development work. For downloading into the memory map, the user must replace any ROM/EPROM that may be on the target system with RAM, so that S1-S9 records may be downloaded into it.

For systems that require not only CMOS technology, but a relatively large address space, the MC146805E3 may fill the bill for many users.

Versatile Chip Set Puts Overlay In Your Graphics Design

Patrick J. Svatek

and

Geoffrey Perkins

MOTOROLA INC.

6501 William Cannon Dr. W.
Austin, Tx. 78735-8598

Many video graphics applications require simultaneous text and graphics to be presented on a display screen that has an external source of video information present. Overlay applications typically occur when a camera, video disk or video cassette recorder output requires additonal information to make the picture more useful.

Where are video overlay applications likely to occur? Applications like medical imaging, video micrometry, industrial inspection systems, complex animation, security systems, environmental systems, such as undersea camera work, video printing systems and video disk based recording or educational systems can all use overlay. Each of these systems will place different demands on the text and graphics generating system.

The fundamental problems encountered by designers of microprocessor based overlay graphics systems can be catagorized into five major areas: (1) The Microprocessor interface, (2) the memory interface, (3) the video interface, (4) the overlay interface, and (5) the software overhead in generating text and graphics. Additonally, memory bandwith, CRT limitations, and , of course, performance/price ratio will influence the design.

The solution to this problem is to create a VLSI graphics system in silicon that fixes the hardware architecture, thus minimizing hardware design time, yet allows enough flexibility for the system designer to choose the amount of memory, horizontal and vertical resolutions, the color selection, the choice of microprocessor and the video interface. The next step is to make the system highly progammable so that the design can easily be adapted to a multitude of applications.

The Raster Memory System (RMS) is a two chip set that generates video displays for a full range of performance. The RMS consists of the MC68486 Raster Memory Interface (RMI) and the MC68487 Raster Memory Controller (RMC). The versatility of the RMS lies in its capability to support many modes of operation with minimal hardware impact and the implementation of software tools in silicon, resulting in a significant reduction in microprocessor overhead in generating text and graphics. A designer starts by defining a hardware system and then proceeds to choose the appropriate video mode. Then the system performance is determined by the choice of microprocessor, the amount of dynamic RAM and the level of sophistication of the software.

Several features have been included into the RMS to ease the designer's task. As shown in Figure 1, the RMI provides the master oscillator for the system. It supplies the RMC with three timing clocks, the microprocessor clock, generates multiplexing signals (ADEN,ADSEL,and DBEN) for display and MPU accesses to memory, provides DRAM address generation including transparent DRAM refresh, and decodes addresses in the memory map other than its own. The additional address decoding brought out on the 3-bit S bus allows chip selection functions to be implimented with a 3 to 8 line decoder, thus providing a system and seven more chip selects for peripherals. The X bus shown in the block diagram is used for address multiplexing as well as a data path for the RMC to communicate with the RMI.

THE MICROPROCESSOR INTERFACE

Since the RMS operates with any of three popular microprocessors -the MC6809E, the MC68008, and the MC68000- the RMI will provide Data Transfer Acknowledge (DTACK) handshaking and a 7.95 Mhz MPU clock if a 68000 family MPU is used or Enable (E) and Quadrature (Q) in a 6809E mode. It is the designers choice to have the Raster Memory System generate DTACK for some or all of the system peripherals. The only requirement is

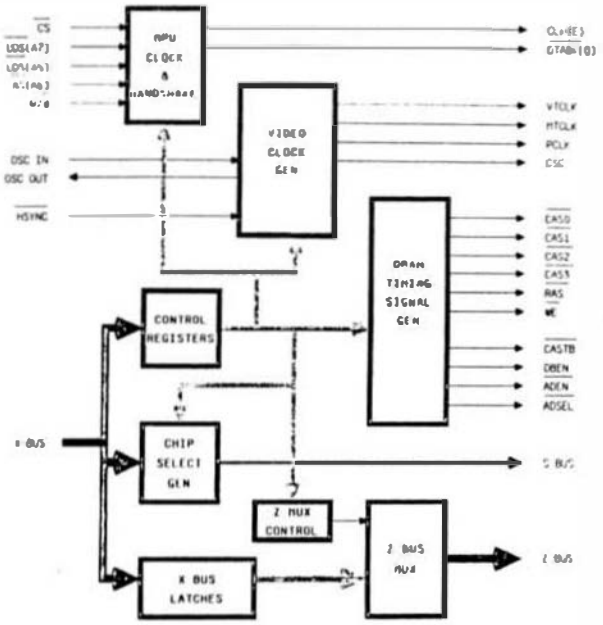


FIGURE 1
RASTER MEMORY INTERFACE BLOCK DIAGRAM

that the peripheral be fast enough to respond to DTACK within the time frame allotted by the system. Any peripheral that does not use the handshaking signals must either inform the 68000 that it is a 6800 family member and will respond synchronously or must furnish its own DTACK.

The other microprocessor interface signals , Address Strobe (AS), Upper Data Strobe (UDS), Lower Data Strobe (LDS), Read/Write (R/W) and Data Bus ENable (DBEN) complete the microprocessor interface. The AS line is a handshaking signal that indicates to the system that the address on ths processor bus is valid and that the processor is ready to initiate a machine cycle. UDS and LDS have two functions: The processor uses them to indicate whether the current cycle involves the high data byte, the low data byte, or both, of a 16 bit transfer. The UDS and LDS signals are also used to inform the RMS that the current processor cycle has ended. R/W connect to both chips and informs them of data direction flow on the data bus. DBEN is used by the RMC for gating the MPU data bus onto Ports A and B of the RMC or DRAM.

THE MEMORY INTERFACE

The system works with several types of dynamic RAMs. A designer can select memory size which can be as little as 16 kbytes in a low end graphics system or as much as 1 Mbyte in the largest system configuration. The type of DRAMs that the system can use are 16k by 1, 16k by 4, 64k by 1 and 256k by 1 and should have a 150 ns access time for proper system operation. The system designer has a choice of one, two, or four banks of DRAM. With a single bank, there will be a loss in performance since the RMS uses the page modes access features of DRAM to improve the data throughput rate. In a 68000 based system, the memory is configured in either two or four banks. Since the memory is not solely used for the display process, cost effective systems can utilize another portion of memory for program storage, stack or scratch pad RAM.

The system operates synchronously to maintain a steady stream of video information to the screen. This basic system timing, called a memory cycle (9 MTCLK cycles), imposes limitations on the microprocessor throughput. In a 6809E system, the MPU is run synchronously with this timing, but in a 68000 type system, the RMS adds two wait states (250 nsec each) to every processor memory access. Additionally, the processor will at times request data at an instant that the system is unable to provide it. While these are limitations, it is unlikely that the design will require a faster version microprocessor because the video memory manipulation capabilities of the RMS substantially reduce the microprocessor's overhead tasks.

During each memory cycle, approximately 1 usec, access to DRAM is multiplexed so that the microprocessor and the display process each get one access. In 8 bit bus systems, microprocessors are allowed to read or write one byte, while a MC68000 can access either a byte or a 16 bit word. The number of bytes fetched by the display process will depend on the number of memory banks used and the type of memory cycle the current access requires. Different types of memory cycles are used by the RMS to handle transparent DRAM refresh, fill list buffers, fill object buffers and arbitrate the type of microprocessor memory access. Data called from RAM is latched into to the RMC via Port A, shown in Figure 2, (and Port B for 16-bit systems only) with a composite CAS strobe (CASTB) generated by the RMI. This raw data will be internally pipelined and processed until it emerges as analog RGB video signals.

THE VIDEO INTERFACE

The Raster Memory Controller, shown in Fig. 2, contains all the circuitry for generating the video timing signals. It also generates all display addresses, passing them to the RMI on the X-bus; receives data from memory; processes that information into pel by pel video information; maintains the DRAM refresh addresses, contains all the registers for programming the RMS, and has an on board display Arithmetic Logical Unit (ALU) that performs real time calculations for screen data manipulation.

One of the most difficult problems encountered in a microprocessor/video design is working out all those stringent timing details. In converting digital data to video, the RMI provides,a video timing clock (VTCLK), a memory timing clock (MTCLK), and a Pixel clock (PEL) to the RMC as shown in the lower left portion of the block diagram. RMC maintains synchronization with the RMI by providing a horizontal sync (HSYNC) reference pulse back to the RMI. Thus all the video timing is defined for the system designer. The designer sets the mode of operation, NTSC or PAL and 6809E or 68000 family, by simply connecting the appropriate X bus line to the system reset circuitry. The RMS maintains proper signal timing for all aspects of the system by clock stretching MTCLK, if necessary, at the end of a memory cycle and re-sychronizing both MTCLK and PCLK to VTCLK at the trailing edge of horizontal sync. VTCLK is a free-running clock and it is never

re-sychronized, while PCLK and MTCLK have their frequency defined by software selecting the horizontal resolution. PCLK varies from 6 to 14 Mhz, while MTCLK is always the master oscillator divided by four plus any stretching for a ninth cycle of the basic memory cycle.

The RMS has programmable horizontal and vertical resolutions allowing the same program to run on a variety of displays. Assume a program created using a high resolution work station monitor must be displayed on a lower resolution system or a home television. If the virtual screen is defined to be the same on both systems, the same data base can be used. The lower resolution system would simply display less of the data at one time and scrolling could be employed to view all of the original screen. There are four horizontal resolutions, 256,320,512, and 640, and up to 10 vertical resolutions, ranging from 192 to 500 lines, offered within the system. Not-interlaced, interlaced sync and interlaced sync and data modes are also software selectable.

applications, it creates many subtly different shades of the same color. In others, it can generate a large selection of divergent colors. With 12 bits defining color, the RMS has a color pallette of 4096 colors. The color selection is limited by the mode of operation: 16 colors in bit plane and 32 colors in list modes. True objects use the color mapping RAM somewhat differently in that they include a transparency option and up to 24 different colors for each object.

Another important tool of the system is the virtual screen, which is used when a picture will not fit onto the displayed screen without a loss of detail. In conventional video graphics systems, the programmmer stores a complete picture in one section of memory, then transfers a subset of that memory to the screen display memory. The technique allows a operator to pan through the picture, concentrating on individual sections. This continual data movement is microprocessor dead time if it is also burdened with complex calculations while it is trying to do screen management

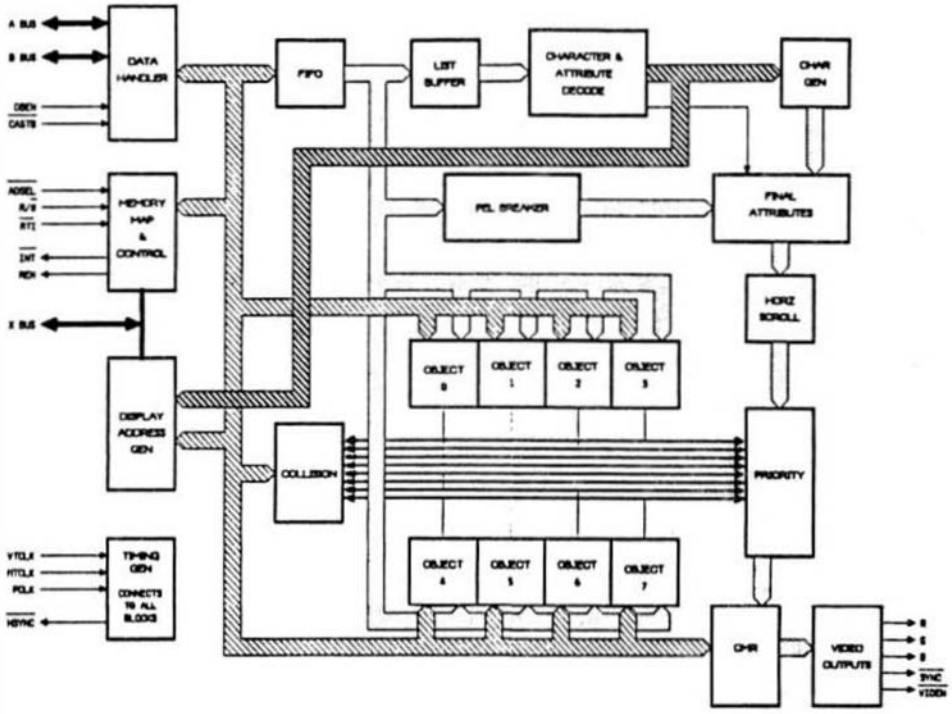


FIGURE 2 RASTER MEMORY CONTROLLER BLOCK DIAGRAM

SOFTWARE TOOLS PUT IN SILICON

All data that is processed within the system results in a 5 bit value that addresses a programmable color look up table or color mapping RAM (CMR) within the final stages of the RMC. The CMRs consist of 32 word locations. The contents of each of these 16-bit locations define a unique color. Twelve of the bits -4 bits each for red, green, and blue can be programmed directly by the microprocessor. An additional bit within each CMR brought out to the VIDEN pin on the RMC. It is used for pixel by pixel switching capability in overlay applications. The color mapping RAM is a powerful tool for controlling a video screen. In some

functions; not to mention operator frustration each time large amounts of data are moved.

The RMS eliminates the microprocessor overhead for large data block movements by supporting a virtual screen in hardware. In standard systems, hardware defined video memory is scanned in a linear fashion, thus the microprocessor has the burden of adjusting that memory with the correct data. The RMS ALU makes real time calculations for each memory cycle fetch of data, thus the displayed memory does need not be manipulated to do scrolling or paging of data.

To use the virtual screen, a programmer starts by defining the height and width of the virtual screen using a few control registers. Then the display screen size is selected by programming the horizontal and vertical resolution. Finally, the address of where the displayed screen starts in the virtual screen is programmed. Once these registers are set, the programmer now has some scrolling options. For one, he can use the smooth scrolling feature of RMS to pan the virtual screen without having to move any bytes of display data. The RMS offers pixel by pixel scrolling both horizontally and vertically for all display modes. A second option might be to use wraparound scrolling, where the display memory is treated as a toroid. That is, if scrolling proceeds long enough in one direction, it will eventually end up at the scroll start location. Yet, another method of scrolling is to treat the screen as a rectangle. When scrolling reaches a border on the rectangle, the screen is filled with a user defined constant, indicating an off screen event.

SELECT THE CORRECT DISPLAY MODE

In converting memory data to pixels, the RMS supports two basic modes: bit-plane mode, where the user can display individual pels, or six text, character, graphics or games modes called list modes. Additionally, independent of the mode, their are eight identical sets of registers to allow simultaneous independent operation of small X,Y positionable windows called true objects. True objects are hardware intensive objects that move around the screen. They have the capability for collision reporting with fixed or other true objects and priority which allows simulating three dimensional effects all defined in hardware.

In the bit plane mode, data in the virtual screen (and display screen) is converted directly to pixel data. (The screen data is always a CMR address). The programmer has the option of selecting 1,2 or 4 bits per pel which in turn determine whether 2,4 or 16 colors can be used, respectively. Each increase in bits per pel (BPP) results in a doubling of memory requirements; a 320 by 210 resolution screen requires 8.4 kbytes at 1 BPP; 16.8 kbytes at 2 BPP; and 33.6 kbytes at 4 BPP. In the bit plane mode, pixel by pixel independence is obtained at a cost of memory and when more memory must be manipulated the resulting drawing rates decrease.

The list modes of the system were designed to use memory efficiently by giving up some but not all of the pixel independence features. List modes can be thought of as indirect addressing modes, where the data fetched from display memory is used as a pointer to another memory location where the pixel data is stored. The pixel data then contains the CMR address for the pixel color. The same 320 by 210 screen described above can be represented by 3.2 kbytes in list mode 4 with 16 colors. This dramatic reduction of memory results when pixel patterns are well defined, such as text and characters. When pixel patterns are more character oriented, the system allows programming of attributes for further character enchancement.

The list modes were optimized to be oriented toward specific applications. Games and animation applications , text and word processing, mixed text and graphics modes, and graphics modes are all supported. Attributes or memory size versus performance trade offs are made by simply choosing the right list mode. For example, list mode 0 offers very little in the way of attributes, but its advantage is the small memory requirements because each byte describes one character. In list mode 4, all three bytes describe one character. This mode mixes several types of characters and offers all the RMS attributes that apply to mixed text and graphics applications or games. List mode 5 is designed for high resolution text applications, primarily word processing, but it also has sufficient graphics capability for other applications.

The programmer must be more careful when using list modes because the display memory may not be contiguous. In the list modes, data from display memory for character description will be one, two, or three bytes. When more than one byte is used to define a character, the additonal bytes will define attributes, multiple characters, multiple characters with limited attributes or one character with extended attribute capability. All the standard terminal attributes like underline, multiple flash rates, invert, double high, and double wide are supported in silicon. Graphics attributes supported include CMR offset, color/resolution selection, foreground/background color selection, and mosaics 4/6 with separation. The true object attributes include priority, color collision reporting, transparency, independent collision enables, and shading. The attributes used for a display screen are fixed by the list mode used for character attributes, while the true object attributes in general are more independent of mode selection.

Three different types of characters are supported by the RMS: alphanumeric, mosaics and redefinable characters. Alphanumerics are generated by an on-board ROM that appears invisible to the microprocessor. Internal routing on the RMC automatically selects the ROM when required. The additional bytes of decoded display data is routed to the attributes section of the RMC. Mosaics are used to create crude graphics with very little memory requirements. Mosaics are allowed to be either 4 or 6 blocks per character size. Actual size of each of the mosiac blocks may vary from 2 to 5 scan lines in size depending on the size of the character block size selected. Character height is programmable for all list modes with 8,10,12 or 16 lines per character available.

Redefinable characters are essentially RAM based characters that allow a programmer to define a custom set of pixel patterns. These characters have their pixel patterns stored in image tables. The programmer determines the size of each of the patterns, calculates the memory requirements based on the resolution and bits per pel used and stores the patterns in RAM. Memory requirements for each character can range from 8 to 64 bytes. Once the first byte location of the first character is programmed into the RMC, the ALU will make all calculations for the pattern fetches, while the programmer only has to refer to them by number. Thus some of the pixel independence of bit-plane is retained and a much easier method of using them in software is available. To support custom character sets, like Kanji, or specialized graphics characters, the RMS allows creation of up to 32,000 redefinable characters in list mode 2.

True objects are hardware intensive objects that move around the screen. In hardware, they have the capability for collision reporting with fixed or other true objects and priority which allows simulating three dimensional effects. Fixed objects are redefinable characters with additional attributes for interaction with true objects. Objects are defined in image tables like redefinable characters. Moreover, the objects can have a base of 3 colors plus transparency for every pixel and there is color offset allowed for each pixel, resulting in 24 usable object colors. Objects can be defined to be from 14 to 49 pels wide by 31 lines and can be expanded by hardware zoom by a factor of 1,2,4 or 8 in both X and Y directions independently.

Overlay with the RMSthe MC1378

To make the RMS operate synchronously with an external video source consider the addition of the MC1378, video overlay sychronizer (VOS), shown in Figure 3. This device contains a complete encoder, i.e. quadrature color modulatore, RGB matrix, and blanking level clamps, plus a complete complement of sychronizers to lock the RMS video outputs to any remote video source. It can be used as a local system timing and encoding source for the RMS or it can be used in the overlay (remote) mode.

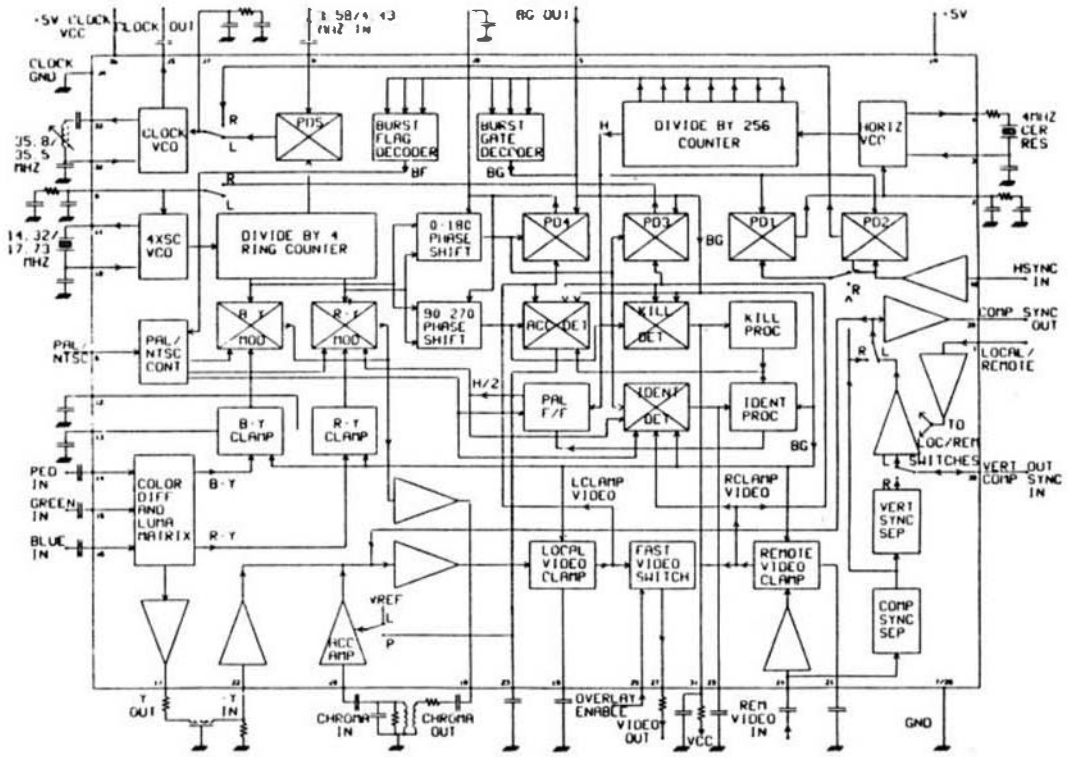


Figure 3 MC1378 COMPOSITE VIDEO OVERLAY SYSTEM

The block diagram in Figure 4 shows the interface between the RMS and the MC1378. Basic system timing for the RMS is derived from the 36 Mhz oscillator on the VOS. The exact frequency is either 8 or 10 times the color burst frequency depending on whether PAL or NTSC timing, respectively, is used. The RMS provides RGB, Horizontal sync, color subcarrier frequency to the VOS. It also controls the Remote/local switch pin for selecting the sychronizing source and the Video Enable pin on the VOS for pixel by pixel switching between the video sources. The vertical/composite pin functions as either an input or output pin depending on the selected mode.

In a local mode, the RMS provides all the sychronizing signals, except the 36 Mhz which the VOS always provides to the RMS. In this case, the RMS provides composite sync via the vert/comp pin. Phase detector 1, PD1, in the VOS (see figure 3) locks the internally counted-down 4 Mhz Voltage Controlled Oscillator (VCO) to the RMS horizontal sync (HSYNC pin). PD2 and PD3 are not used in the local mode. PD4 is active providing an arbitrary phase shift between the color oscillator and the output burst phase. PD5 locks the 36 Mhz RMS clock to the 14 Mhz color oscillator. Therefore the 14 Mhz oscillator is the system standard in the local mode.

In the remote mode, the incoming video signal supplies all the sychronizing information. A phase lock loop (PLL) is locked to the separated composite sync from the remote signal (PD1, horizontal VCO and 256 divider/counter). This produces a noise protected horizontal reference signal. Vertical lock is obtained by continuously resetting the sync generator in the RMC with a separated vertical pulse also obtained from the external video signal. The 14 Mhz VCO is phase locked to the external color burst frequency using a divide by 4 counter and PD3. Phase detector 4 controls an internal phase shifter (0-180 degrees) to assure that the outgoing color burst phase is the same phase as incoming burst phase. The internal

horizontal reference signal is compared to the H sync signal from the RMI and thus forms a PLL locking the 36 Mhz oscillator such that the horizontal sync has the correct phase relationship. PAL identification is obtained from the subcarrier lock system (indentification detector, PAL Flip Flop). The internally generated PAL chroma is then controlled from this identification phase. In the NTSC mode, the identification system is disabled.

Composite Video Generation

The red, green and blue (RGB) analog signals from the RMC are used to generate composite color NTSC or PAL signals. First, color difference and luminance (Y) signals are generated by resistively matrixing RGB. The color difference outputs of the resistive matrix (R-Y and B-Y) then drive two double balanced chroma modulators. The carriers for the modulators are supplied in quadrature from the divide by four ring counter output of the crystal oscillator. The modulator outputs are combined to form the chroma signal and are added to the Y signal along with composite sync to form the complete composite color video signal. In the PAL mode, the R-Y color difference signal changes polarity on alternate horizontal lines.

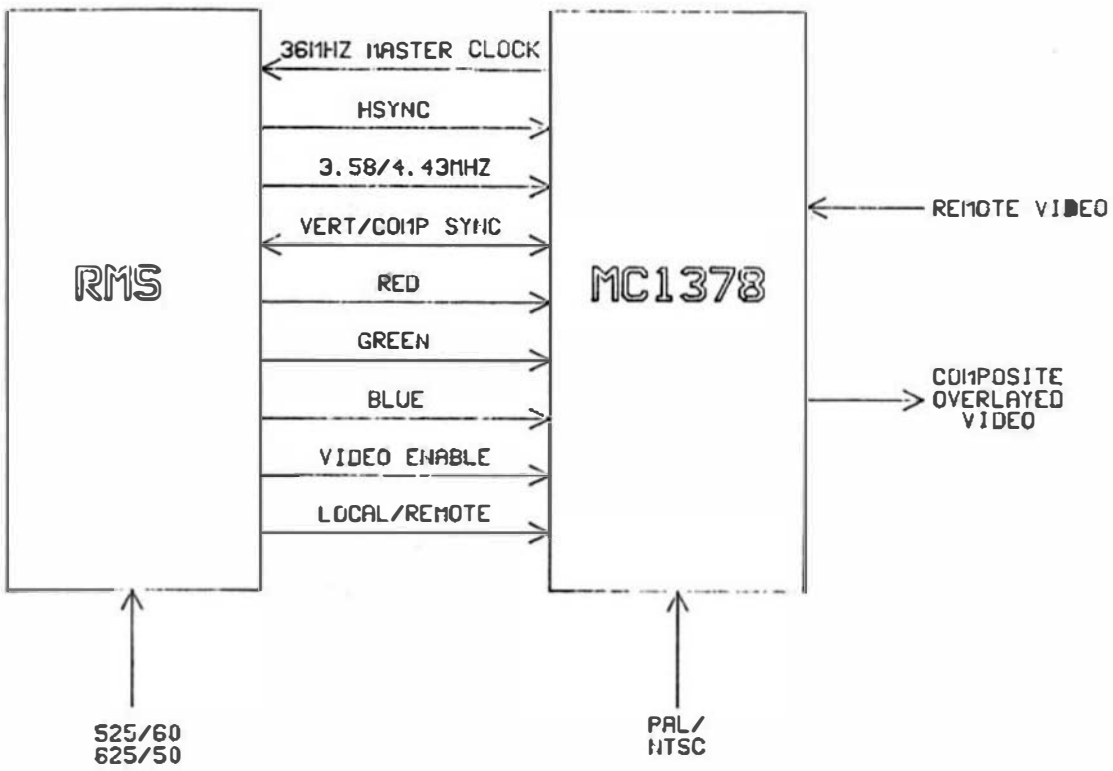
In the remote mode, the phase and the amplitude of the chroma signal is matched to that of the external video signal so that the two composite color signals are completely compatible and may be displayed in an overlay format by operating the video enable change over switch.

Building a RMS based graphics system

A general RMS system based on a MC68000 microprocessor is shown in figure 5. In such a system, the RMS is assigned 1 megabyte of the 16 megabyte address space by decoding the upper four address lines to form a chip select for the RMS. In thie basic configuration, the remaining 19 lines, plus Upper Data Strobe, are connected to the system X bus through three 74ALS257 multiplexers. The 20 address lines are time division multiplexed along with RMS memory accesses using only 10 pins.

The data bus is interfaced to Port A and Port B of the RMC using two packages of 74ALS244 bus drivers and two packages of 74ALS374 octal latches. The dynamic RAM is organized in byte-wide banks. Banks 0 and 1 are used together to address 16-bit words and separately to address bytes. Banks 2 and 3 are used similiarly. Banks 1 and 3 are always the least significant byte of a 16-bit word and are connected to Port A, while the most significant byte, Banks 0 and 2, are connected to Port B.

The system works with a 8 Mhz version microprocessor for which the RMI provides both the clock (7.95 Mhz) and the asychronous handshake, DTACK. In higher performance applications, the designer may supply a separate higher frequency clock to the MPU , however, access to the RMS will always occur at the RMS rates.



The video interface block shown in figure 5 can be one of several types of interfaces. The simplest interface is to buffer the RGB and composite sync then drive a monitor directly. In other applications, the choice might be to create a composite video source, using the MC1377, RGB encoder, or in overlay work, use the MC1378, overlay synchronizer. Additionally, separate horizontal and vertical syncs can be derived, as required by some monitors, providing a variety of ways to interface the output video.

This basic system configuration can be expanded or compressed with minimal hardware impact from application to application. It has numerous system variables defined for the system designer, yet allows enough freedom for a custom design. The basic 68000 system can be implemented in a 68008 system for cost effective applications without sacrificing software efforts. Minimum hardware applications can use the MC6809E microprocessor with only one bank of memory. Again, the RMS will configure the correct timing for that MPU, while the hardware impact consist of changing the MPU interface logic in a predefined configuration.

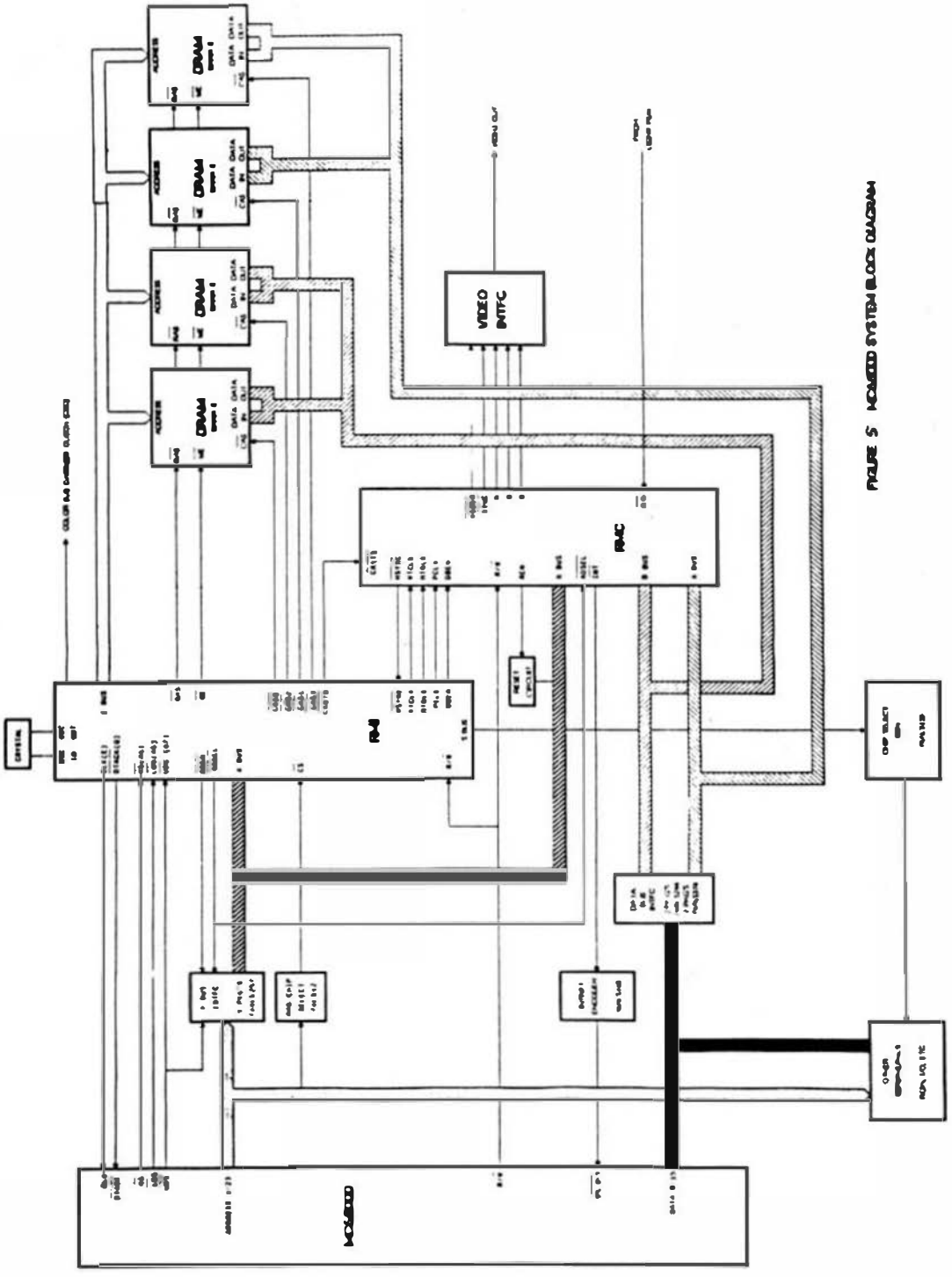


FIGURE 5 MC68000 SYSTEM BLOCK DIAGRAM