A COLOR REAL-TIME ANIMATION SYSTEM

A. S. Malowany, B. Kashef

Department of Electrical Engineering, McGill University

ABSTRACT

Work is underway in the Department of Electrical Engineering at McGill University to develop a parallel processor system that is used by a general purpose host computer to perform real time computer generated animation of displays. The paper describes the system architecture, principles of operation, the communication protocols, and software development. This presentation focuses on the Z8000 implementation and test results. Finally, overall performance estimates of the system are drawn.

KEYWORDS : Real Time, Animation, Color Graphics, Parallel Processing, Microprocessors

1. Introduction.

The Electrical Engineering Department of McGill University has been involved in research projects in Computer Vision, Computer Graphics, and Robotics for a number of years. The laboratory facilities include one DEC VAX 780 and two VAX 750 computers running the VAX VMS and the UNIX multiuser multitasking operating systems, imaging peripherals and a PUMA 260 robot.

In this multiuser research environment, there is a frequent need to generate and display complex computer generated images in real time, for interactive program development. To answer this need, the GRADS (an acronym for Graphics Real-time Animation Display System) has been designed to generate, store, manipulate and display animated colored images on a standard raster-scanned TV monitor (Ref. 1). It uses the frame buffer concept to obtain high density flicker free displays by refereshing the images at a rate of 30 frames per second. The system is capable of displaying 512 * 512 (high resolution) or 256 * 256 (low resolution) pictures having three primary colors and five intensity bits (32 levels) per color. This involves a tremendous information flow of about 8 million 15 bit words per second to the display monitor. By using a parallel processor architecture, the GRADS is capable of animating complex 3-D pictures in real-time. A multiple-bus structure has been adopted to accommodate the large throughput rates that are required for such a system.

The structure of the GRADS may be divided into three levels of hierarchy (Figure 1). The first level includes the Host Computer and the Host Computer Interface module. The second level consists of a set of paralleled microcomputers. Finally, the third level consists of the Graphics Controller, The Video Memory Planes, the TV Sequencer, and the color TV Monitor.

The Host Computer is responsible for supervising the entire system as well as executing the application programs by processing the image at the high level only. The throughput capability of the system is increased by partitioning the high level information to the microcomputers operating in parallel and letting them perform the tedious basic operations for expansion into the detailed information which is fed to the display. The dramatic reduction in price of memory and microprocessor integrated circuits makes this design increasingly cost effective as witnessed by by the recent development of several commercial multiprocessor graphic systems (Refs. 2 and 3).

GRADS is versatile in terms of modularity and expandability. The time-shared bus structure enables us to use the system with a variable number of modules. The system throughput can be enhanced by adding more microcomputer modules or faster video memory modules. One very versatile feature of the GRADS is that both the number of frame buffer planes selected and the display resolution are software programmable. The programmer can reconfigure the system to support black and white displays using a minimum number of memory planes. A choice between "opaque" writing mode which overwrites previous frame content, and "transparent" writing mode, which combines new image data with the previous frame content is available. The modularity of GRADS provides an easier way to design, debug, and maintain this system as compared to designing a single large integrated system.

2. Implementation of the GRADS

2.1 The Host Computers and the Host Computer Interface

Two Host computers are presently used in the GRADS. The role of the Host Computers is to generate the high level commands to be implemented by the array of processors and to control the activity of the system. To be able to handle complex animations, a powerful computer having large main and secondary memories is required. The VAX 11/780 DEC Computer in the image processing laboratory of the University is connected to the Host Computer Interface through the UNIBUS adapter.

A second host computer was provided for the purpose of performing simpler tasks such as debugging and system maintenance. A S-100 based Z80 microprocessor system running CROMEMCO's CDOS operating system is adequate for this purpose.

The HCI provides a fast communication channel to move blocks of informations in a DMA (Direct Memory Access) fashion. The HCI provides the host computers with a path to access the status of the different microcomputers and to control them using an interrupt network.

2.2 The Microcomputer Modules

Each microcomputer module in the GRADS consists of a CPU, a local memory, and local software. The slave microcomputer modules are responsible for executing the flow of graphic instructions downloaded from the Host Computer through the Host Computer Interface and for generating low level commands and data to be submitted to the display subsystem. The GRADS can currently accommodate up to five microcomputers. Each microcomputer is being built around a different CPU. This allows valuable comparisons among different popular CPUs currently available on the market. To date, three microcomputers were built using the 2901 bit slice microprocessor, the Intel 8086 and the Zilog Z8002 microprocessors.

In the current implementation, the Host Computer is responsible for executing the specific application programs as well as performing the necessary three dimensional transformations, windowing, and clipping of the image scene. This results in macro-instructions such as points, lines, polygons, and shaded polygons. The software provided in the microcomputer modules is intended to receive these macro-instructions and produce the output format required by the graphics subsystem. Two input buffers are used at each microcomputer to efficiently accomodate the latencies involved in downloading macro-instructions.

2.3 The Graphic Subsystem

The Graphic Subsystem consists of the Graphics Controller, the Video Memories, the T.V. Sequencer, and the Color T.V. Monitor.

The Graphic Controller is responsible for merging partial image buffers from the parallel microcomputers into the video memory planes. It also interprets the data from the local memory of the microcomputers before storing the corresponding data into the video memory. The interpretation

depends on what graphics mode the system is using.

The Graphics Controller is microprogrammed and currently support four graphic modes :
1) The Point Mode is used to draw many pixels having the same colour.
2) The Solid Area Mode is useful for painting areas in a chosen uniform colour.
3) The Shading Area Mode is useful for efficiently displaying the areas of non-uniform or shaded color.
4) The Read Back Mode has been designed mainly for maintenance purposes allowing the Host Computer to verify the fault-free operation of the video frame memories.

3. Communication in the GRADS.

The GRADS design has sought to incorporate a fast communication network as an integral part of the system. Much of this communication support is offered by the hardware of the Host Computer Interface (HCI). The communications in the GRADS can be divided into two classes : the data communication class and the control and status communication class. To provide fast response times, the two classes have been implemented independently.

The data communication is provided by the HCI which is a special DMA machine. The HCI is capable of doing block moves between buses of different widths and protocols. This machine can be requested by any processor in the GRADS. The DMA machine then takes over and carries out moves in packets of 16 words. The termination of the data block move is signaled to the requester of the DMA by means of an interrupt.

The control communication is carried out by interrupts and control word transfers. The interrupts are used to signify a predefined event by drawing attention to the contents of the status register. Each microcomputer is provided with two special registers, namely the control and the status registers. Typically, a processor wanting to convey some information to the Host Computer, loads its status register with a selected bit pattern and sends an interrupt causing the Host Computer to look at the status register and thus receive the information. The sending of control information from the Host to the microcomputers is done by outputting a specific word, setting the bits of the control register of that particular processor. Some control register's bits are hardwired to the microcomputer systems in order to generate a desired effect such as 'reset' or 'interrupt'.

4. The Microcomputer Operating System.

Each microcomputer module is also governed by its local operating system. The main functions of each microcomputer operating system can be summarized as initializing the microcomputer and managing the local buffer resources. The implementations of the operating system for each of the microcomputers vary slightly from each other as the resources available in each module are not all identical. In this paper we concentrate on the Z8000 module, however the main idea is similar for all the microcomputers in the GRADS.

The software of the Z8000 can be divided into two classes, the Executive and the Macro-programs. The Executive has the control over the entire module from the time this subsystem is started up, by the Host computer, until the halt of the entire system. The Host Computer resets each module by issuing a reset pulse using the control register of the microcomputer. A non-maskable interrupt from the Host invokes the executive of the module.

The Executive first initializes the tables of parameters and switches used for its proper functioning. In the case of Z8000 software, this includes the stack pointer setting, initialization of program status area and its pointer settings. During this time, the maskable interrupts are disabled.

After the initialization, the Executive enables the interrupts and requests input buffers from the Host Computer. This is done by setting the appropriate bits in the status register and issuing an interrupt to the Host Computer. Likewise, immediately after

finishing the processing of an input buffer of macro-instructions the Executive updates the appropriate status register bit and issues another interrupt to the Host Computer.

As soon as a filled input buffer is available, the Executive starts processing this buffer. The macro-instructions will be interpreted by the Executive and the appropriate macro-program will be called for each. The End Macro-instruction defines the end of an input buffer causing the Executive to switch to the next full input buffer.

The Executive is responsible for managing the output buffers. Before calling any macro-program, the executive compares the current size of the output buffer against a chosen threshold length. If this length is exceeded that output buffer is considered full, its actual length is noted and the Executive continues using the second buffer.

When an output buffer is filled, the Executive requests the Micro-Bus, by asserting its private Micro-Bus Request line to the Graphics Controller. The Micro-bus Grant signal engages the graphics controller DMA operation previously explained in section 2.3. At the end of the DMA, the Micro-Bus DMA Done interrupt causes the Executive to free that buffer for further usage.

This programming development was realized in assembly language using the ZAS Z8000 cross assembler by Western Wares under CDOS. Extensive use was made of the sixteen internal registers of the Z8000 CPU for managing loop counters and buffer pointers. The general flexibility of register usage contributed to efficient coding and absence of register bottlenecks. When necessary, the multiple loading and storing instructions made register saving and restoring very easy. The multiple bit position shifting instructions were frequently used for processing fields of the color word. The sixteen bit word size effectively handled the 256 X 256 resolution implementation using byte oriented instructions.

5. Testing and System Performance

The software for Z8000 module was recently completed and tested as shown in figure 2. The Z8000 hardware module is realized using an S100 base system incorporating a Z8000 CPU, 64K of RAM, an I/O card, and a specially constructed interface card to link this module with the Graphics Controller through the Microbus. By using several 8-bit parallel channels on the I/O card, it was possible to connect the system under test to the second Host Computer of the GRADS.

The Host Computer is a S-100 based system consisting of a Z80 CPU, 64K of RAM, an interface card for a video terminal and dual floppy disc drives, and finally an I/O board to communicate with the I/O board of the Z8000 system. Using a test program running on the Host, the performance of the Z8000 module could be easily measured.

5.1 The Test Program

The Test Program performs a number of services selectable from a display menu with several options. First it is used to download the microcode of the Graphics Controller as well as the Executive and the macro-programs into the Z8000 system from the Cromemco CDOS floppy disc files. Another option allows the user to program a timer for measuring the performance of the system by displaying a special character at the time intervals requested by the user on the host console.

The menu also allows the user to select a continuous display operation containing sequences of points, or lines, or polygons. Here the Test Program repeatedly fills the Z8000 input buffers whenever they are emptied. Each routine utilizes a random number generator program to generate random numbers used as the coordinates of points, or the two end points of lines, or the vertices of rectangular polygons as well as their colors. Finally, the Test Program provides the user with extensive messages appropriate to each stage of the program execution.

5.2 Deadlock detection and recovery

In such a networked environment, where the processors are communicating with each other using multiple shared buses, the problem of deadlock is critical. Deadlock may occur because of environment noise problems, hardware malfunctioning, or even design errors in either hardware or software. Any of the above might result in a situation where both the Host and the Z8000 wait on each other indefinitely. For example, the mutual waiting may consist of the Host Computer waiting for request for filling an empty buffer from the Z8000 and the Z8000, in turn, waiting for an input buffer to be filled from the Host Computer. In order to break this tie, a mechanism for detecting deadlocks and recovering from them is essential.

Both the Host Computer and the Z8000 System are provided with time-out mechanisms. If one system is waiting for the other one for more than a specified time-out period a deadlock is declared. By using different time-out intervals for different parts of the system (for example the Graphics Controller DMA transaction or processing of the input buffer of macro-instructions), the existence of each deadlock type can be easily detected and classified. Proper recovery procedures are then automatically engaged. For example to recover from a Graphics Controller deadlock situation, the Host Computer observes the presence of the DMA state. It must issue a reset pulse followed by a non maskable interrupt (NMI) to reinstate the Z8000 CPU operation from its previous tristated DMA mode, and then issue a vectored interrupt to the Z8000 system to inform it to reprocess the same buffer. Appropriate messages are printed for the user by both the Z80 Host Computer Test Program and the Z8000 Executive program to assist the debugging process.

5.3 Performance Evaluation

The performance of the system driven by the Z8000 module operating at a 2 MHz clocking rate was evaluated using an oscilloscope. The Host repeatedly fills the Z8000 input buffers with a set of 2048 points where the (x,y) coordinate of each is randomly specified, or a set of 512 lines where each line is defined by the (x,y) coordinates of its two randomly selected end points, or finally with 256 rectangles where the (x,y) coordinates of the diagonal vertices of each are provided using the random number generator.

Measurements were made to evaluate the Z8000 software performance including buffer management, graphics primitives, and error recovery. This showed average times of 22.5 microseconds per point, 2.3 miliseconds per line, and 12.7 miliseconds per rectangular polygon. Statistics gathered offline showed that the average line contained 106 points and the average polygon had 1000 pixels. This performance could be improved by using hardware capable of operating at the 8 MHz maximum CPU clock rate. A further 10% improvement is estimated possible by program optimization.

During these tests, measurements were also made to assess the performance of the graphics display hardware. These showed that the current system can update, on the average, 16,900 points, 85 lines or 22 polygons in 33 mili-seconds. These slow hardware execution times are primarily limited by the slow access time of the Z8000 memory to Microbus interface card. These rates should improve by a factor of 5 or 6 using the current microcomputer modules presently under development. A further improvement could be realized by upgrading the speed of the video frame memory board from the present 500 nanoseconds cycle time.

As mentioned earlier, a number of such microcomputer modules can be paralleled to enhance these figures proportionally in obtaining a required performance.

Conclusion

In this paper an attempt was made to explain the GRAD System intended for real-time animation and being developed in the Department of Electrical Engineering of McGill University. The parallel operation of the GRAD system,

its communication protocols including deadlock detection and recovery, its software development emphasizing the Z8000 module, and finally the testing and performance estimation of the GRADS are presented. References.

1. Malowany A. S., Levine M. D., "A Raster Color Display System for Real Time Computer Graphics", 6th Man-Computer Communications Conference, Ottawa, May 1979

2. Matrox GXT-1000 "2D/3D Interactive Color Terminal" IEEE Computer Graphics, vol 3, No. 5, Aug. 1983, p69

3. ORCA, "ORCAS for all reasons", IEEE Computer Graphics, vol 3, no. 3, May/June 1983, p16

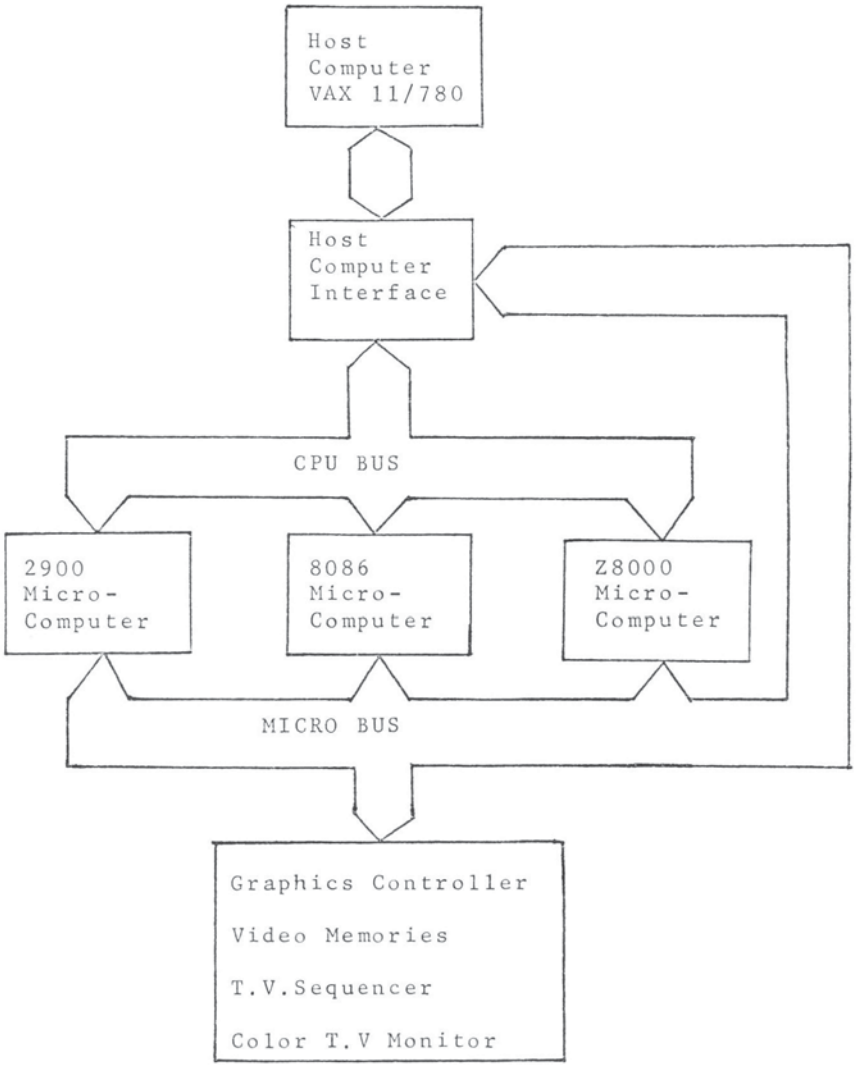

Fig 1. The Architecture of GRADS

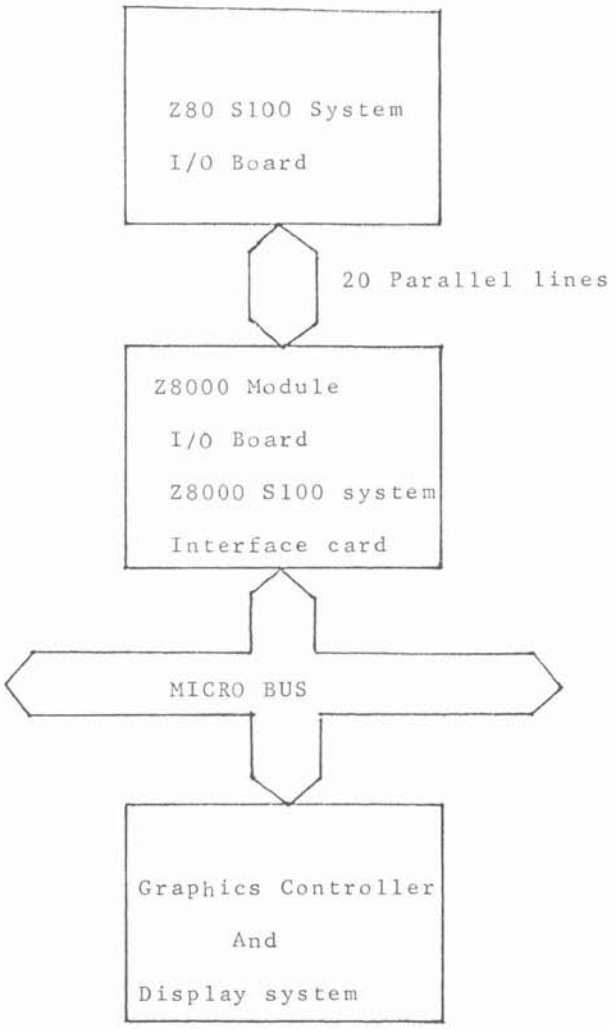

Fig 2. The test configuration