***Yoram Bernet** and **James M. McCarthy** are members of technical staff with AT&T's Data Systems Division in Skokie, Illinois. Mr. Bernet, who is in the Bit Mapped Displays Group, works on graphics hardware design. Currently his work involves low-level graphics software, writing routines to map Graphical Kernel System calls to specific hardware. He joined the company in 1983 and has a B.S.B.M.E. in biomedical engineering from Case Western Reserve University. Mr. McCarthy, who is in the Application Systems Engineering Group, does applications engineering for UNIX® system terminals. He joined the company in 1983, has a B.A. in English from the University of Iowa, and is pursuing graduate studies in computer science from Illinois Institute of Technology.*

THE 630 MULTITASKING GRAPHICS TERMINAL

Introduction

The 630 MTG (multitasking terminal with graphics) implements technology that has its origins in AT&T's Blit[1] terminal and its successor, the Teletype® 5620 dot-mapped display (DMD) terminal. (Panel 1 defines acronyms used in this paper.) With the 630 terminal, substantial progress has been made in display quality, terminal packaging, circuit design, connectivity, and firmware functionality, while the cost of delivering the technology has been greatly reduced. These innovations have led to greatly improved business prospects for this class of device, because several major applications will be delivered to their users via 630 terminals.

This paper describes the advances made in the 630 terminal in three sections: hardware enhancements, firmware enhancements, and new business applications of the device. We assume that readers have some familiarity with the Blit or the 5620 terminal.

The Genesis of 630 Hardware

The immediate predecessor to the 630 multitasking graphics terminal was the 5620 dot-mapped display terminal. In comparison, the 630 MTG provides the following advantages:

- Extremely sharp and clean screen image; ghosting was eliminated.
- Square screen to accommodate both portrait and landscape modes (tall and wide images, respectively).
- Redesigned keyboard, tested and approved by a variety of users.
- Improved esthetics; lighter weight.
- Silent operation.
- Greater I/O (input/output) and expansion capability.
- Lower cost.

The 630 hardware (Figure 1) was specifically developed to provide these improvements. First, we sought an ergonomically superior display at a far lower cost. Second, we could not compromise performance because the 630 MTG was intended for CPU-intensive tasks. (The CPU is the central-processing unit.) Third, we recognized that future requirements would include a high-speed interface such as AT&T's Starlan network or Xerox Corporation's Ethernet™ network. Finally, it was clear from years of user experience that we needed a completely new, and better, keyboard.

To improve the ergonomic quality of the display, we decided to use a 60-Hz, noninterlaced amber or white display. A square CRT (cathode-ray tube) was suggested to accommodate applications written for either portrait or landscape mode. The Blit and 5620 DMD terminals had offered only portrait mode, with a format of 1024 by 800 pixels compared to the 630 MTG's 1024 by 1024 pixel screen. (A pixel is a discrete element, such as a dot, used to form a screen image.)

Based on the cost of components and support circuitry, we selected a 16/32-bit Motorola 68000 CPU, rather than AT&T's WE® 32100 chip found in the 5620 DMD terminal. The use of this chip raised some performance concerns. For example, could it handle increased demands on memory bandwidth because of the larger format display and higher

Panel 1. Acronyms in this paper

ASCII	American Standard Code for Information Interchange
CPU	central-processing unit
CRT	cathode-ray tube
DCE	data communications equipment
DTE	data terminal equipment
DMD	dot-mapped display (terminal)
DRAM	dynamic random access memory
DUART	dual universal asynchronous receiver/transmitter
ECL	emitter coupled logic
EPROM	erasable programmable read-only memory
I/O	input/output
MIC	mouse interface controller
MTG	multitasking terminal with graphics
NVRAM	nonvolatile random access memory
RAM	random access memory
ROM	read-only memory
SSI	standard serial interface
VRAM	video random access memory
VRC	video RAM controller
VSR	video shift register

refresh rate? These concerns led to a decision to use VRAMs (video random access memories) as opposed to standard DRAMs (dynamic random access memories). We will discuss the effect of the VRAM architecture on performance in greater detail later.

The Monitor. Our basic requirements were stringent: 100 pixels per inch resolution, square CRT measuring 16 inches on the diagonal, and capability to handle 44-MHz video data (88-MHz pixel rate). There were no commercial monitors available to suit our requirements.

J. H. Light of the Data Terminal Development Department in Skokie, Illinois, developed criteria for the newly designed monitor including highest possible quality display phosphors, as well as video amplifier characteristics and a geometry that long-term users would appreciate. These criteria were based on user studies with the 5620, as well as other terminals. Eventually, a monitor that met all the criteria was developed exclusively for the 630 terminal.

The new monitor's high video rate, coupled with the phosphors selected, made for an unusually high-quality display with crisp, high-contrast dots. Unlike its predecessors, the 630 screen shows no evidence of phosphor "smearing" or ghosts. The high quality of the screen image is a major outstanding characteristic of the 630 MTG.

Logic Integration. To achieve the required cost reduction, much of the terminal control logic has been integrated into three semicustom gate arrays:

- *Video shift register (VSR)*—The VSR, Figure 2, handles the delivery of pixel information from memory to the monitor.
- *Video RAM controller (VRC)*—The VRC, Figure 3, controls the combined program and video dynamic RAM and generates timing signals for the monitor.
- *Mouse interface controller (MIC)*—The MIC provides the mouse I/O interface. In the 5620 terminal, this interface required several programmable logic arrays.

These semicustom devices represent the key elements in building a powerful, competitively priced AT&T display system. In particular, the VSR and VRC (Figure 4) form a state-of-the-art video and memory system that provides low

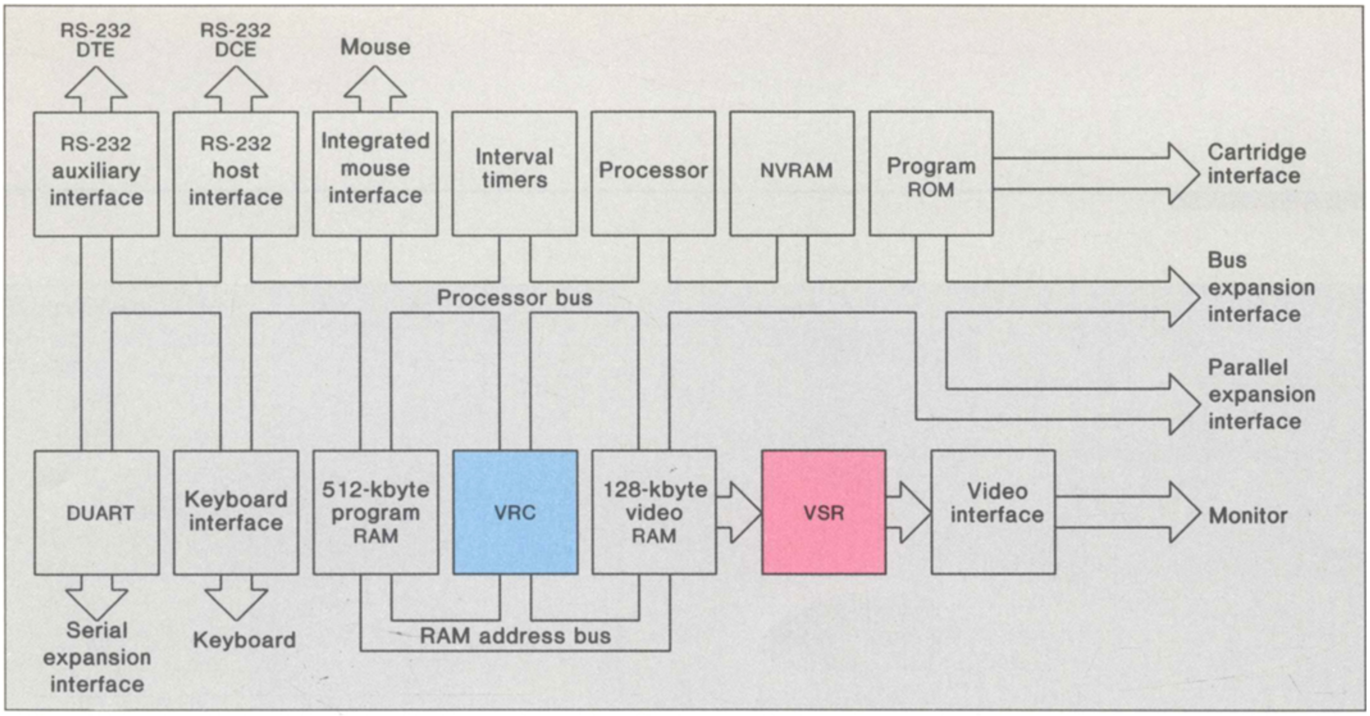


Figure 1. Block diagram of the 630 MTG hardware.
DCE = data communications equipment; DTE = data terminal equipment; DUART = dual universal asynchronous receiver/transmitter; NVRAM = nonvolatile random access memory; RAM = random access memory; ROM = read-only memory; VRC = video RAM controller; VSR = video shift register

cost and high performance both in terms of video quality and program execution.

The Video Shift Register. To realize the monitor's benefits, video data must be provided at a minimum rate of 88 MHz, or about 11 ns per pixel. This requires high-speed video circuitry to retrieve parallel video data from memory, serialize it, and send it to the monitor.

Initial designs used discrete *fast* or *advanced Schottky* bipolar devices capable of internal clock rates in the realm of 88 MHz. Unfortunately, because of the I/O delays from moving data into and out of these devices, a reliable design could not be realized.

We then studied *emitter coupled logic* (ECL) circuits, which typically handle video rates up to 88 MHz. But the cost penalty was too great. ECL requires an additional power supply, discrete support circuitry, and transmission line treatment of device interconnects. ECL devices run hotter and, ultimately, less reliably than standard bipolar logic.

Our last alternative was to integrate the video circuitry into a single high-speed, bipolar gate array and thus avoid the I/O delays encountered using discrete devices. Only the leading edge of bipolar gate-array technology could provide the required video rate. Nonetheless, the cost to implement this design was 80 percent less than a similar solution using ECL. It also had the advantages of both approaches—low cost and high reliability—and still met our speed requirements.

Functionally, the VSR (Figure 2) consists of:

- A *control unit*, the control shift register—A 16-bit ring counter is used as a high-speed state counter to drive load/shift control logic. The control unit also generates the clock for the VRAM serial ports (more on VRAMs later), and for the monitor timing section of the video RAM controller.
- A 16-bit *data shift register*—to convert parallel video data to serial.
- A 10-bit *bus timeout counter*—This miscellaneous function was absorbed into the VSR to free space and components on the logic card.

The first-pass design yielded devices that were 100-percent logically operational. However, at the highest video rate (attained when displaying alternating *on* and *off* pixels),

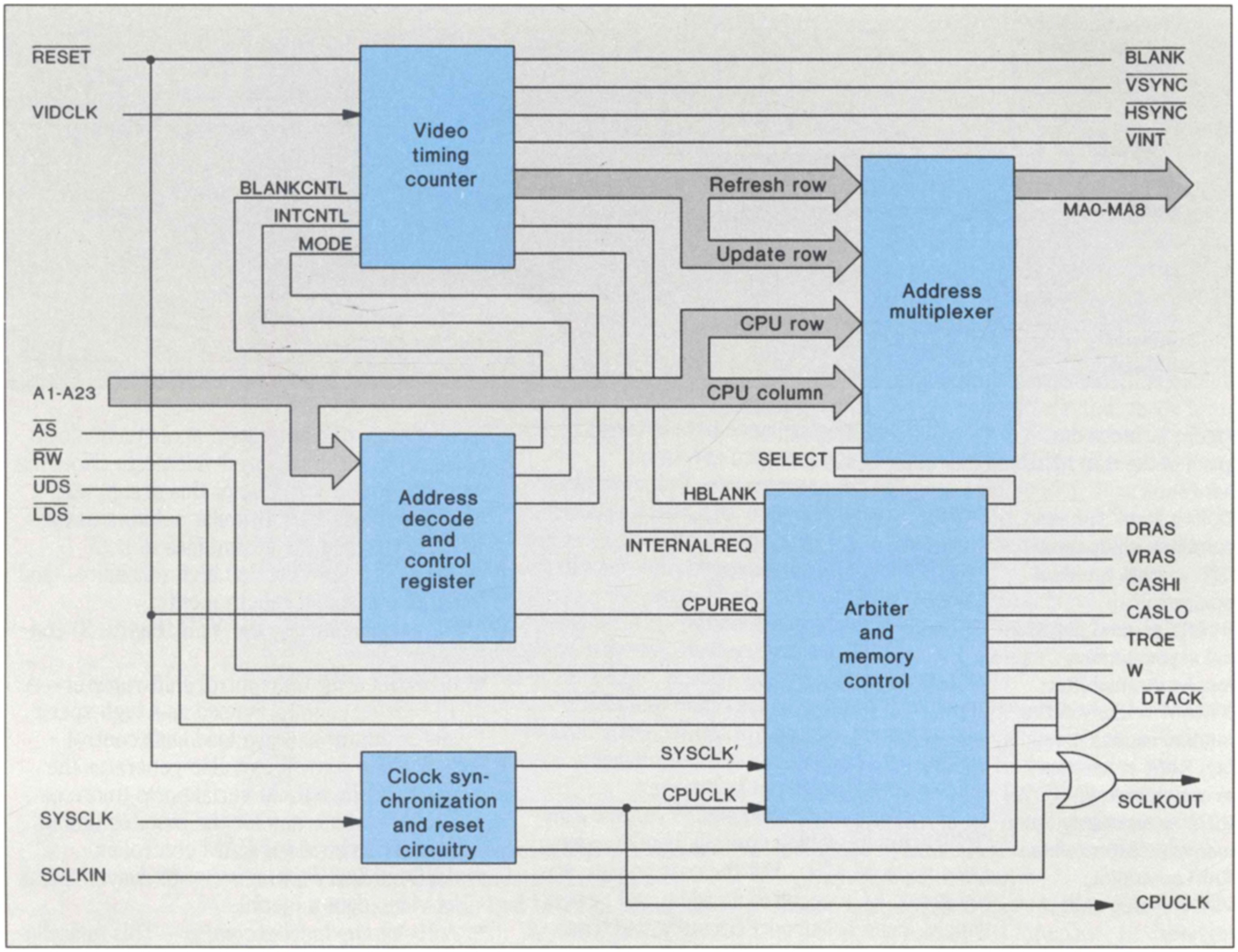
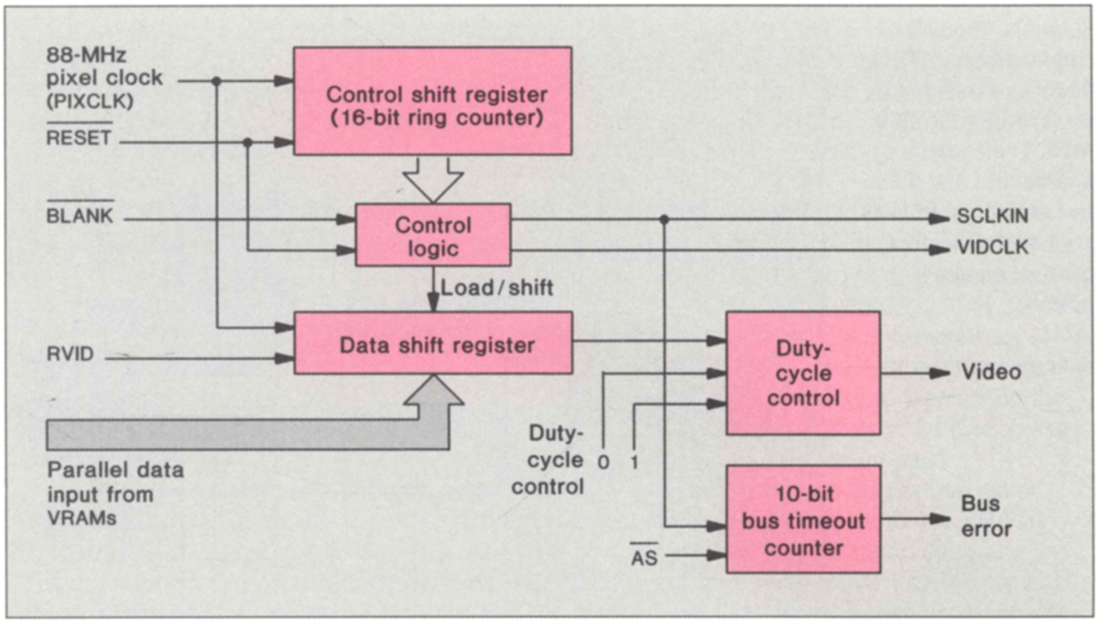


Figure 2. Diagram of the video RAM controller (VRC). Input signals interface to the left side of blocks, output signals leave from the right side. Lines terminated by an arrow represent clock signals or a bus.

Figure 3. Diagram of the video shift register (VSR).


88-MHz pixel clock (PIXCLK)
RESET
Control shift register (16-bit ring counter)
BLANK
Control logic
SCLKIN
VIDCLK
Load/shift
RVID
Data shift register
Duty-cycle control
Video
Parallel data input from VRAMs
Duty-cycle control
0
1
10-bit bus timeout counter
Bus error
AS

the bipolar outputs were analog in nature, rather than digital. The video signal, which should have resembled a square wave, was closer to a sawtooth, with a reasonably fast high-to-low transition but a slow low-to-high transition. This results from the inherent speed limitations of the bipolar process.

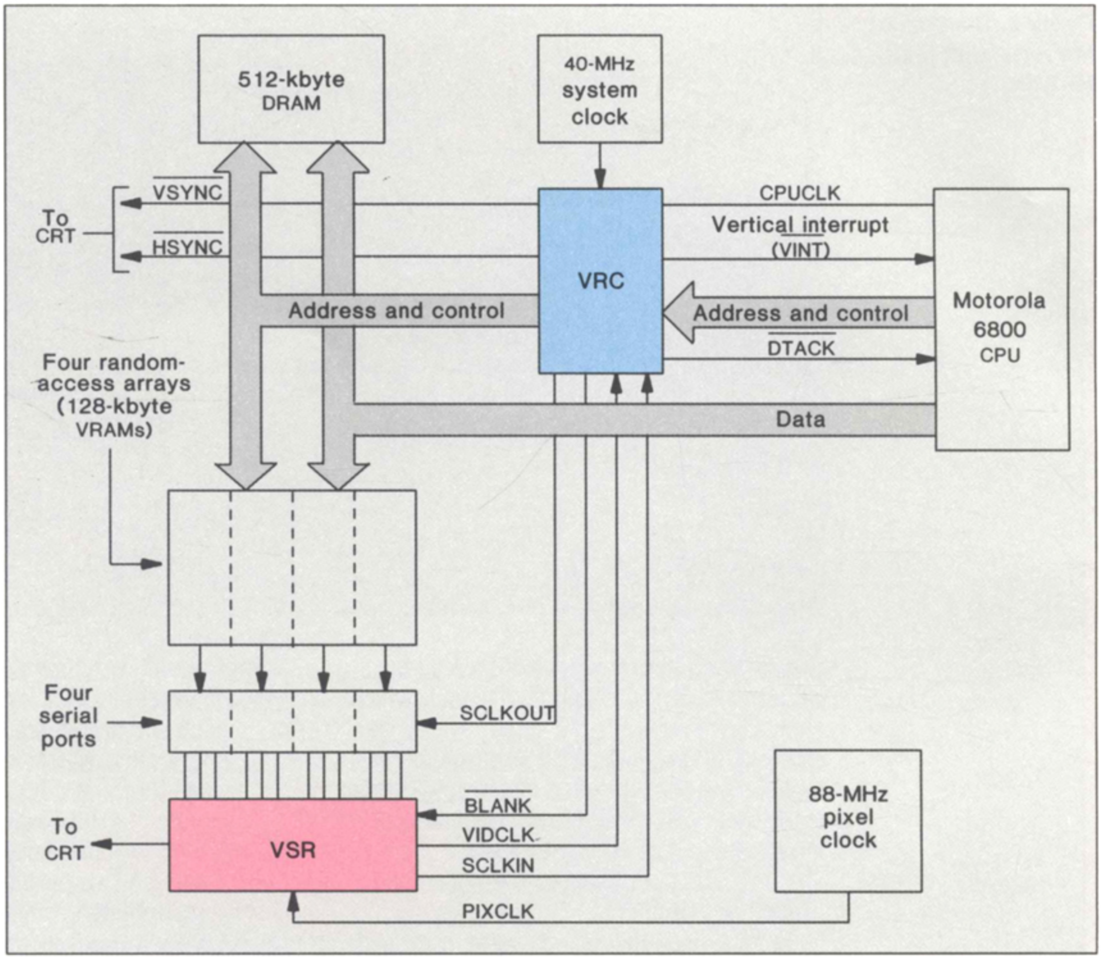
The poor shape of the video signal caused two problems:

- Vertical lines that are only one pixel wide appeared dim, in comparison to horizontal lines of the same width. (When displaying a single-pixel vertical line, the video signal makes a low-to-high transition followed immediately by a high-to-low transition. Thus, the slow rising edge of the video signal "steals" a significant part of the *on* time of the CRT beam, appearing as reduced brightness.)
- Internal switching noise appeared at the video output as amplitude modulation of the video signal. On the display, this appeared as a periodic variation in brightness across scans (horizontally). This period corresponded to the 16-pixel load/shift cycle of the video shift register.

Actually, the two problems were interrelated. With fast-enough transitions on the video signal, minor variations in amplitude (provided the minimum switching threshold of the digital monitor inputs are met) should not affect the display's brightness.

Given that the video output's slow rising edge was an inherent limitation of the process, we needed a solution to increase the video signal's duty cycle (*on* time). In addition, it was necessary to reduce the output signal's modulation by internal switching

Figure 4. The video RAM controller (VRC) and video shift register (VSR) for the 630 MTG. The address-and-control and data buses and the SCLKIN signal drive each of the four memory devices.
VRAM = video random access memory.



noise, or to buffer the monitor inputs such that the monitor circuitry would be less susceptible to variations in video signal amplitude.

To increase the video signal's duty-cycle, the VSR was modified. Four paths were provided from the last stage of the internal video shift register to the output pin. Each path incorporates logic cells that have different low-to-high and high-to-low transition characteristics. By applying static levels to two external *duty-cycle control* pins, one of four paths can be selected. In this way, the characteristics of the video output can be compensated to yield an effective duty cycle close to 50 percent. This added flexibility may prove to be useful in compensating for potential variations in monitor characteristics.

To tackle the amplitude modulation of display brightness, we used two methods: increasing the number of ground pins to the gate array, and widening the ground buses internally (especially near the video output). This helped reduce the noise amplitude at the output. We used an external Schottky diode at the video output, as a level shifter. This causes the following stage to reach its switching threshold earlier in the low-to-high transition of the video output (where the signal slope is greater and amplitude variations have less effect).

Memory Architecture and the VRC. The 630's memory architecture is similar to that of its predecessors in that display memory and program memory are combined in contiguous memory space directly accessible to the CPU. This approach allows the CPU to manipulate pixel data directly, with freedom to draw almost anything.[2]

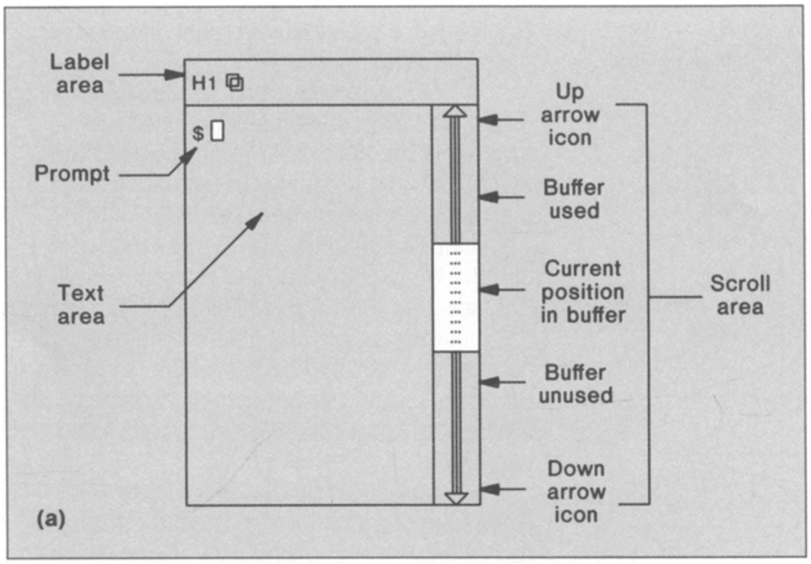
The downside of this architecture is the typical dual-ported display memory bottleneck. Maintaining the display (display refresh) requires continual access to display memory, and while display refresh is going on, the CPU cannot access display memory. Thus, the CPU will often be forced to wait for a display access to complete.

The situation is bad enough when the CPU is forced to wait only for data accesses. But in a terminal such as the 630 MTG, much of the executed code is downloaded into the terminal's memory and hence must reside in the same dynamic memory as data. Thus, the CPU must wait for display accesses to complete, not only on data accesses, but on instruction accesses as well. If the CPU is memory bound and not processing bound, this waiting translates into reduced performance. In the 5620 terminal, for example, display refresh steals 33 percent of the memory bandwidth, resulting in 33-percent lost performance.

One solution that provides partial relief from the display bottleneck is to use different controllers for program memory and display memory. Unfortunately, adding another memory controller can increase cost significantly. Furthermore, this solution provides only partial relief; the extent depends on the mix of data accesses versus program accesses for the particular tasks of interest.

Instead, for the display portion of dynamic memory, we chose to use VRAMs, devices designed specifically to eliminate the display bottleneck. Because VRAMs are dual-ported memories, the CPU can access their contents in standard fashion via port A, while display accesses are being executed simultaneously via port B, which is really a large shift register. As long as data resides in the shift register, CPU and display accesses may proceed concurrently. When the shift register contents are exhausted, the memory controller (described later) must access port A to reload the shift register. In our application, such an access is required every 4096 pixels (as opposed to conventional DRAM display architectures, which require display accesses every 16 or 32 pixels). As a result of this architecture, the 630's CPU sees a memory bandwidth loss of less than a half percent for display accesses. The inevitable 2-percent overhead required to refresh regularly the dynamic cells of any dynamic memory leaves a net CPU bandwidth of 97.5 percent.

While we increased the availability of



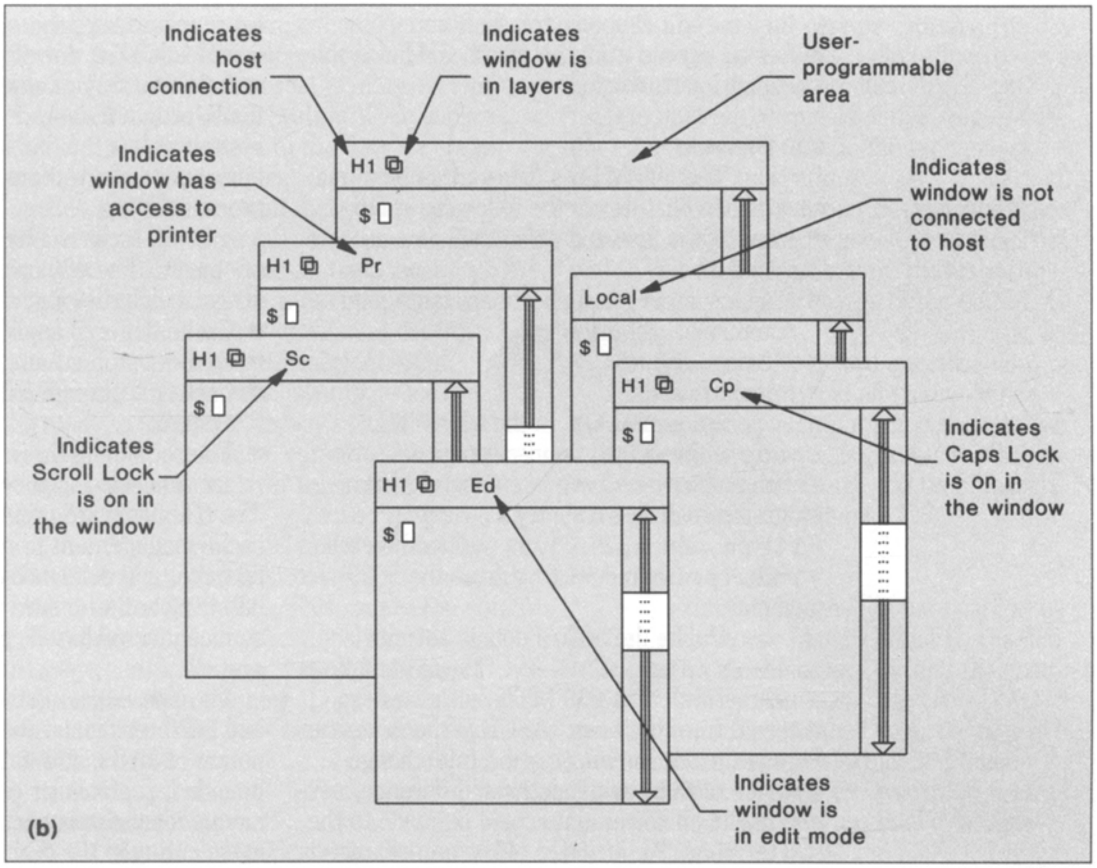
memory to the CPU, it was also highly desirable to guarantee zero-wait-state accesses. This means that memory, when it is available, should supply data to or accept data from the CPU as fast as the CPU can handle it. This requires a fast controller and arbitration circuit. For each CPU request, the arbitration circuit determines if memory is available. In general, such a controller and arbiter will be implemented using a large number of discrete devices, especially in our case where off-the-shelf integrated controllers that were suitable for the then state-of-the-art VRAMs were not available. (One such integrated controller was available but was costly and provided poor performance.)

To save costs in hardware and provide the speed required, we chose to implement the memory arbiter-controller circuit in a second high-speed gate array. This array is more complex than the video array but has a maximum operating frequency of only 50 MHz.

The memory controller was designed to squeeze performance from the memory system, which includes 512 kbytes of standard DRAM and 128 kbytes of VRAM. It provides zero-wait-state accesses for CPU speeds up to 12.5 MHz, and overlaps consecutive accesses to the degree possible. To achieve this performance, the memory controller generates the CPU clock, maintaining synchronization between memory and CPU.

The memory controller also provides all the necessary video timing signals and automatically schedules and executes display accesses and dynamic-memory refresh cycles. By using common state counters to control video timing and schedule DRAM and display refresh requests, we were able to fit the necessary functions into an inexpensive, 600-gate array, which required manual routing to exploit over 95 percent of the available logic.

Figure 5. Window presentation and user interface. (a) Each window has its own label (status), text, and scroll areas. (b) Typical information supplied in label area. Windows may overlap and have different shapes.


Indicates host connection
Indicates window is in layers
User-programmable area
Indicates window has access to printer
Indicates window is not connected to host
H1
$
H1
Pr
$
Local
$
H1
Sc
$
H1
Cp
$
H1
Ed
$
Indicates Scroll Lock is on in the window
Indicates Caps Lock is on in the window
Indicates window is in edit mode
(b)

Miscellaneous. The memory and video system discussed above forms the heart of the 630 hardware. In addition, up to 384 kbytes of EPROM (erasable programmable read-only memory) may be provided on the 630's motherboard (main circuit board), and an additional 384 kbytes of EPROM may be provided on a removable cartridge pack. The cartridge can be configured to append to firmware on the motherboard or logically replace it, thus providing an easy means for integrating terminal emulations.

Two RS-232 ports are provided on the motherboard, and an I/O expansion connector will accommodate an additional RS-232 port and an SSI (standard serial interface) port for connection to the AT&T 6544 Multifunction Communications Controller. An integrated mouse interface that uses a third gate array is also provided.

To provide the high-speed interface, we used a bus expansion connector in addition to the limited I/O connector. This connector makes all signals from the 68000 CPU available to expansion cards.

630 Firmware

The 630 MTG's firmware implements several new features in the following areas:

- New, highly featured default window process (wproc)
- Cache system for applications, fonts, and commands
- Message queues
- Local windows.

Window Process. In the 630 MTG, wproc is the default window process. Some features of wproc have been derived from programs such as mux, myx, emacsterm, viterm, and hp2621, as well as the earlier windowproc and vitty from the 5620 terminal.

Unlike the 5620's default terminal emulator, wproc stores ASCII representations of characters in the 630 MTG as they are received from the host. (ASCII is the American Standard Code for Information Interchange.) Because of this basic functional difference, several useful enhancements could be made to the 630 terminal. We included off-screen character buffering in wproc, along with a scroll bar (Figure 5a) for simplified access to the buffer contents. A local editing mode also provided mouse-based positioning, text sweeping, and interwindow cutting and pasting. New escape sequences were added to program menus and window labels (Figure 5b) and to reshape windows. Multiple fonts were made resident in EPROM.

Cache System. The basic idea of the 630 cache system is to download information from a host computer and store it in the 630's RAM for repeated use, thereby avoiding multiple downloads. The downloaded information may include a variety of objects such as applications, program data, or fonts. The cache system allows the 630 MTG to store these objects and share them among numerous 630 processes.

A *cache* is a named, globally linked list of objects. The 630 cache system supports an array of eight distinct caches:

- Cache 0 stores applications.
- Cache 1 stores fonts.
- Caches 2 through four are reserved by the system.
- The remaining three caches (5 through 7) are available for general applications.

The 630 firmware provides a collection of cache management primitives (e.g., store, retrieve, and delete objects from the designated cache) to ensure the utility of this semiconductor-based, general-purpose storage system.

Messages and Local Layers. From the Blit and 5620 terminals, the 630 MTG inherits the notion of *layers*. A 630 layer is a visually bounded, rectangular bitmap that provides an independent display context for a process that is executing in the 630 MTG. Typically, the 630 process maintains a connection to a complementary host process.

One can visualize the 630 application program as executing *in a layer*. By default, this application program is an asynchronous terminal emulator (wproc), and that layer becomes a *virtual terminal* with respect to the host process. As one replicates this type of layer, many host programs can simultaneously connect to several virtual-terminal layers multiplexed onto a single physical display.

To provide programmed interaction

among processes that are executing in different layers, the 630 MTG supports an interprocess communication scheme that is modeled after messages under UNIX System V. An application that is downloaded to the 630 MTG can *request* affiliation with one or more message queues and can then *wait*, based on changes in the relevant message queue or queues. Typically, other 630 processes initiate these changes by invoking the message primitives used to create, send, receive, and control messages and queues of messages. Through messages, events in one layer or process can be made to affect the display of another layer. For example, the mouse cursor reaching a particular region in one layer might trigger an event in another layer.

Many applications have no need to communicate with a host once they are downloaded to the 630 MTG. This causes the notion of *local layers*, i.e., applications that download to a layer and call a function to "disconnect" themselves from the host. Local layers are allocated dynamically; their number depends on available RAM.

Applications

The most significant market for 5620 DMD terminals was the technical professional—in particular, software developers. The 630 MTG will meet all requirements for this environment.

From its inception, however, the 630 MTG had significant new target applications for operations systems, notably in AT&T's special-services world (preservice and maintenance), where thousands are being installed. Because of its large and high-quality screen, convenient user interface, potential for graphical representation of complex objects (like a private-line circuit), and local processing capacity, the 630 MTG has the potential to help reduce costs and substantially increase productivity. It can replace multiple terminals with a single, windowed display and provide the technology base to deliver new applications.

AT&T's Customer Service and Sales Centers are planning to use several thousand 630 MTGs for telemarketing. In this application, the essential strengths of the 630 MTG include:

- The screen and keyboard interface does not become uncomfortable after long periods.
- Multiple windows into many databases can be opened to provide fast customer service.
- Considerable processing will be offloaded from the host computers.

Summary and Future

Primarily because of hardware integration and the use of a 16-bit CPU, the 630 MTG can be manufactured for less than half the cost of its predecessor, the 5620 DMD terminal. Even at the reduced cost, the 630 MTG offers the user a larger format and higher-quality display. Despite the transition from a 32-bit CPU in the 5620 terminal to a 16-bit CPU in the 630 terminal, the 630's hardware performance is comparable. (Further performance improvement could be realized using a 12.5-MHz CPU.)

To address the high-speed interface requirement, a prototype Starlan/Ethernet/RAM card has been designed. This network card interfaces to system memory via the enhanced bus connector and semicustom memory controller.

Several significant firmware enhancements have been introduced on the 630 MTG. Also, an expanded programming environment is being explored for the 630 MTG. The intent of this environment is to provide C language linkages to a subset of the UNIX system library routines, increasing the portability between UNIX programs and those of the 630 MTG.

As higher speed connectivity becomes a reality for the 630 MTG, additional applications will be explored that take advantage of the increased bandwidth. For example, the newly emerging X-Windows System[3] industry standard works best with higher speed connections among hosts and display systems. This technology is a likely candidate for support in a 630 environment when the 630 MTG is connected to hosts via a higher speed network. The 630 MTG would then be able to support the applications packages that are being designed to work with the X-Windows System.

The user community for AT&T's operations systems will be among the first large group of 630 terminal users. We expect the internal user base to grow as older systems are delivered in 630 layers—instead of their current stand-alone antiquated terminals—and new systems are developed to exploit the 630 MTG's resolution and processing power.

Acknowledgments

The authors would like to recognize the contributions of the development organizations that were involved. George Zobel's group developed the hardware for the bit-mapped display, while Dennis Dugan's group was responsible for the firmware. Jeff Light developed the monitor, and Tom Collura's group was responsible for the product requirements.

References

1. R. Pike, "The Blit: A Multiplexed Graphics Terminal," *The UNIX System, AT&T Bell Laboratories Technical Journal,* Vol. 63. No. 8, Part 2, October 1984, pp. 1607-1631.
2. R. Pike, B. Locanthi, and J. F. Reiser, "Hardware/Software Trade-offs for Bitmap Graphics on the Blit," *Software Practice and Experience,* Vol. 15, No. 2, February 1985, pp. 131-151.
3. R. Scheifler and J. Gettys, "The X Window System," *ACM Transactions on Graphics,* April 1986.

(Manuscript received September 2, 1987)