_RIGHT)) | |
| 1 | CLIP_LEFT | (0=Disabled, 1=Enabled) | |
| 0 | CLIP_RIGHT | (0=Disabled, 1=Enabled) | |

Figure 3-5 Clip Checking Format Register

## 3.5 VDC MATRIX FORMAT REGISTER

The second matrix (VDC) transformation which is done is the virtual device coordinate transformation. The VDC_MATRIX register contains a pointer that points to the first element of the VDC matrix. The elements in the VDC matrix are consecutive data in the register file. The VDC matrix can only scale and translate the point to which it is applied. See Figure 3-6 for the configuration of the VDC matrix.

```
A 0 0 0    A 0 0 0
0 C 0 0    0 C 0 0
0 0 0 0    0 0 E 0
B D 0 0    B D F 0

  k=0        k=1
```

Figure 3-6 Elements used in VDC matrix

The output will be placed in data registers DATA08 thru DATA10.

The TEC can automatically load the FBC with the results of this transform by setting the AUTOLOAD_EN bit in the MV_MATRIX register. The TEC will autoload the z coordinate if k = 1, followed by the y coordinate, and then the x coordinate. The output of the MV matrix should be 3D (j >= 2) if the size of the VDC matrix is 3D (k = 1), and 2D (j >=1) if the size is 2D (k = 0). If the AUTOLOAD_EN is selected, the VDC output type will be set to integer. The VDC matrix can be enabled or disabled with the VDC_EN bit (the VDC is assumed to be enabled if the perspective divide is enabled).

The VDC_MATRIX register can be read and written for context switching.

```
31            24 23           16 15            8 7             0
.-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-.
|0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0|     | |0 0 0|k|0 0|    &VDC    | VDC_MATRIX(RW)
`-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-'
```

| Bits | Description | |
|---|---|---|
| 15-13 | Data Type | (000=Signed Integer, 001=Fixed Point, 010=Floating Point, 011=Illegal 100=INTERNAL0, 101=INTERNAL1, 110=Illegal, 111=Illegal) |
| 12 | VDC_EN | (1=Enable VDC Matrix Multiply) |
| 8 | k | (0=2D, 1=3D) |
| 5-0 | Pointer to VDC | (0=DATA00, 1=DATA01, ... 63=DATA63) |

Figure 3-7 VDC Matrix Format Register

## 4 - INDEXED ADDRESS REGISTERS

The TEC has a set of indexed address registers that duplicate the FBC's indexed address registers. Whenever the CPU writes to one of the TEC's indexed address X registers, the TEC will transform the point and automatically load (if AUTOLOAD is set) the integer result into the corresponding indexed address register in the FBC.

The coordinate type of the indexed address registers can be integer, fixed point, or floating point. The indexed address registers do not actually store the X, Y, Z, and W coordinates written to them but route them to the data registers DATA00 thru DATA03 respectively.

All the indexed address registers are write only. They do not have to be saved or restored during context switching.

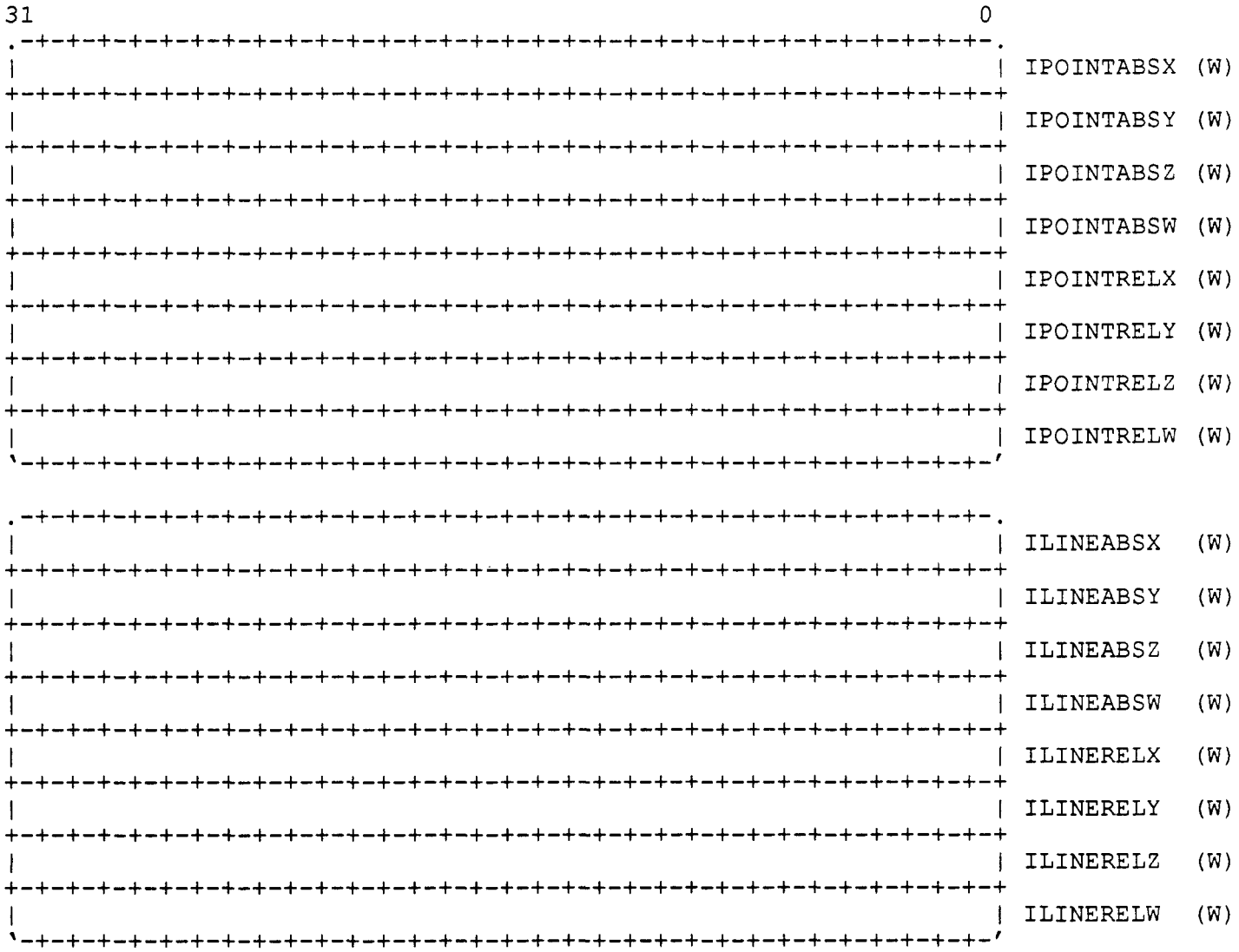



Figure 4-1 Indexed Address Registers (Signed Integer)

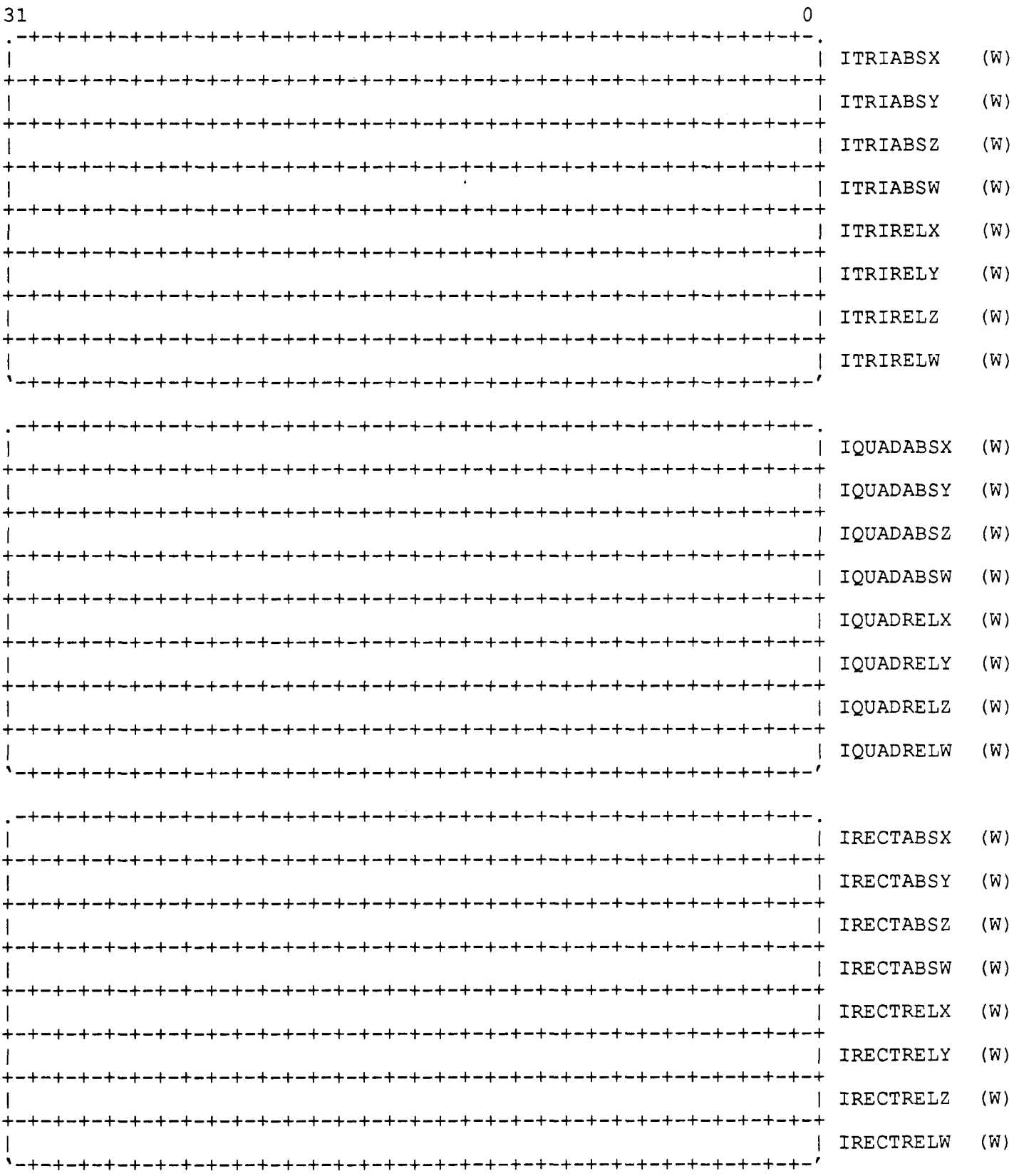



Figure 4-2 Indexed Address Registers (Signed Integer)

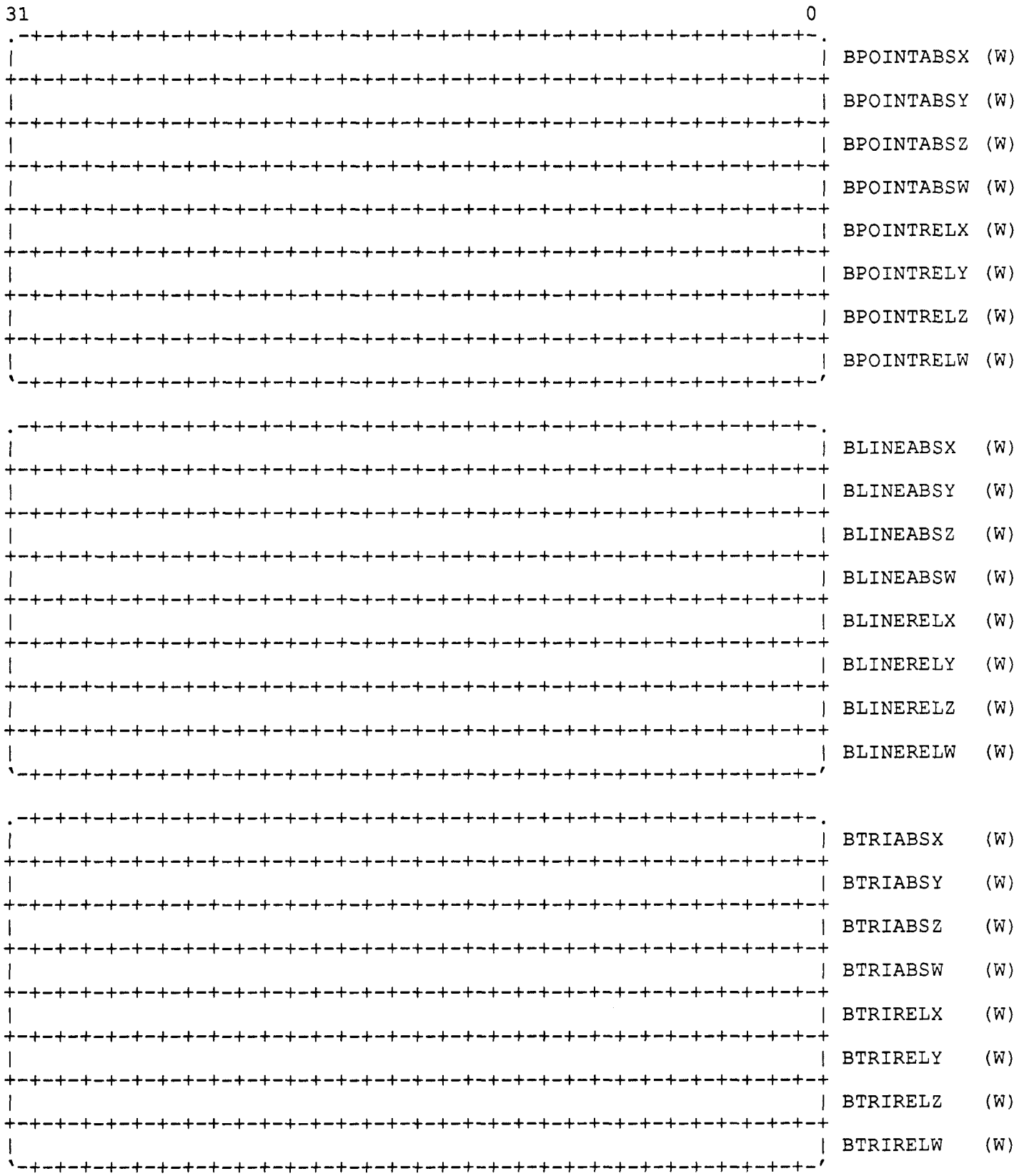



Figure 4-3 Indexed Address Registers (Fixed Point)

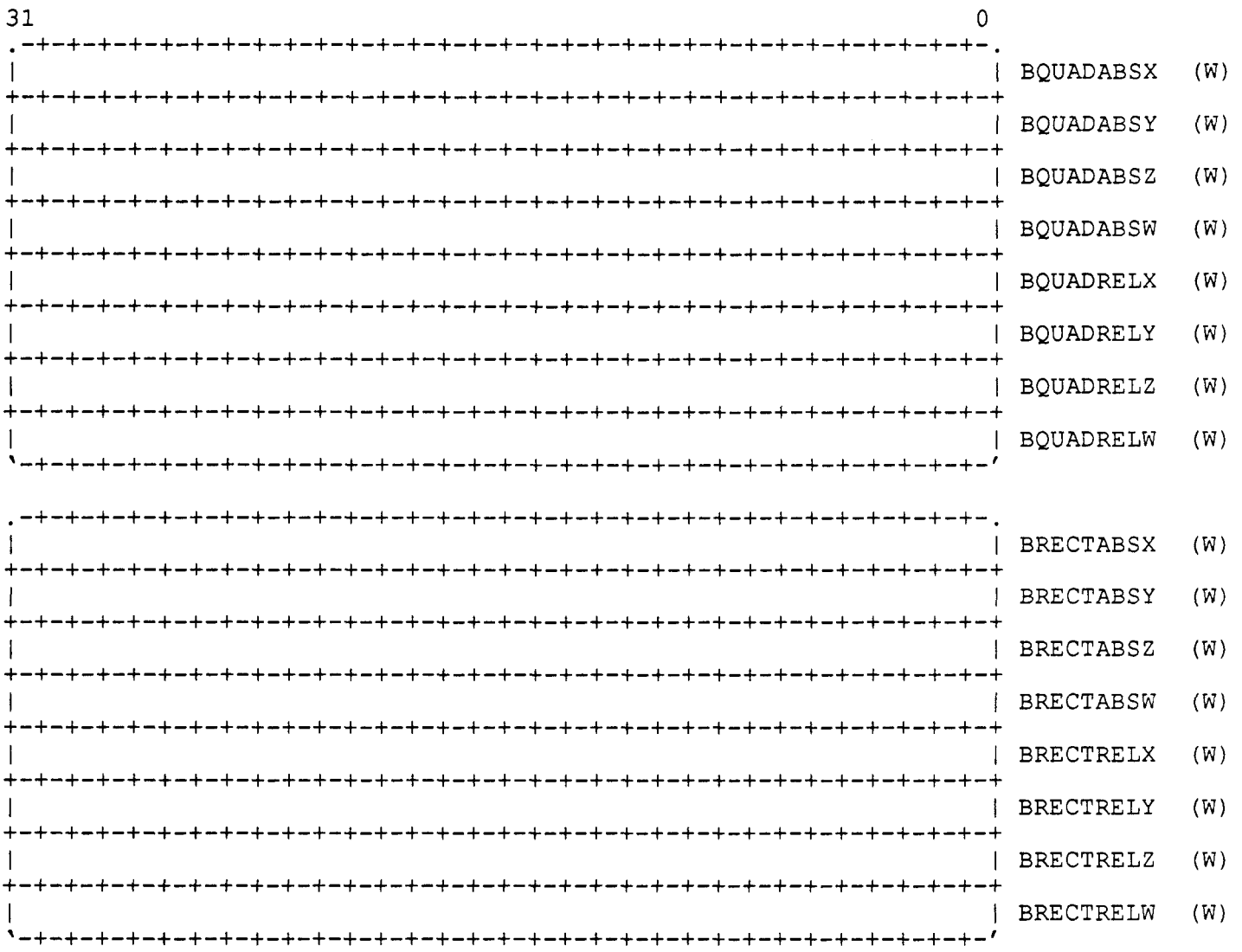



Figure 4-4 Indexed Address Registers (Fixed Point)

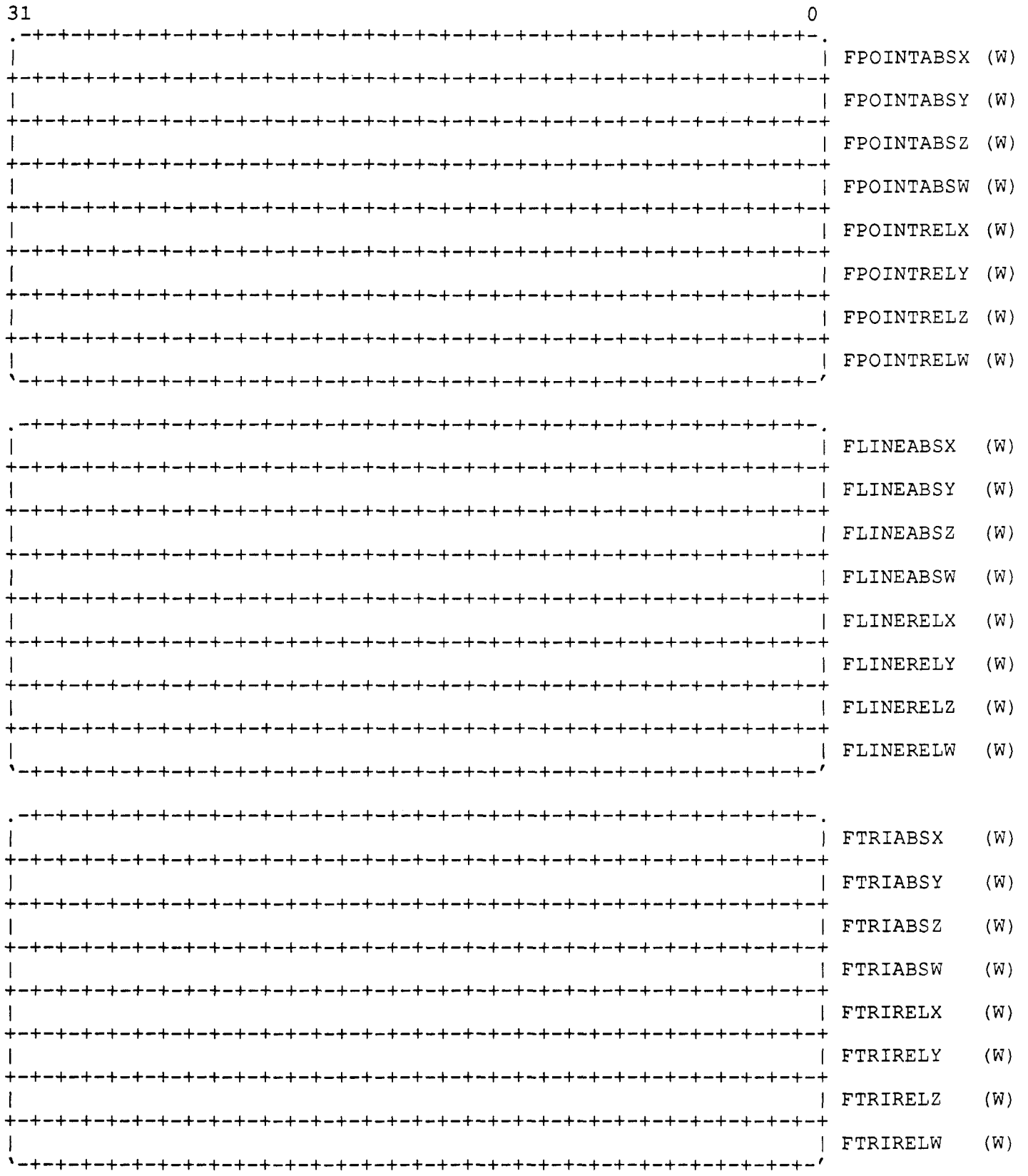



Figure 4-5 Indexed Address Registers (Floating Point)

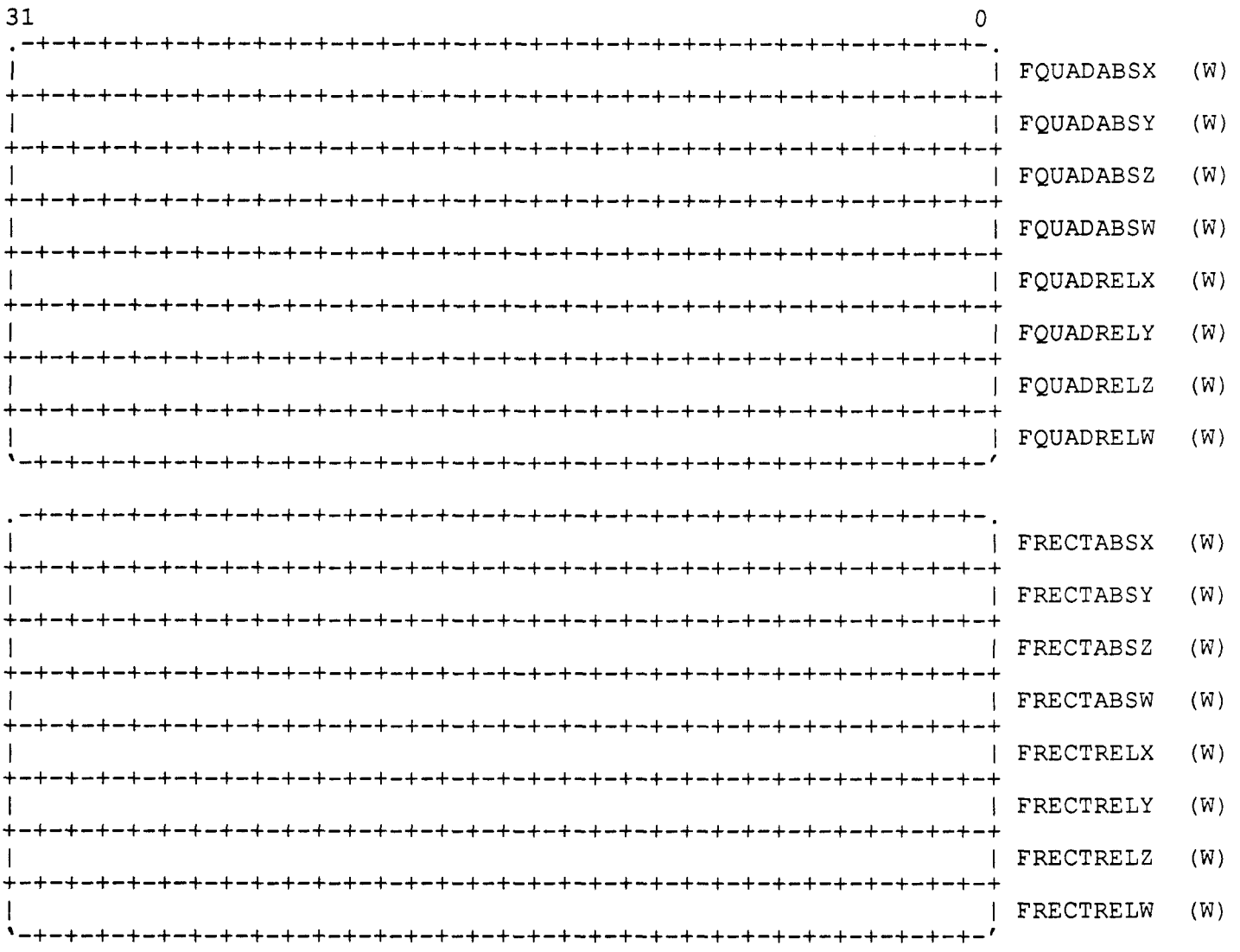



Figure 4-6 Indexed Address Registers (Floating Point)

5 - COMMAND REGISTERS

The TEC executes the following general equation in Figure 5-1 when a command is written to one of the COMMAND registers.

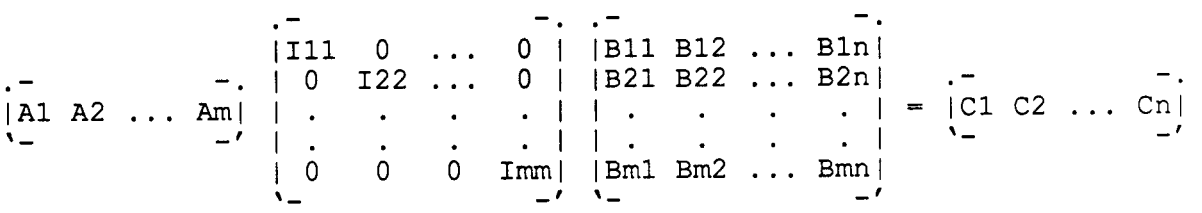
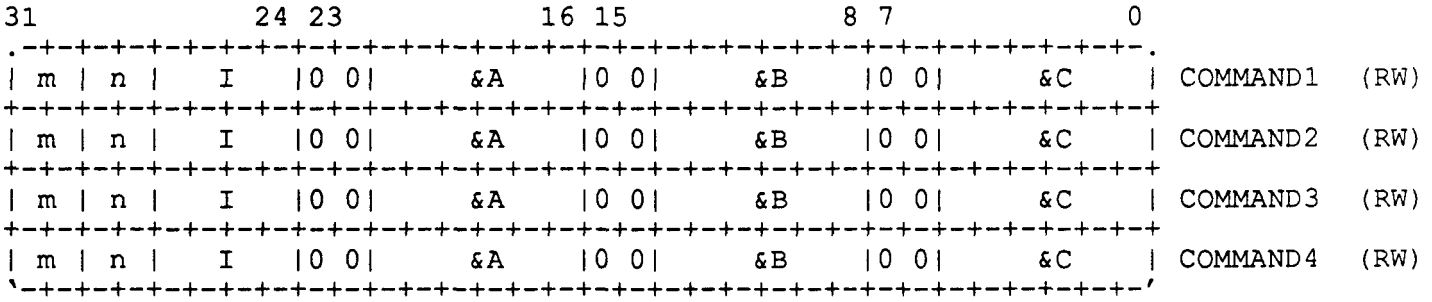



Figure 5-1 Matrix Equation

Matrix A is 1 by m, matrix I is m by m, matrix B is m by n, and matrix C is 1 by n, where m and n are 1 to 4 inclusive. Matrix I is an identity-like matrix where the diagonal elements are 1 or -1. The elements in the B matrix are consecutive data in the register file organized by column. A 2D matrix (3x2) would take up 6 consecutive locations in the register file with its order being B11, B21, B31, B12, B22, B32. The elements in the A and C matrices have data in the register file that is organized by a constant offset depending on which COMMAND register is written to. The offset is one, two, three, and four for command regiters COMMAND1, COMMAND2, COMMAND3, and COMMAND4, respectively. The elements in the A matrix [A1 A2 A3 A4], when the command is written to COMMAND3, would be located in the register file [&A &A+3 &A+6 &A+9].

The command register can be read and written. The data read will be the last data written to any of the command regiters. It does not have to be saved or restored during context switching.




| Bits | Description | |
|---|---|---|
| 31-30 | m | (0=1, 1=2, 2=3, 3=4) |
| 29-28 | n | (0=1, 1=2, 2=3, 3=4) |
| 27 | Element I11 | (0=1, 1=-1) |
| 26 | Element I22 | (0=1, 1=-1) |
| 25 | Element I33 | (0=1, 1=-1) |
| 24 | Element I44 | (0=1, 1=-1) |
| 21-16 | Pointer to A | (0=DATA00, 1=DATA01, ... 63=DATA63) |
| 13-8 | Pointer to B | (0=DATA00, 1=DATA01, ... 63=DATA63) |
| 5-0 | Pointer to C | (0=DATA00, 1=DATA01, ... 63=DATA63) |

Figure 5-2 Transform Command Registers

# 6 - HARDWARE CURSOR

The TEC contains a two plane 32x32 pixel programmable hardware cursor. The two planes define three possible colors of the cursor and when the cursor should be transparent (ie. allow the 8 color planes to be displayed). The color of the cursor is programmed in the LUT/DAC, see Figure 6-1. The cursor outputs of the TEC are connected to the overlay inputs of the LUT/DAC.

| Plane A | Plane B | Cursor Color |
|---|---|---|
| 0 | 0 | Color Palette RAM |
| 0 | 1 | Overlay Color 1 |
| 1 | 0 | Overlay Color 2 |
| 1 | 1 | Overlay Color 3 |

Figure 6-1 Cursor Colors

The location of where the cursor is to be placed on the screen is written into the cursor ADDRESS register. The TEC automatically aligns the cursor to the correct pixel boundary and shifts it to the LUT/DAC at the proper time. If the cursor is not being displayed, the TEC will output zeros to force the LUT/DAC to use the Color Palette RAM.

The TEC can be configured either for 4 pixel multiplexing (1024x768 and 1152x900) or 8 pixel multiplexing (1600x1280).

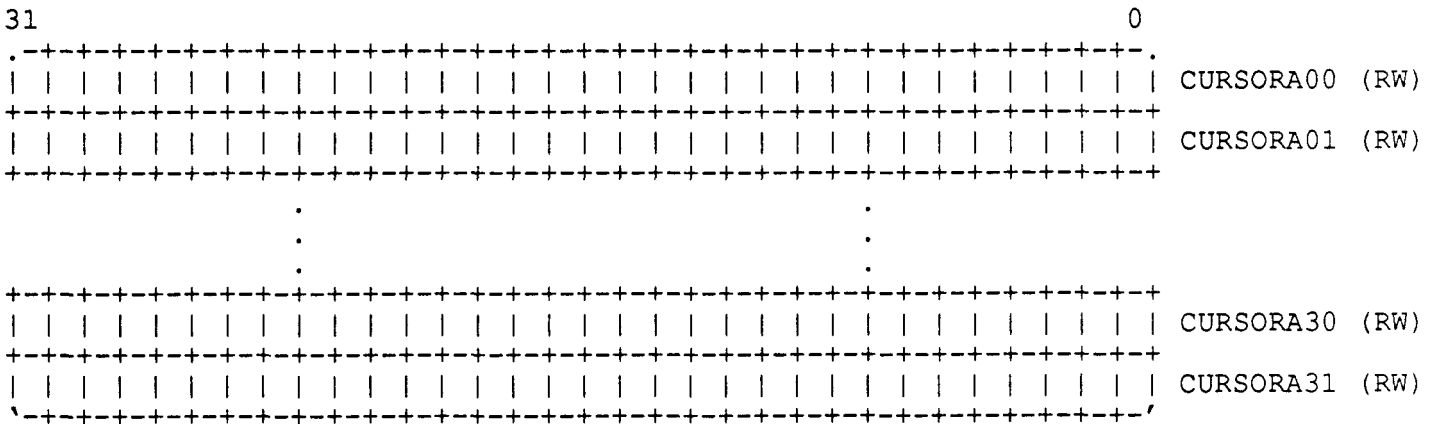
The TEC can not generate a cross-hair cursor.

The hardware cursor registers reside in a different page from the user registers so that they can be protected by the MMU.

## 6.1 CURSOR DATA REGISTER

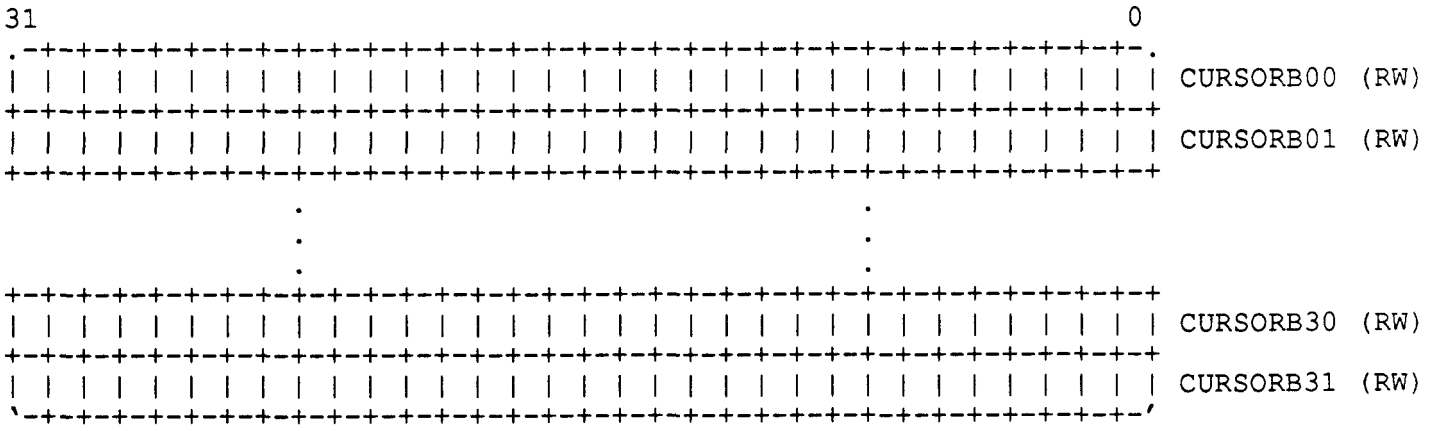
The 64 32 bit cursor data registers contain the bit mapped description of the 2 plane 32x32 cursor. The CURSORA registers contain the horizontal lines of plane A where CURSORA00 is the top line (See Figure 6-2). The CURSORB registers contain the horizontal lines of plane B where CURSORB00 is the top line (See Figure 6-3).

The cursor data and pointer registers can be read and written to for diagnostics.




| Bits | Description |
|---|---|
| 31 | Left-most Pixel of Cursor |
| . | . |
| 0 | Right-most Pixel of Cursor |

Figure 6-2 Cursor Data Register (Plane A)




| Bits | Description |
|---|---|
| 31 | Left-most Pixel of Cursor |
| . | . |
| 0 | Right-most Pixel of Cursor |

Figure 6-3 Cursor Data Register (Plane B)

## 6.2 CURSOR ADDRESS REGISTERS

The cursor ADDRESS register contains the XY coordinate of where the upper left corner of the 32x32 hardware cursor will be placed on the screen. The cursor can be moved around the screen by loading a new address in the ADDRESS register. The coordinate system of the register is the same as the screen coordinate system where (0,0) is the upper left corner of the screen. This allows the cursor to be partially displayed if small negative values are written to the ADDRESS register or values just under the horizontal or vertical resolution of the screen. The equation for setting the cursor address is:

```
ADDRESS = ((X - x_hot_point) << 16)  |  ((Y - y_hot_point) & 0xffff);
```

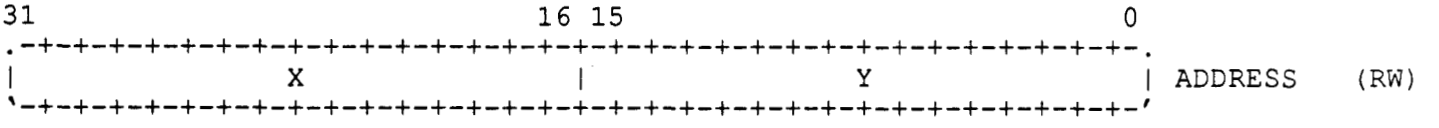
where X and Y are screen coordinates and x_hot_point and y_hot_point are the offset from the upper left corner of the cursor.

The ADDRESS register is read by the TEC's cursor control logic only at the beginning of every vertical blanking period. This prevents tearing of the cursor when it is being moved around the screen.

The address register contains both the X and Y coordinate so that the updating of the address is a single operation. This prevents the cursor from moving to an intermediate position if the X is load before vertical blanking and the Y is loaded after.

When changing the cursor to a different shape, tearing of the cursor will only happen 0.1 to 0.3 % of the time. If no tearing is required, 0xffe0ffe0 should be written to the ADDRESS register and the vertical interrupt set. The interrupt handling routine should change the cursor and update the ADDRESS register. This method prevents tearing but removes the cursor from the screen for one frame and is slower.

The cursor ADDRESS register can be read and written for diagnostics.




| Bits | Description |
|---|---|
| 31-16 | X coordinate of Cursor Address |
| 15-0 | Y coordinate of Cursor Address |

Figure 6-4 Cursor Address Register

# 7 - HARDWARE CONFIGURATION REGISTERS

The hardware configuration registers are programmed with system specific information. This information is loaded in to these registers by the boot proms during diagnostics, and are not changed during the normal operation of the workstation. These registers reside in a different page from the user registers so that they can be protected by the MMU.

The data in the hardware configuration registers does not have to be saved or restored for context switching, since the information in these registers is not context dependent.

See the TEC Hardware Manual for more information on these registers.

## 7.1 TIMING REGISTERS

The hardware configuration registers are used to generate the correct timing for the sync and blank signals which control the monitor.

The TEC is capable of supporting a variety of monitors since the timing parameters are programmable. There will be four monitors that will be fully supported; a color and grayscale 66 Hz for the 1152x900 and 1600x1280 resolutions.

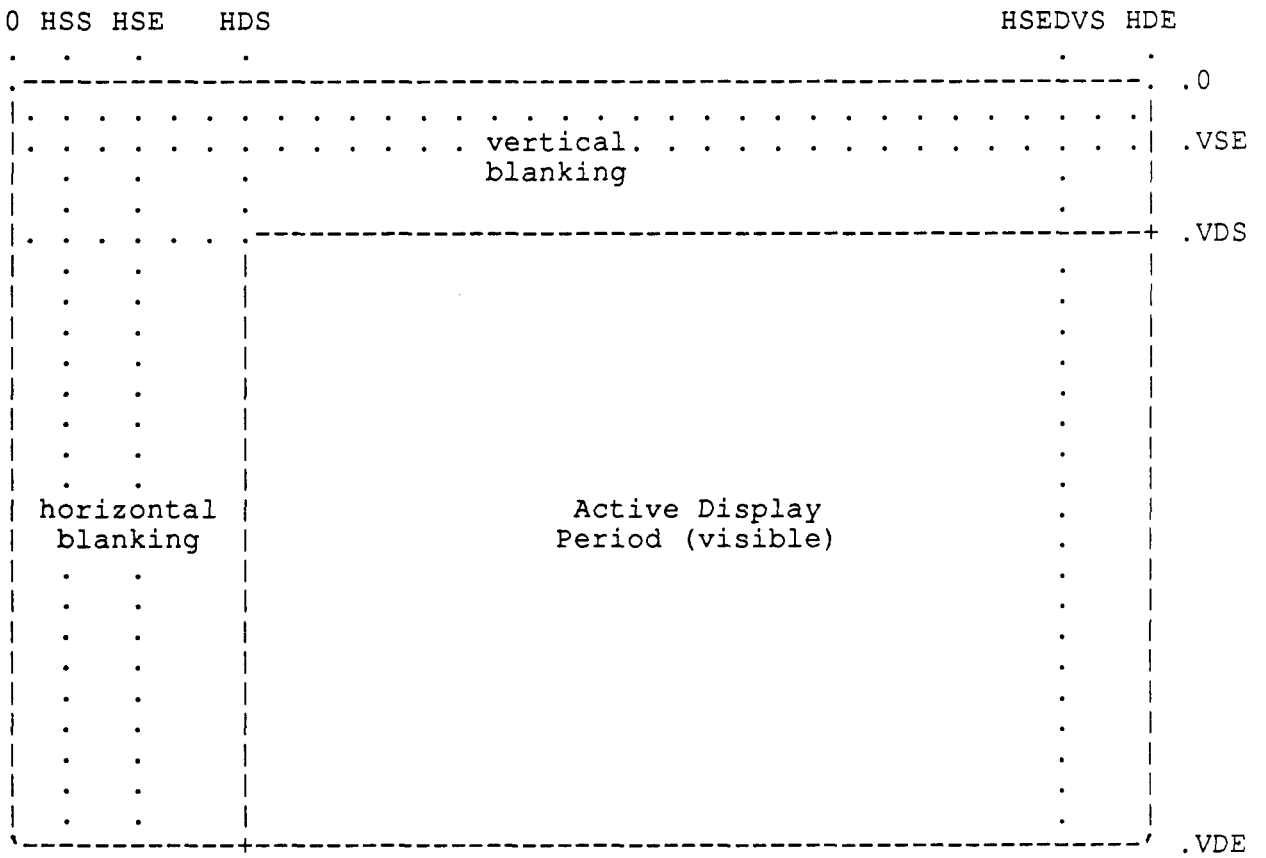



Figure 7-1 Control Signals

Table 7-1 contains the timing data for Sun's 1024x768, 1024x1024, 1152x900, 1152x870, and 1600x1280 monitors.

| Resolution | 1024 x 768 | | 1024 x 1024 | | 1152 x 900 | | 1152 x 870 | | 1600 x 1280 | |
|---|---|---|---|---|---|---|---|---|---|---|
| Pixel Freq | 70.4 MHz | | 92.9405 MHz | | 92.9405 MHz | | 100.00 MHz | | 200.00 MHz | |
| Pixel Period | 14.2 ns | | 10.76 ns | | 10.76 ns | | 10.00 ns | | 5.00 ns | |
| **Horz Timing** | **usec** | **pixel** | **usec** | **pixel** | **usec** | **pixel** | **usec** | **pixel** | **usec** | **pixel** |
| Resolution | 0.227 | 16 | 0.172 | 16 | 0.172 | 16 | 0.160 | 16 | 0.160 | 32 |
| Frontporch | 0.000 | 0 | 0.344 | 32 | 0.344 | 32 | 0.320 | 32 | 0.000 | 0 |
| Sync Width | 1.818 | 128 | 1.377 | 128 | 1.377 | 128 | 1.280 | 128 | 1.280 | 256 |
| Backporch | 2.272 | 160 | 2.582 | 240 | 2.064 | 192 | 1.440 | 144 | 1.920 | 384 |
| Invisible | 4.090 | 288 | 4.304 | 400 | 3.787 | 352 | 3.040 | 304 | 3.200 | 640 |
| Visible | 14.541 | 1024 | 11.018 | 1024 | 12.395 | 1152 | 11.520 | 1152 | 8.000 | 1600 |
| Total | 18.630 | 1312 | 15.322 | 1424 | 16.182 | 1504 | 14.560 | 1456 | 11.200 | 2240 |
| Frequency | 53.676 KHz | | 65.267 KHz | | 61.795 KHz | | 68.681 KHz | | 89.286 KHz | |
| **Vert Timing** | **usec** | **lines** | **usec** | **lines** | **usec** | **lines** | **usec** | **lines** | **usec** | **lines** |
| Resolution | 18 | 1 | 15 | 1 | 16 | 1 | 14 | 1 | 11 | 1 |
| Frontporch | 0 | 0 | 31 | 2 | 32 | 2 | 44 | 3 | 0 | 0 |
| Sync Width | 112 | 6 | 61 | 4 | 65 | 4 | 44 | 3 | 112 | 10 |
| Backporch | 727 | 39 | 506 | 33 | 502 | 31 | 568 | 39 | 482 | 43 |
| Invisible | 838 | 45 | 598 | 39 | 599 | 37 | 655 | 45 | 594 | 53 |
| Visible | 14308 | 768 | 15689 | 1024 | 14564 | 900 | 12667 | 870 | 14336 | 1280 |
| Total | 15147 | 813 | 16287 | 1063 | 15163 | 937 | 13322 | 915 | 14930 | 1333 |
| Frequency | 66.022 Hz | | 61.399 Hz | | 66.950 Hz | | 75.062 Hz | | 66.981 Hz | |

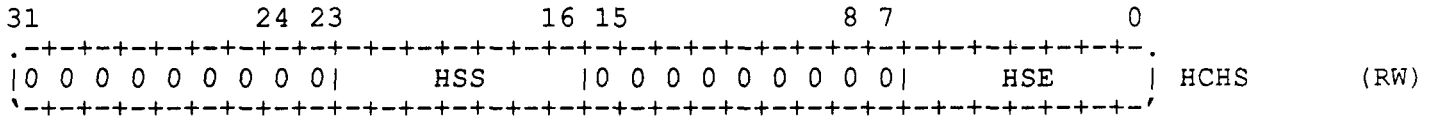
Table 7-1 Monitor Timings

Given the timing data for the monitors, the TEC would be programmed with the values in Table 7-2. The values are calculated by subtracting one from the state of the horizontal and vertical counters in which the signals should change. If the value is less than zero the number of counts per line or screen of the respective counter must be added to the value to make it positive.

| Description | Symbol | 1024x768 Value | 1024x1024 Value | 1152x900 Value | 1152x870 Value | 1600x1280 Value |
|---|---|---|---|---|---|---|
| Horz Sync Start | HSS | 81 | 1 | 1 | 1 | 69 |
| Horz Sync End | HSE | 7 | 9 | 9 | 9 | 7 |
| Horz Dsp Start | HDS | 17 | 24 | 21 | 18 | 19 |
| Horz SE DVSYNC | HSEDVS | 73 | 82 | 87 | 84 | 61 |
| Horz Dsp End | HDE | 81 | 88 | 93 | 90 | 69 |
| Vert Sync Start | VSS | 812 | 1 | 1 | 2 | 1332 |
| Vert Sync End | VSE | 5 | 5 | 5 | 5 | 9 |
| Vert Dsp Start | VDS | 38 | 38 | 36 | 44 | 52 |
| Vert Dsp End | VDE | 812 | 1062 | 936 | 914 | 1332 |
| Horz Sync | HCHS | 0x00510007 | 0x00010009 | 0x00010009 | 0x00010009 | 0x00450007 |
| Horz S DVSYNC | HCHSDVS | 0x00490000 | 0x00520000 | 0x00570000 | 0x00540000 | 0x003d0000 |
| Horz Dsp | HCHD | 0x00110051 | 0x00180058 | 0x0015005d | 0x0012005a | 0x00130045 |
| Vert Sync | HCVS | 0x032c0005 | 0x00010005 | 0x00010005 | 0x00020005 | 0x05340009 |
| Vert Dsp | HCVD | 0x002c032c | 0x00260426 | 0x002403a8 | 0x002c0392 | 0x00340534 |

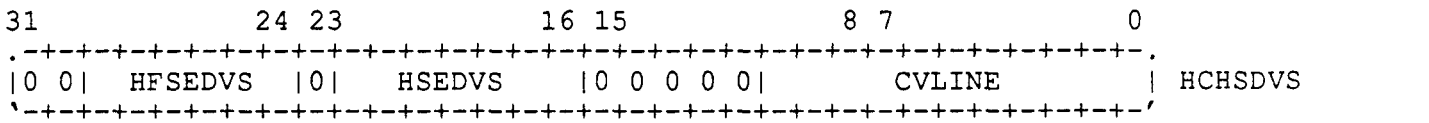
Table 7-2 Timing Values

The timing registers can be read and written for diagnostics.



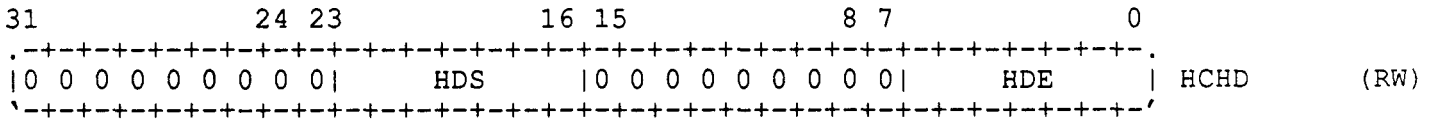

| Bits | Description | Units | |
|---|---|---|---|
| 22-16 | Horizontal Sync Start | 16/32 Pixels | |
| 6-0 | Horizontal Sync End | 16/32 Pixels | |

Figure 7-2 Horizontal Sync Start/End Register



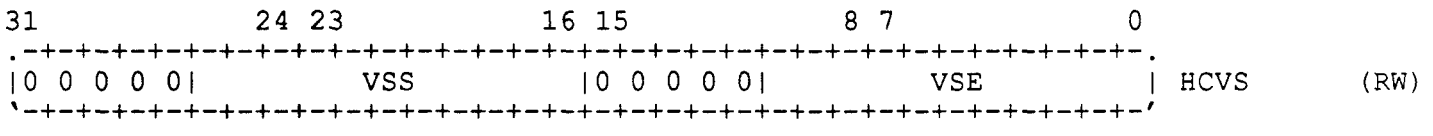

| Bits | Description | Units | |
|---|---|---|---|
| 29-24 | Half Horizontal Sync End DVS | 16/32 Pixels | (RO) |
| 22-16 | Horizontal Sync End DVS | 16/32 Pixels | (RW) |
| 10-0 | Current Vertical Line | Line | (RO) |

Figure 7-3 Horizontal Sync End During Vertical Sync Register



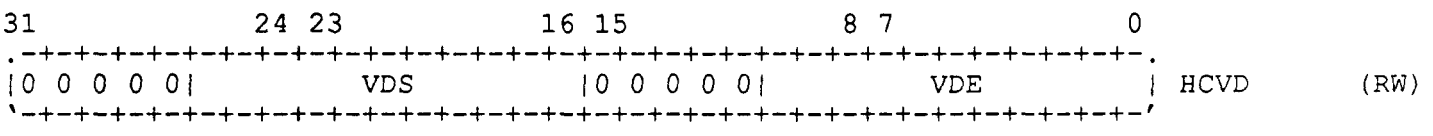

| Bits | Description | Units |
|---|---|---|
| 22-16 | Horizontal Display Start | 16/32 Pixels |
| 6-0 | Horizontal Display End | 16/32 Pixels |

Figure 7-4 Horizontal Display Start/End Register




| Bits | Description | Units |
|---|---|---|
| 26-16 | Vertical Sync Start | Lines |
| 10-0 | Vertical Sync End | Lines |

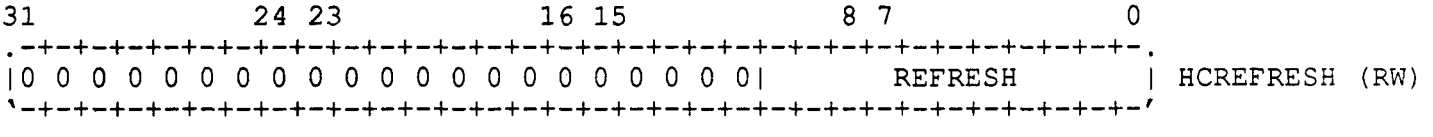
Figure 7-5 Vertical Sync Start/End Register




| Bits | Description | Units |
|---|---|---|
| 26-16 | Vertical Display Start | Lines |
| 10-0 | Vertical Display End | Lines |

Figure 7-6 Vertical Display Start/End Register

The refresh counter counts the CLK transitions. The upper bit of the counter is made available through the REFRESH pin to indicate to the memory controller when to do a refresh cycle. The referesh value are calculated by subtracting one from the refresh period divided by the CLK period.




| Bits | Description | Units |
|---|---|---|
| 10-0 | Refresh Counter | CLK |

Figure 7-7 Refresh Counter

## 7.2 MISCELLANEOUS REGISTER

The hardware configuration miscellaneous register contains a bunch of miscellaneous leftover stuff that is system dependent.

The Chip Revision Number is a unique number for every TEC developed.

When the RESET bit is set all state machines except for the CPU interface are reset.

The ENABLE_VIDEO bit is the same as the enable video bit in the system registers. This allows the TEC to reside on a bus that does not have the system ENABLE_VIDEO signal on it. ENABLE_VIDEO is cleared on reset.

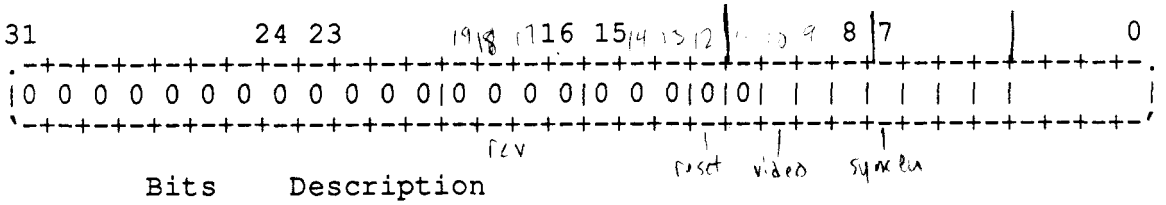
The SYNC_ and VSYNC_ are read only bits which show the state of these two signals.

The ENABLE_SYNC bit enables the composite sync output. ENABLE_SYNC is cleared on reset.

When the color look up table needs to be changed, the software would set the EN_VBLANK_IRQ bit. On the next vertical blanking, an interrupt is generated to the operating system. Control would be transferred to the graphics driver to handle the interrupt. The software can poll the VBLANK_IRQ_OCCURED bit to see if the interrupt has occured. The driver clears the interrupt by setting the VBLANK_IRQ_OCCURED bit.

The CYCLES_BEFORE_TRANSfer value tells the TEC how many video load cycles there must be before the start of a transfer cycle to the video ram. The TRANS pin will be active low for (CYCLES_BEFORE_TRANS+2)*(LD_CLK Period).

The HCMISC register can be read and written. This data does not have to be saved or restored for context switching, since the information in this register is not context dependent.




| Bits | Description | |
|---|---|---|
| 19-16 | Chip Revision Number | (RO) |
| 12 | RESET (1=Reset TEC) | (WO) |
| 10 | ENABLE_VIDEO (0=Screen Blanked, 1=Screen Displayed) | |
| 9 | SYNC_ | (RO) |
| 8 | VSYNC_ | (RO) |
| 7 | ENABLE_SYNC (0=Sync Disabled, 1=Sync Enabled) | |
| 6 | Cursor Resolution (0=1152, 1=1600) | |
| 5 | EN_VBLANK_IRQ (1=Enable Interrupt) | |
| 4 | VBLANK_IRQ_OCCURED (1=Clear Interrupt) | |
| 3-0 | CYCLES_BEFORE_TRANS | |

Figure 7-8 Hardware Configuration Miscellaneous Register