October 1992

Chapter 1. Features

2-D GRAPHICS ACCELERATOR

Single-chip controller

Accelerates Microsoft Windows and AutoCAD

Supports X-Windows drawing modes

POWERFUL GRAPHICS PRIMITIVES

Draws lines and polygons with pattern fill at full VRAM bandwidth (up to 132 million pixels per second, and 1.53 million vectors per second)

Performs bit block-transfer (BitBlt) from screen to screen at VRAM bandwidth (up to 40 million pixels per second), and from host to screen at host bus bandwidth

Supports two BitBlt modes: one with color expansion for text, the other for graphics

Supports patterning, plane masking, and boolean operations of pixels during drawing

Performs automatic clipping against window and screen edges

Supports polylines and mesh polygons for faster drawing

Supports pick mode

Supports resolutions up to 1600x1200x8, 1280x1024x8, 1024x768x16, and 800x600x24

HIGH INTEGRATION

Supports 32-bit non-multiplexed general host interface

32-bit interleaved memory interface supports 1, 2, or 4MB (double buffered) VRAM frame buffer

Supports video rates to 165 MHz; connects directly to a standard 32-bit serial RAMDAC interface (such as the Bt 458, 459, or compatible)

SOFTWARE SUPPORT

Windows 3.0/3.1

AutoCAD R10 and R11

1.1. Description

The WEITEK Power 9000 User Interface Controller is an accelerated 2-D graphics device used with Microsoft Windows and AutoCAD. It supports draw, fill, and bit block-transfer operations at the full speed of interleaved page-mode VRAMs — 132 million pixels per second — at screen sizes of up to 2 million pixels. The Power 9000 is a single 25–33 MHz CMOS chip which comes in a 208-pin PQFP (Plastic Quad Flat Package).

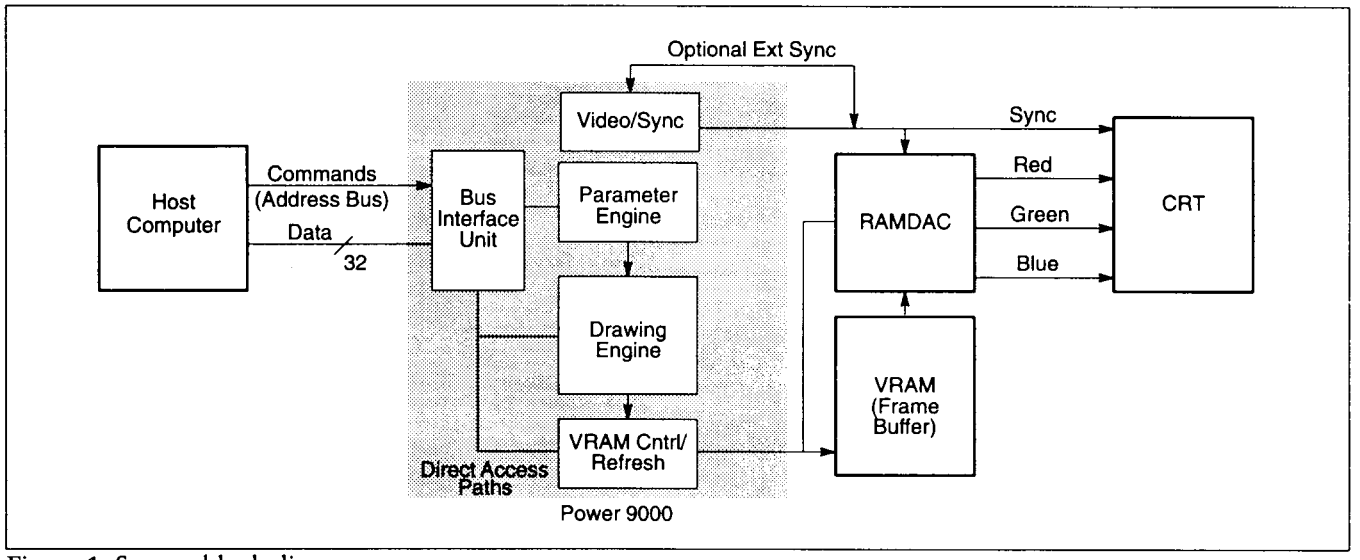

Figure 1. System block diagram

1.2. Block Diagram

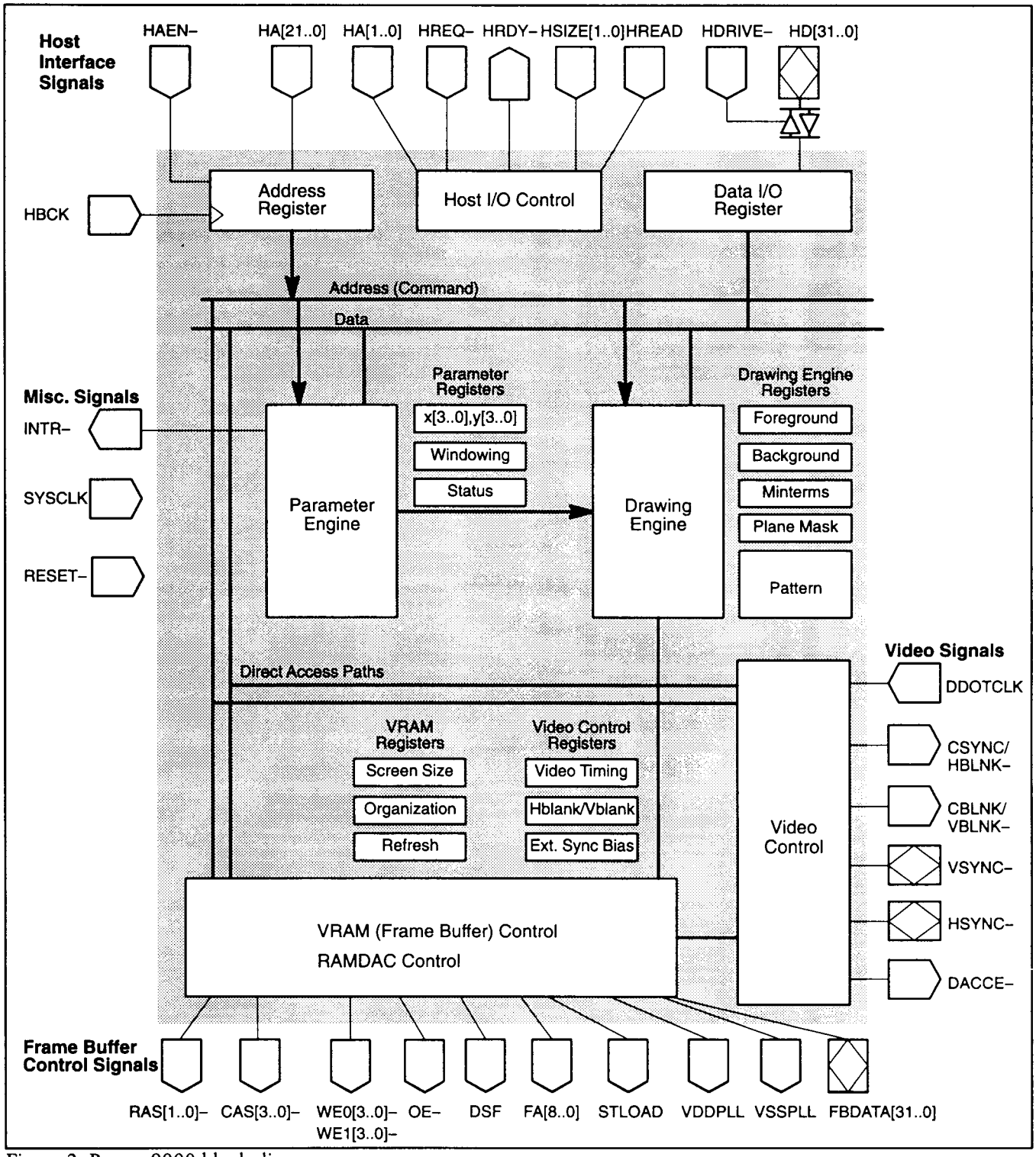

Figure 2. Power 9000 block diagram

1.3. Architecture

At the chip level, the Power 9000 is a hard-wired graphics processor. Its operation is determined by its internal registers and by command words sent over the host interface bus. At the system level, it is a memory-mapped peripheral. It operates as a bus slave; the host initiates all bus reads and writes. This results in simple interface and programming models.

With its direct control of VRAMs, the Power 9000 draws at the speed of its memory system. The VRAM frame buffer runs in interleaved mode for maximum performance, using standard VRAMs. The Power 9000 feeds video data into a RAMDAC, which contains internal color look-up tables and digital-to-analog converters. The graphics board then feeds the resulting analog RGB video signal, along with the horizontal and vertical synchronization signals generated by the Power 9000, into a high-resolution color monitor with a screen size of up to 2 megapixels.

The Power 9000 contains the following functional units: the *parameter engine,* the *drawing engine,* the *host interface,* the *frame buffer controller,* and the *video controller.*

1.3.1. PARAMETER ENGINE

The parameter engine prepares drawing operations for the drawing engine; its basic function is to take input coordinates from the host and convert them to a form usable by the drawing engine. The input parameters include the *x,y* vertices of polygons and the corners of bit block-transfer *(BitBlt)* regions. The parameter engine tests the vertices against window and screen boundaries, tests for exceptions, and performs trivial rejection. Finally, it transfers commands that pass these tests to the drawing engine to be executed.

The parameter engine prepares four kinds of "polygons" for drawing: quadrilaterals, triangles, lines, and points. It converts points, lines, and triangles into quadrilaterals by automatically replicating vertices; a point, for example, is a quadrilateral with all four vertices at the same *x,y* location. Thus, the user has to load only a single vertex to draw a point; the parameter engine passes four copies of that vertex to the drawing engine and instructs it to draw a quad.

The parameter engine also handles screen-to-screen BitBlt and two kinds of host-to-screen BitBlt, one optimized for text and the other optimized for graphics. The parameter engine handles all exception testing and access to parameter engine registers, while it passes operations that actually write to the display to the drawing engine.

1.3.2. DRAWING ENGINE

The drawing engine performs three basic functions: (1) It draws quadrilaterals (the *quad* operation), (2) it performs screen-to-screen BitBlt (the *blit* operation), and (3) it performs host-to-screen BitBlt (the *pixel1* and *pixel8* operations).

The quad operation draws quadrilaterals in one of two modes: *X11 mode,* an X-Windows compatible mode, and *oversized mode,* a Bresenham mode. Triangles, lines, and points can always be rendered correctly, but the drawing engine cannot draw horizontally convex quadrilaterals. That is, it cannot cross from the inside to the outside of the same object more than once per scan line. This means that "bowties" cannot be drawn (such quads must be rendered in software), though "hourglasses" can (see figure 3).

The blit operation copies a rectangular area of the display from one screen location to another.

The pixel1 operation takes monochrome, one-bit-per-pixel data from the host, expands the pixels internally to eight bits per pixel, and writes them to the frame buffer. Up to 32 pixels can be transferred to the Power 9000 in a single word. The pixel8 operation takes color, eight-bits-per-pixel data (up to four pixels per word), and writes them to the frame buffer.

1.3.3. HOST INTERFACE

The Power 9000 appears to the host as an array of memory. The Power 9000 uses only the lower 22 address bits. The chip ignores higher-order bits; external glue logic generally decodes them to distinguish Power 9000 from non-Power 9000 bus accesses. Half of this space (2 MB) is used for direct frame-buffer access, allowing the host to use the Power 9000 as a dumb frame buffer; 1MB is reserved for non-Power 9000 operations. The rest of the address space is decoded into Power 9000 instructions. Power 9000 commands are specified by the address, allowing the data bus to be devoted solely to data transfers and minimizing the number of bus accesses.

The Power 9000 supports both big-endian and little-endian address formats. It can run with both multiplexed and non-multiplexed buses, and run either synchronously or asynchronously to the bus clock.

1.3.4. FRAME BUFFER CONTROLLER

The Power 9000 controls the VRAM frame buffer directly; VRAM chips are wired directly to the Power 9000 (the Power 9000 has high-current output drivers to support this). Frame-buffer control registers in the Power 9000 determine the VRAM refresh rate, the screen size, and single- or double-buffering; they also select interleaved or non-interleaved VRAM modes. The host initializes these registers at system start-up.

1.3. Architecture, continued

1.3.5. VIDEO CONTROLLER

A typical board design feeds the VRAM shift registers into a Brooktree-compatible RAMDAC, which uses a color lookup table to convert eight-bit pixel data into a 24-bit analog video signal. The host initializes the control registers and look-up tables in the RAMDAC through the Power 9000's RAMDAC access instructions.

The Power 9000 also generates horizontal and vertical synchronization signals and controls the clocking of the video data. The divided dot clock used by the Power 9000's video subsection is completely asynchronous to the Power 9000's main system clock.

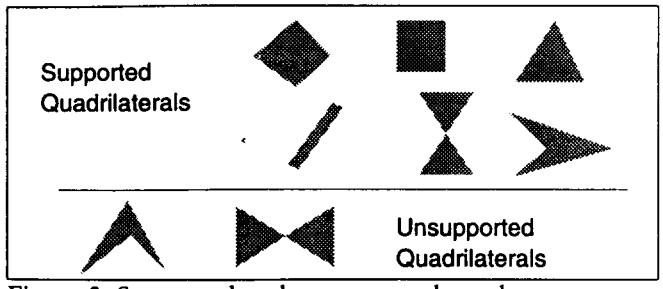

Figure 3. Supported and unsupported quads

1.4. Signal Description

Signal	Subsystem	Definition
CAS[3..0]–	Frame Buffer	Column address strobe output signals
CBLNK/VBLNK–	Video	Composite blank/vertical blank output signal
CSYNC/HBLNK–	Video	Composite synchronization/horizontal blank output signal
DACCE–	RAMDAC Control	RAMDAC enable output signal
DDOTCLK	Video	Divided dot clock input signal
DSF	Frame Buffer	Shift register load control output signal
FA[8..0]	Frame Buffer	Frame buffer output address lines
FBDATA[31..0]	Frame Buffer	Frame buffer I/O data bus
HA[21..0]	Host Bus Interface	Host to Power 9000 input address bus
HAEN–	Host Bus Interface	Host address clock enable input signal
HBCK	Host Bus Interface	Host bus clock input signal
HD[31..0]	Host Bus Interface	Host to Power 9000 I/O data bus
HDRIVE–	Host Bus Interface	Host bus data output enable input signal
HREAD	Host Bus Interface	Host read/write input signal
HRDY–	Host Bus Interface	Host bus ready output signal
HREQ–	Host Bus Interface	Host request input signal
HSIZE[1..0]	Host Bus Interface	Host transfer size input signals
HSYNC–	Video	Horizontal synchronization input/output signal
INTR–	Host Bus Interface	System interrupt request output signal
OE–	Frame Buffer	Frame buffer data transfer/output enable output signal
RAS[1..0]–	Frame Buffer	Row address strobe output signals
RESET–	Host Bus Interface	System reset input signal
STLOAD	Frame Buffer	Self-timing load output signal
SYSCLK	Subsystem	Power 9000 system clock input signal
VDDPLL	Frame Buffer	Supply voltage for on-chip clock generator
VSSPLL	Frame Buffer	Ground for on-chip clock generator
VSYNC–	Video	Vertical synchronization input/output signal
WE0[3..0]–, WE1[3..0]–	Frame Buffer	Frame buffer write enable output signals

Figure 4. Power 9000 signals

1.4.1. HOST BUS INTERFACE SIGNALS

HOST ADDRESS BUS

The HA[21..0] *host address* bus is the 22-bit synchronous, unidirectional address bus from the host system processor to the Power 9000. This bus specifies the Power 9000 command. The two low order bits, combined with the HSIZE[1..0] *host transfer size* signals, define the data bus bytes to be written to the Power 9000 (see figure 5). For a host read, the Power 9000 drives an entire word of data onto the data bus; the Power 9000 ignores the two low order address bits and the host transfer size.

HOST TRANSFER SIZE

The HSIZE[1..0] *host transfer size* input signals, combined with the two lowest host address bus signals, determine which bytes are valid on the data bus to write to the Power 9000. See figure 5.

Figure 6 defines the equivalencies between the Power 9000 HA[1..0] and HSIZE[1..0] bits and the EISA/VESA local bus bits be[3..0]–.

1.4. Signal Description, continued

Values		Host Data Bus Bytes Written			
HSIZE[1..0]	HA[1..0]	HD[31..24]	HD[23..16]	HD[15..8]	HD[7..0]
00	00	X	X	X	X
00	01		X	X	X
00	10			X	X
00	11				X
01	00	X			
01	01		X		
01	10			X	
01	11				X
10	00	X	X		
10	01		X	X	
10	10			X	X
10	11				X
11	00	X	X	X	
11	01		X	X	X
11	10			X	X
11	11				X

Figure 5. Host data bus bytes written as a result of combinations of HSIZE[1..0] and HA[1..0] values

EISA/VESA Local Bus Byte Enables	Power 9000 Bits	
be[3..0]–	HSIZE[1..0]	HA[1..0]
0000	00	00
0001	11	00
0010	not used	not used
0011	10	00
0100	not used	not used
0101	not used	not used
0110	not used	not used
0111	01	00
1000	11	01
1001	10	01
1010	not used	not used
1011	01	01
1100	10	10
1101	01	10
1110	01	11
1111	not used	not used

Figure 6. HA[1..0], HSIZE[1..0] relation to be[3..0]–

HOST DATA BUS

The HD[31..0] *host data* bus is the 32-bit bidirectional data bus between the Power 9000 and the host system processor. This bus allows direct transfer of data to and from each of the major Power 9000 units (the parameter engine, the drawing engine, the frame buffer controller, and the video controller).

HOST ADDRESS REGISTER ENABLE

The HAEN– *host address register enable* input signal enables data transfer to the host address register on the rising edge of the host bus clock, HBCK. This function is necessary only when the host employs a multiplexed bus for transferring addresses and data (the signal can be tied low for hosts with separate address and data buses). The host bus clock (HBCK) clocks the address into the address register only if HAEN– is asserted; otherwise, the register holds its previous value. HA[1..0] are not latched in the host address register (these bits are not used by that circuitry which uses the host address register).

HOST BUS CLOCK

The HBCK *host bus clock* input signal clocks the host address register. This clock signal can be asynchronous to the Power 9000 clock (see chapter 2).

HOST REQUEST

The HREQ– *host request* input signal indicates host initiation of a Power 9000 read or write. The HREAD signal determines the direction of the data transfer. The Power 9000 accepts a host request only when the Power 9000 host interface is ready (HRDY– asserted).

HOST READ

The HREAD *host read* input signal is a read/write signal for host requests. Asserting both HREAD and HREQ– initiates a read; asserting HREQ– without asserting HREAD initiates a write.

HOST DRIVE

The HDRIVE– *host drive* input signal is an asynchronous tri-state enable for driving data from the Power 9000 to the host data bus. This signal is driven by external logic. Its assertion forces the Power 9000 to drive data onto the host bus.

HOST BUS READY

The HRDY– *host bus ready* output signal is asserted when the Power 9000 is ready to accept a request. The Power 9000 deasserts this signal for at least one cycle after accepting a host request. For a read, the Power 9000 deasserts HRDY– when it starts the read and re-asserts HRDY– one cycle before the data is valid.

1.4. Signal Description, continued

INTERRUPT

The INTR– *system interrupt* output signal indicates a Power 9000 interrupt request. This is an open-drain signal; it is driven low when asserted and floats when deasserted.

RESET

The RESET– *system reset* input signal performs a system reset on the Power 9000. The system reset signal may be asynchronous to the system clock.

The RESET– signal must be asserted for a minimum of 50 SYSCLK cycles. While RESET– remains asserted, no VRAM, video, or host bus activities can occur. The Power 9000 deasserts all VRAM control signals (RAS[1..0]–, CAS[3..0]–, WE0[3..0]–, WE1[3..0]–, and OE–) and puts the frame buffer data bus lines (FBDATA[31..0]), the host data bus (HD[31..0]), and the system interrupt signal (INTR–) into the high-impedance state.

A system reset affects the Power 9000 registers as summarized in Chapter 3, figure 57.

1.4.2. SYSTEM SIGNALS

SYSTEM CLOCK

The SYSCLK *system clock* input is the main Power 9000 clock signal. The Power 9000 system clock may be asynchronous to all other clocks in the system.

1.4.3. VIDEO DISPLAY SIGNALS

DIVIDED DOT CLOCK

The DDOTCLK *divided dot clock* input signal is the clock signal used to generate all video control signals. This signal is generated by external logic. Since the dot clock usually exceeds the Power 9000's maximum frequency, the Power 9000 accepts a divided dot clock. The divided dot clock is most commonly the dot clock divided by four, but the divisor can be any power of two that produces a signal that satisfies the Power 9000's maximum frequency.

HORIZONTAL SYNCHRONIZATION

The HSYNC– *horizontal synchronization* signal can be used as either an output signal or an input signal. When defined as an output signal, HSYNC– drives the horizontal synchronization signal used by the monitor, RAMDAC, and timing logic. When HSYNC– is defined as an input signal, the Power 9000 accepts an external horizontal synchronization signal and generates screen repaints based upon that signal's timing. A system reset defines HSYNC– as an input signal by clearing the screen repaint timing control (srtctl) register (see section 3.4.4).

VERTICAL SYNCHRONIZATION

The VSYNC– *vertical synchronization* signal can be used as either an output signal or an input signal. When defined as an output signal, VSYNC– drives the vertical synchronization signal used by the monitor, RAMDAC, and timing logic. When VSYNC– is defined as an input signal, the Power 9000 accepts an external vertical synchronization signal and generates screen repaints based upon that signal's timing. A system reset defines VSYNC– as an input signal by clearing the screen repaint timing control (srtctl) register (see section 3.4.4).

COMPOSITE SYNCHRONIZATION/HORIZONTAL BLANK

The CSYNC/HBLNK– output signal can be used either as a *composite synchronization* (combined horizontal and vertical synchronization) signal or as a *horizontal blank* signal. As a *composite synchronization* signal, it is a combination of HSYNC– and VSYNC–. As a *horizontal blank* signal, it generates the horizontal blanking signal used by the monitor and the timing logic. A system reset defines CSYNC/HBLNK– as a *horizontal blank* signal by clearing the screen repaint timing control (srtctl) register (see section 3.4.4).

COMPOSITE BLANK/VERTICAL BLANK

The CBLNK/VBLNK– output signal can be used either as a *composite blank* signal or as a *vertical blank* signal. In either case, the monitor and timing logic use the signal. A system reset automatically defines CBLNK/VBLNK– as a *vertical blank* signal by clearing the screen repaint timing control (srtctl) register (see section 3.4.4).

1.4.4. FRAME BUFFER SIGNALS

ROW ADDRESS STROBE

The RAS[1..0]– *row address strobe* output signals connect directly to the VRAM RAS inputs.

COLUMN ADDRESS STROBE

Each of the CAS[3..0]– *column address strobe* output signals connects directly to the VRAM CAS inputs. The multiple CAS lines reduce loading and enable double buffering (see chapter 2).

WRITE ENABLE

The WE0[3..0]– and WE1[3..0]– *write enable* output signals connect directly to the VRAM WB/WE– signals. These signals provide a byte write control, enabling individual eight-bit pixels to be written with a 32-bit bus transfer.

Chapter 2 defines how these signals must be connected for various VRAM configurations. The WE1[2..0]– signals also function as control signals for the host access port of the RAMDAC, as illustrated in figures in chapter 2. WE1[1..0]– control the RAMDAC control signals (C0 and C1) and WE1[2]– controls the RAMDAC R/W signal.

OUTPUT ENABLE

The OE– *output enable* output signal enables transfer of data from the VRAMs. The OE– signal connects directly to the DT/OE– signal for each VRAM.

FRAME BUFFER DATA BUS

The FBDATA[31..0] *frame buffer data bus* is the 32-bit bidirectional data bus connected through damping resistors to the VRAMs in each bank.

FRAME BUFFER ADDRESS BUS

The FA[8..0] *frame buffer address bus* is a nine-bit multiplexed output bus connected directly to the VRAM address lines in both banks, through damping resistors.

SPECIAL FUNCTION CONTROL

The DSF *special function control* input signal controls split-shift register loads. This signal is connected to the DSF of each VRAM. See the VRAM specifications for details.

SELF-TIMING LOAD

The STLOAD *self-timing load* output signal is used to generate VRAM timings. It should be connected to a capacitor equal in value to the capacitance of the most heavily loaded FA line.

CLOCK GENERATOR SUPPLY VOLTAGE

The VDDPLL clock generator supply voltage output signal supplies voltage for the on-chip clock generator. Keep the supply as clean as possible; you may want to use a voltage regulator.

CLOCK GENERATOR GROUND

The VSSPLL clock generator ground output signal supplies the ground for the on-chip clock generator. Decouple VDDPLL to VSSPLL with a .1μF capacitor located as close to the pins as possible.

1.4.5. RAMDAC CONTROL SIGNALS

RAMDAC CHIP ENABLE

The DACCE– *RAMDAC chip enable* output signal enables a data transfer to or from the RAMDAC. This signal connects directly to the RAMDAC CE– signal. The WE1[2] signal determines the direction of the transfer; the WE1[1..0] signals determine the transfer type.

NOTE: This signal is intended to control workstation RAMDACs. For PCs, it may be unused, with the RAMDAC connected to the ISA bus.

1.5. Graphics Operation

1.5.1. THE GRAPHICS PIPELINE

Graphics operations flow through the *graphics pipeline,* first through the parameter engine, and then through the stages of the drawing engine.

THE PARAMETER ENGINE

Functions. The parameter engine determines what will happen as the result of each drawing operation. It calculates status information, including whether the operation is clipped by the viewing window or the screen edge, and whether the operation can be drawn at all before passing the operation to the drawing engine. If the drawing engine is busy, or if the request contains illegal parameters, the parameter engine does not pass on the request. (The parameter engine performs these tests only on coordinate register loads and drawing commands. Other operations, such as setting color registers or video timing registers, bypass the parameter engine.)

The parameter engine performs clipping calculations on each *x,y* vertex. It compares these points against all four edges of the screen and viewing window, and against each other. It also tests for trivial rejection and trivial acceptance.

The parameter engine detects, but cannot correct an illegal request, such as a request for a horizontally convex quad. Such quadrilaterals must be rendered in software. The status register flags such problems. In addition, the interrupt signal can be used to interrupt the host when such exceptions occur.

Access. All accesses to parameter engine functions and registers complete in a few cycles; the parameter engine is always accessible.

THE DRAWING ENGINE

Functions. The drawing engine accepts one drawing operation at a time from the parameter engine. When processing a drawing operation, it first determines which pixels are "touched" by the drawing process *(scan conversion),* then determines the color value of each touched pixel *(raster ops).*

Scan conversion. Two line-drawing engines follow the left and right edges of the quadrilateral on each scan line, and a pixel-processing engine fills the region between these edges. (Pixel1, pixel8, and blit operations are limited to rectangular areas, while the quad operation can have edges at any angle).

Raster ops. The color of each touched pixel is determined by the raster-op function, which is further conditioned by the contents of other registers (see section 1.5.5).

Access. The drawing engine can remain busy for long periods when drawing large quads or performing a blit operation on a large portion of the screen. The quad and blit operations are started by a read operation. The read requests that the operation take place, and returns the contents of the status register, which indicates whether or not the request has been granted. The Power 9000 does not accept a request if the drawing engine is busy, or if an exception has occurred, nor does it queue requests. Therefore, the software must check the status register, which contains both exception and drawing engine status bits, and resubmit any quad or blit request that is not accepted. While no new drawing operations can be started until the drawing engine is idle, the host can load the parameter engine's coordinate registers with the vertices of a new operation while the drawing engine is busy, thereby overlapping the processing of two objects.

1.5.2. DRAWING QUADRILATERALS

The drawing engine assigns the two edges at either the topmost or bottom-most vertex to its two Bresenham *line-drawing engines.* These engines do not actually draw anything in the frame buffer, but they traverse the boundaries of the quadrilateral on each scan line. The *pixel-processing engine* then fills in the pixels between the boundaries found by the line-drawing engines. (This filling operation is also clipped against the clipping window and screen boundaries.) See figure 7. When a line-drawing engine reaches another vertex, it starts down the new edge.

In oversized mode, the Power 9000 draws perimeter pixels according to the Bresenham algorithm. Oversized mode must be selected to draw points and lines, as X11's drawing rules do not "touch" pixels in zero-width objects (meaning nothing is drawn).

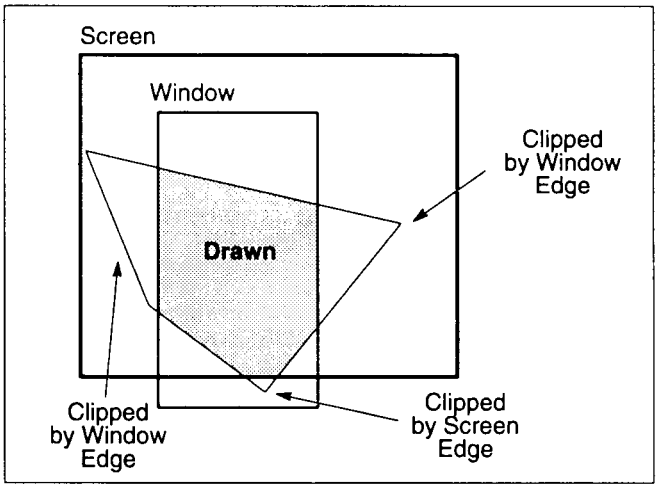

Figure 7. A quad clipped against window and screen

1.5. Graphics Operation, continued

In *X11 mode,* the Power 9000 draws only those pixels whose centers lie within the perimeter of the quad. Pixels whose centers lie precisely on the perimeter are drawn in accordance with X11's tie-breaker rules. (See figure 8.)

Coordinates can either be specified relative to the clipping window or to the previous vertex.

Polylines, meshed triangles, and meshed quadrilaterals are all supported; they are drawn by transferring the new vertices and issuing another draw command.

1.5.3. DRAWING BIT MAPS

To transfer a bit map to the screen, the host first sets up the *x,y* coordinates of the destination, and then transfers data to the Power 9000 with the pixel1 and pixel8 operations. The Power 9000 draws the pixels on the screen, auto-incrementing the current *x,y* location after each pixel. Bit-map drawing proceeds from left to right; when it reaches the right-hand edge of the target bit map, the Power 9000 automatically wraps to the next line.

The pixel8 operation draws multi-colored pixels; up to four eight-bit pixels are transferred and drawn through a single bus transfer from the host. This mode is used to transfer color images.

The pixel1 operation, which draws monochrome bit maps, is ideal for text transfer. Its basic operation is similar to pixel8, but instead of transferring four eight-bit pixels per operation, it transfers up to 32 one-bit pixels. The Power 9000 expands the one and zero bits of the data word into eight-bit pixels of value 255 and 0, respectively. (The Power 9000 expands these values according to the raster-op function described in section 1.5.5.)

Pixel data must be padded out to multiples of four bytes per scan line for pixel8 transfers; leftover pixels in the current word do not wrap to the next line. Pixel1 allows the left-over pixels to wrap, however, making it especially suitable for sending narrow blocks of monochrome data, such as character bit maps.

1.5.4. BIT BLOCK TRANSFER

The blit operation moves a rectangular block of pixels from one part of the screen to another. It handles overlapping source and destination blocks properly; the source block arrives unchanged at the destination, as if it had been moved off-screen, then copied back at its new destination.

Like quad, blit is a "fire-and-forget" operation; once initiated, it runs to completion without additional attention from the host. As a large blit can involve over a million pixels, the host driver code should test the busy bit in the Power 9000's status register before attempting another drawing operation when a blit could be in progress.

1.5.5. COLOR SELECTION

Once a pixel has been touched, there still remains the question of what color it will be. Screen color is selected at five levels: (1) the initial (source) color, (2) the pattern color, (3) the raster-op color, (4) the plane mask, and (5) the RAMDAC look-up table color. The Power 9000 applies each of these in turn. The *pattern RAM* allows imposition of a 16x16-bit repeating pattern on the data. See figure 9. The *raster-op* function is a four-input boolean function controlled by a 16-bit minterm array. The four inputs are: (1) the source color, (2) the destination color (the color value currently at the *x,y* location being written), (3) the foreground color, and (4) the background color. The 16-bit minterm array enables the raster-op function. The Power 9000 applies the same function to each of the eight bits in the pixel. The *plane mask* is an eight-bit mask that write-enables individual bits in the eight-bit pixel. The *RAMDAC* takes the pixel value from the frame buffer and uses it as an index into three parallel lookup tables: one each for red, green, and blue. Thus, the RAMDAC produces 24-bit color output from an eight-bit input (256 colors from a palette of 16 million colors). The output from the RAMDAC lookup table is fed to three eight-bit digital-to-analog converters, which produce the actual color video signals.

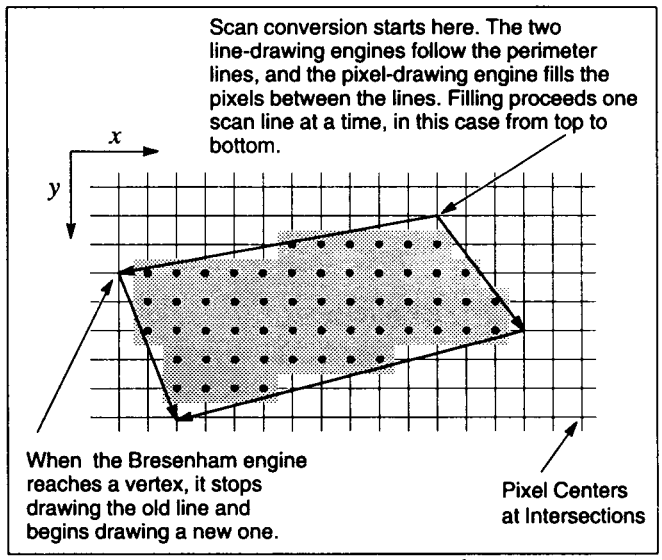

Figure 8. Scan conversion using X11 rules

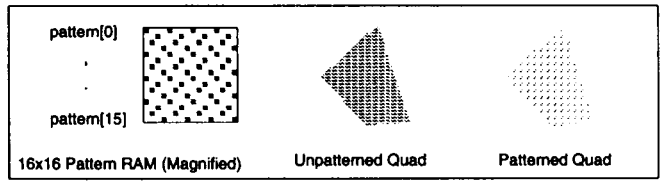

Figure 9. Patterning

1.6. Performance

Figure 10 illustrates the results of benchmark tests run on an EISA bus Compaq 486-25 with: (a) the 33 MHz Power 9000 running WEITEK's 1280x1024 eight-bit Windows driver, and (b) the S3 SC911 (Diamond Stealth VRAM) running a 1024x768 eight-bit Windows driver.

The results illustrated are for the PC Labs Windows Benchmarks 2.5 tests identified in figure 11.

Figure 12 presents WINMARK results.

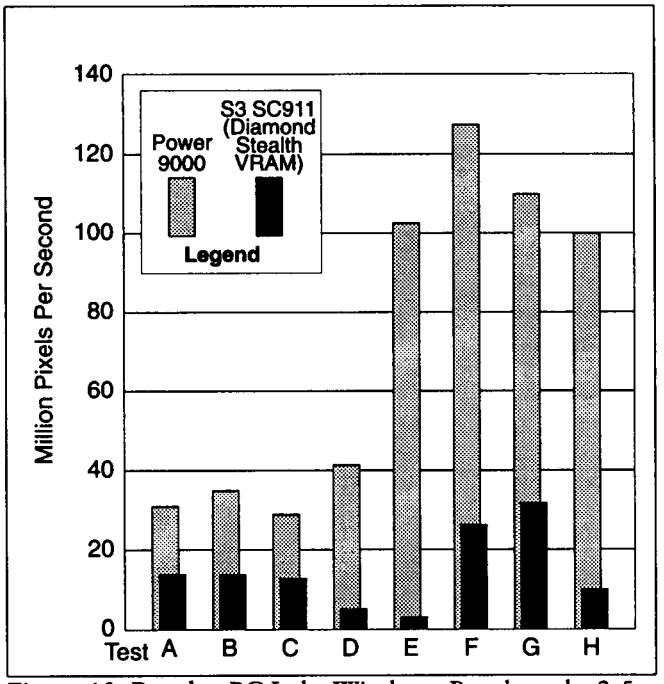

Figure 10. Results: PC Labs Windows Benchmarks 2.5

Test	Description
A	BitBlt alignment: 256x256 source aligned, destination aligned
B	BitBlt overlap: horizontal overlap
C	BitBlt raster-ops: screen to screen SRCCOPY
D	BitBlt raster-ops: screen to screen PATCOPY
E	Draw and fill rectangles: square rectangles
F	Fill with patterns: hatch patterns
G	Erase window
H	Draw single lines: single horizontal

Figure 11. Benchmark tests with results shown in figure 10

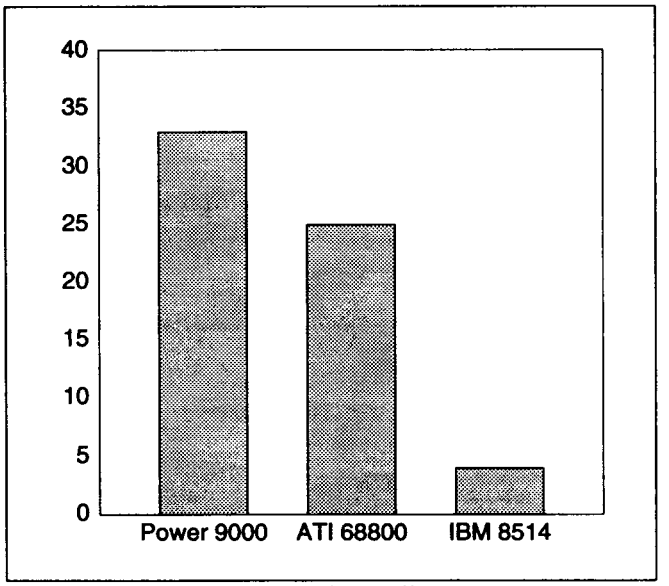

Figure 12. WINMARK results (millions)

1.7. Controller Examples

This section presents two controller examples: an EISA Bus graphics system and a Local Bus graphics system. These designs illustrate the simplicity of a Power 9000-based controller.

The controller in figure 13 is a complete Power 9000 EISA Bus graphics system, including bus interface, frame buffer, RAMDAC, and display controller.

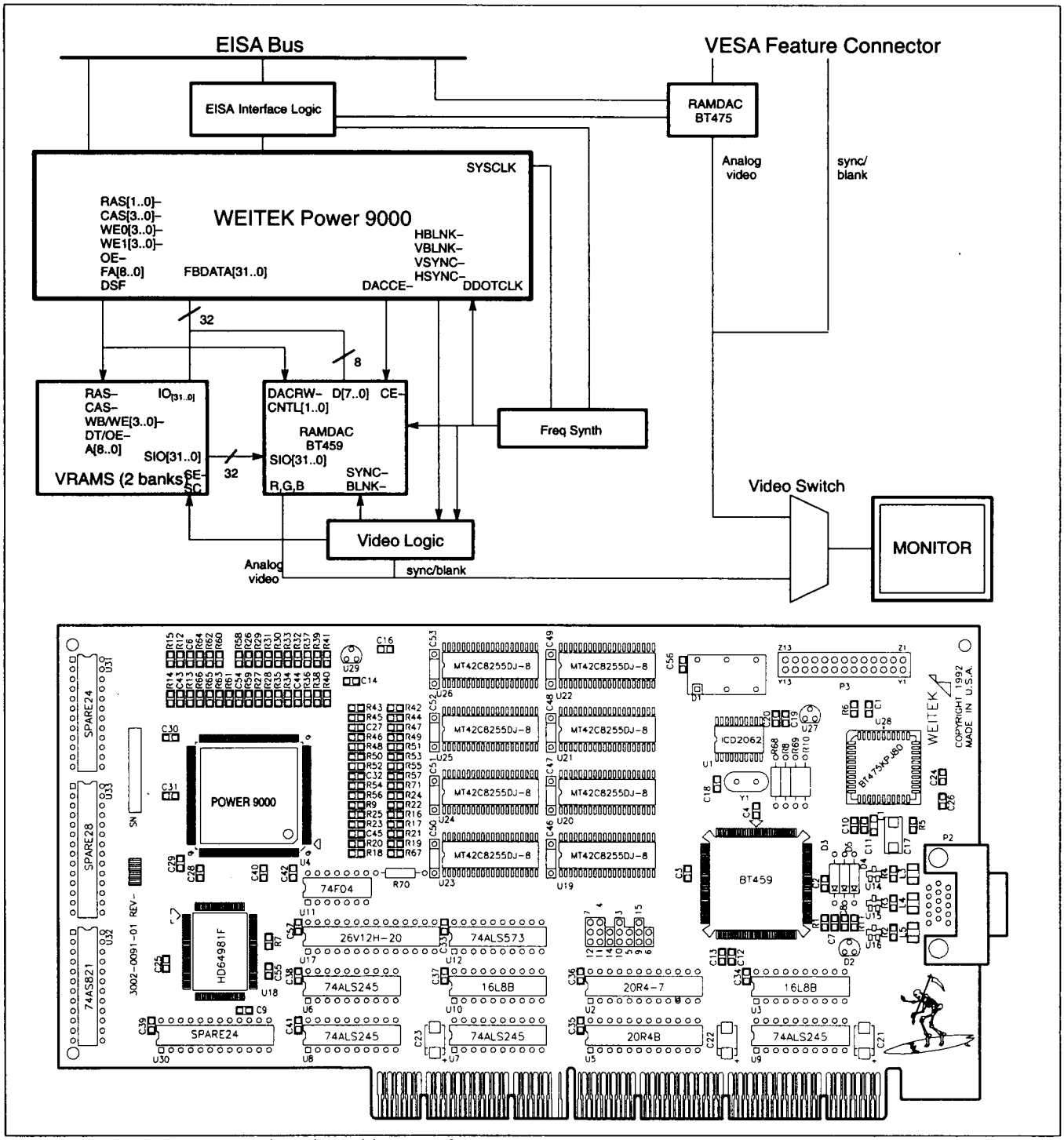

Figure 13. Block diagram and PC board layout of an Power 9000 EISA Bus system

1.7. Controller Examples, continued

The controller in figure 14 is a Power 9000 VESA Local Bus graphics system.

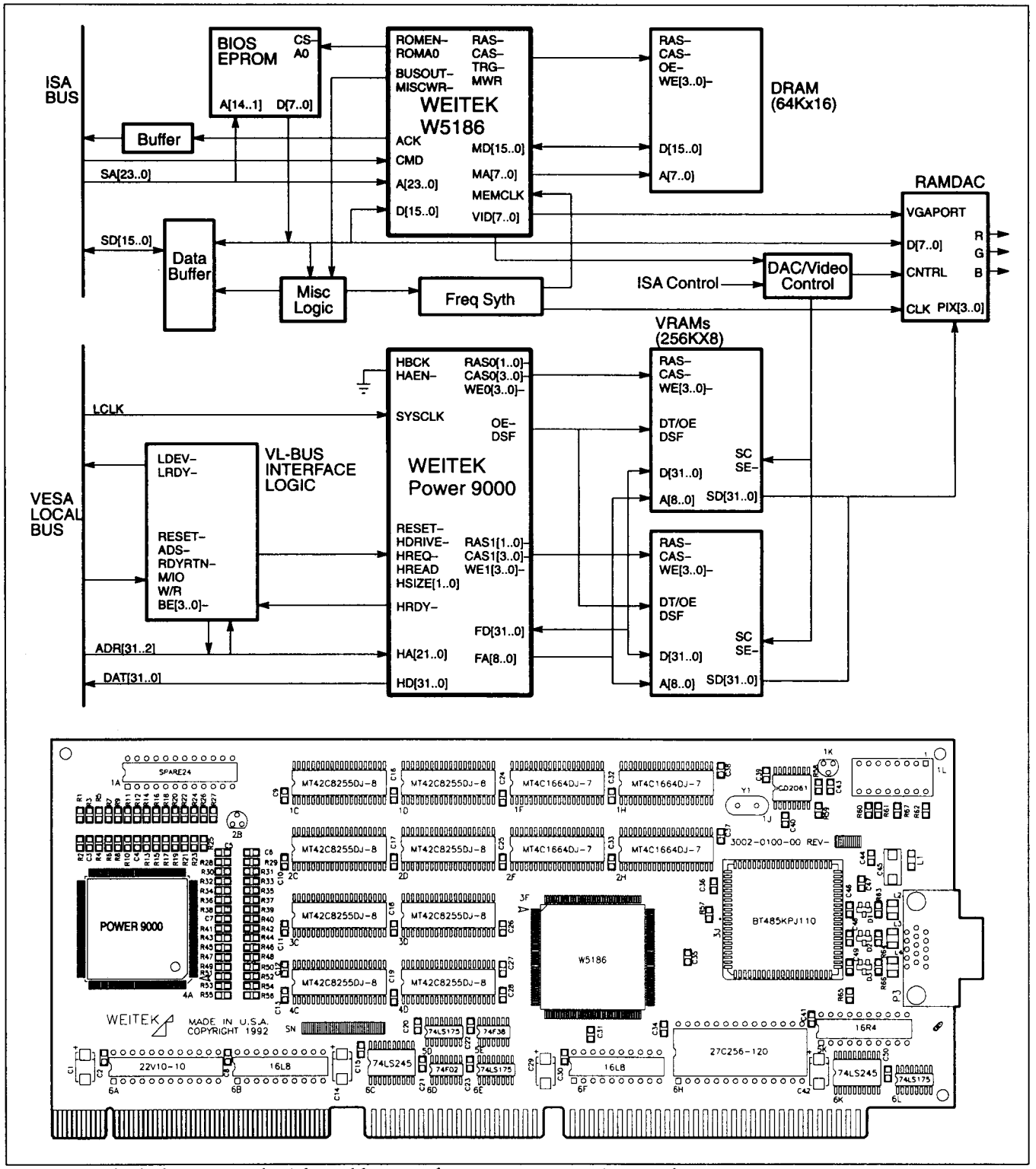

Figure 14. Block diagram and PC board layout of a Power 9000 VESA Local Bus system

1.8. Software and Development Tools

The following development tools are available for the Power 9000:

Microsoft Windows drivers. WEITEK-supported drivers for Windows 3.0 and 3.1.

AutoCAD drivers. WEITEK-supported drivers for AutoCAD R10 and R11.

EISA Bus board. This is the Power 9000 board shown in figure 13, designed for use with an EISA Bus 386-, 486-, or 586-based system.

Local Bus board. WEITEK's two Power 9000 boards are designed for use with a Local Bus 386-, 486-, or 586-based system. Figure 14 illustrates a VESA Local Bus board; an OPTI Local Bus board is also available.

1.9. Specifications

1.9.1. PACKAGE

The Power 9000 is offered in a 208-pin PQFP package.

1.9.2. SPEED

Power 9000 system clock: 25 or 33 MHz

Host interface clock: up to 33 MHz

1.9.3. FRAME SIZE

Frame size is programmable, with resolutions up to two megapixels. Typical sizes are 1600x1200, 1280x1024, 1024x768, 800x600, and 640x480 pixels.

1.9.4. MONITOR REFRESH RATES

Monitor refresh rates are programmable, with typical values being 60 Hz, 66 Hz, 70 Hz, 72 Hz, and 76 Hz.

1.9.5. MONITOR TYPES

The Power 9000 can provide separate horizontal and vertical sync and blanking signals. Pixel data from its 32-bit frame buffer is converted into analog video signals by an external DAC.

1.10. Related Documents

Power 9000 EISA Bus Evaluation Board Application Note. Describes an EISA Bus implementation of a Power 9000 video system.