SLAC-199
UC-32

THE INTERNALS OF THE VIDEO GRAPHICS TERMINAL

LEONARD SHUSTEK

STANFORD LINEAR ACCELERATOR CENTER

STANFORD UNIVERSITY

Stanford, California 94305

PREPARED FOR THE

ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION

UNDER CONTRACT NO. E(04-3)-515

December 1976

Printed in the United States of America. Available from National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Road, Springfield, Virginia 22161. Price: Printed Copy $4.00; Microfiche $3.00.

TABLE OF CONTENTS

1. INTRODUCTION

The Stanford-SLAC Video Graphics Terminal ("VGT") is an experimental computer terminal which is intended to provide low cost, high quality graphics and programmable text processing by taking advantage of the latest memory and microprocessor technology. The basic characteristics of the terminal and an overview of the internal organization have already been described in [1], and the reader is encouraged to have read that document before proceeding. This technical memo is a more detailed description of the prototype that has been constructed; logic drawings and program listings are included as appendices.

This is NOT intended to be the instruction manual for the assembly of a "VGT kit". Information as detailed as logic drawings and program listings have been included only to serve as an example of one approach to the implementation of such a device. No particular care has been taken to ensure that the information contained herein is entirely complete, consistent, or correct. Any attempt to mechanically reconstruct the VGT using this information and without a thorough understanding of the entire device is most vigorously discouraged.

One important reason for such advice is that there are better ways of implementing many of the ideas incorporated in the design. There would be many things I would change in the next version of the VGT; partly because of gained experience and partly because even in the short time between the design and construction of the VGT prototype (Fall 1975) and now (Spring 1976) the fast-moving IC technology has changed enough to affect some decisions made in the course of the design. "No design is so complete, that a redesign can't improve it." [2].

2. FUNCTIONAL OVERVIEW

The display device for the VGT is a standard 525-line television monitor which accepts a single composite sync and video signal. A fully interlaced display of 482 lines is generated, and to minimize visible flicker a long-persistence P39 or P40 phosphor is recommended. On each of the 482 visible raster lines there are 648 dot positions which can be bright or dark. The ratio of 648 to 482 is approximately the same as the 4 to 3 aspect ratio of most monitors, so the dot density is the same in both coordinate directions (i.e. a 10-dot by 10-dot box is a square).

The VGT operates in either of two modes: "TEXT MODE" for text-only displays, and "GRAPHICS MODE" for displays with both text and graphics. The entire screen is always in one mode or the other, and the mode can be changed by the local microprocessor at any time in response to keyboard or remote computer commands.

The main storage medium in the VGT is a 48K byte random access memory whose use depends on the operating mode. In GRAPHICS mode each of the 312236 (482 x 648) bits visible on on the screen is represented as a single bit in the RAM. The local microprocessor (hereafter called the "CPU") can therefore create, erase or modify graphic images by changing the contents of a segment of its memory. The software currently implemented simulates the Tektronix 4010-series terminals [7] and draws vectors and points in response to commands which are received from the serial interface.

In TEXT mode each byte in the RAM is taken to be the ASCII code for a character and completely specifies the pattern to be displayed in a small rectangle on the screen. Each character position is eight bits wide, so 81 characters are displayed in a single row. The character height can be selected to be 13 raster lines (which allows 37 character rows on the screen) or 16 raster lines (which allows 30 character rows on the screen). The standard 128 ASCII characters are predefined by a standard ROM with a 7 by 9 dot matrix, which is positioned within the 8 by 13 rectangle. There is a memory (separate from the 48K byte display memory) which can be loaded with up to 256 arbitrary character patterns. Each such character can be as large as 8 by 16 dots.

Much less memory is used in TEXT mode to display a single screenful of information than is used in GRAPHICS mode. (For 13-line characters, only 37 x 81 = 2997 sequential bytes in memory are used). The rest of the memory is used to save text which is not currently being displayed. At the start of each frame the CPU can specify both the character at which the display is to start and the raster line within that character which is to appear at the top of the screen. By changing those values every few frames, the image can be made to scroll at a rate which is controlled by the CPU. Instantaneous changes of displayed text can also be made by moving the display to another section of memory. In normal use as a computer terminal, text is made to appear at the bottom of the screen, existing lines are scolled up one position, and the top line disappears from the screen. Keyboard commands are used to scroll the text at user-selectable rates, or to move to other sections of text.

Devices for communication with the external world consist primarly of a keyboard and a serial interface conforming to RS232C specifications [3]. Other devices include an A/D converter for joystick or handheld mouse input, a Versatec printer interface, and a special rotating control for scrolling.

3. THE HARDWARE

3A. General Organization

Figure A.1 of Appendix A is an overview of the hardware organization of the VGT. The terminal may be considered to be a combination of two asynchronous semi-independent processors sharing a large common memory. The control processor, which is an INTEL 8080A microprocessor, has the responsibility for managing the contents of the large memory which contains the text and graphics information, and for communicating with the outside world. The display processor, which is a collection of counting registers and random logic, is concerned with maintaining the image on the raster-scan TV monitor by using the data in the memory and, possibly, one of the two character generators.

The control processor is a conventional microprocessor system consisting of a 16-bit address bus and an 8-bit bidirectional data bus. The PROM program memories and a

small RAM for local variables (called the "Local RAM") can be accessed only by the CPU and are connected directly to the CPU busses. Similarly the I/O devices (keyboard, USART for serial communication, and A/D converter) are connected directly to the CPU data bus in the obvious way.

The RAM used for the display (the "large RAM") consists of 48K 8-bit bytes of 22-pin 4K dynamic RAMs (INTEL 2107A-4 or equivalent). Since the microprocessor is the only source of data for the large RAM, the data inputs are connected directly to the CPU data bus. The output of the RAM can also be directed back to that bus when the CPU wishes to read from that section of its address space. Another possible destination for the RAM data is an 8-bit parallel-load shift register which is used in graph mode for directly displaying the memory data bits as dots on the video image.

When the terminal is in text mode, the data from the large RAM must be used as the address of a particular character in one of the character generators. The output of the character generator is then loaded into another parallel load shift register and serially transmitted to the TV monitor. The data from the large RAM is only part of the address for the character generator; the other part - the identification of the line within the character which is to be displayed - is supplied by the four-bit "character row counter".

Both the standard (ROM) character generator and the writable (RAM) character generator are accessible by the CPU as a part of its address space. For that purpose, the CPU address bus can be multiplexed as the address of the character generator instead of the combination of large RAM output and character row counter. The data output from the character generators can then be multiplexed and enabled onto the CPU data bus.

The address for the large RAM can come either from the CPU address bus, or from a two-level bank of 16-bit counters maintained by the display processor. The initial address for the counters is specified from the CPU by an output instruction before the start of each displayed frame. In graph mode both counters are simply incremented together for each byte used for the display. In text mode the top counter ("row address counter") is used to maintain the address of the first character of the row being displayed, and the bottom counter ("character address counter") is incremented as each character is being displayed. At the end of the displayed line, the character address counter is reloaded from the row address counter so that the next raster line of the character row can be displayed. On the

last raster line of the character row, both counters are incremented so that the next row of characters from the RAM will be output.

3B. The CPU

Figure A2 is the heart of the control processor. The CPU proper consists of the standard triumvirate - 8080A Microprocessor, 8224 Clock Generator, and 8228 System Controller. The address bus is buffered for higher drive capability by 74367 hex buffers (equivalent to 8097, 8T97). The ¬IOR and ¬IOW signals are used to enable one of 8 input ports or one of 16 output ports for I/O instructions.

The 8228 version of the System Controller has a defect (which may be eliminated by the newer 8238 version) which requires that MEMW or IOW cycles be anticipated so that the ready-line computation can be done early enough. Since the CPU cannot always get immediate access to the large RAM, the anticipated access causes a flipflop to be set which withholds the ready signal from the CPU , and it is reset only after the CPU has been granted its memory cycle (end of ¬CPU CE signal).

The System controller is programmed (using the INTA pin as an input) so that it will automatically supply a RST 7 instruction in response to interrupts. There are three possible interrupts: end-of-frame, USART receiver buffer full, and USART tranmitter buffer empty. The end-of-frame interrupt occurs at the start of the vertical retrace, and must be reset by an output signal from the CPU. The USART receiver buffer full interrupt is cleared by reading the received character. The USART transmitter buffer empty interrupt is buffered by a flipflop so that the CPU can acknowledge and turn off the interrupt without initiating another character transmission.

3C. The Display Processor

The top part of Figure A3 contains the addressing logic for the large RAM. The two rows of 74197 four-bit counters are the row- and character-address counters used by the display processor to maintain the address of the character or byte to be displayed. The 7483 four-bit adder is a mechanism to allow the display processor to wraparound to a non-zero address after displaying the last byte (at X'FFFF') in memory, so that continuous scrolling can be performed. The position to which it wraps can be selected by wiring the "A" inputs of the adder; as drawn it wraps to location X'4400' so that the 1024 bytes at the beginning of the large RAM can be used by the CPU for non-display data.

The access time of the dynamic RAMs must be less than the display time for 8 bits of video information, which is 650 ns. The design allows, however, for the cycle time (access time plus minimum chip-enable off time) to be greater than 650 ns by having two-way interleaved access, that is, the low-order address bit is used to select a particular row of ICs rather than a byte within a given row. By doing so it is guaranteed that no two consecutive accesses will be made to the same IC.

The large RAM is organized as three blocks of 16K bytes each, and each block (corresponding to a single PC board in the prototype) consists of four rows of eight ICs. The selection logic for each block (see Figure A8) has the property that all the ICs in a single block can be chip-enabled (and hence refreshed) without being chip-selected. Since the display processor makes sequential accesses to consecutive memory locations for each raster line being displayed, the need to refresh the dynamic RAMs can be satisfied with very little extra hardware. The interleaving of memory addresses is constructed so that adjacent memory locations are in different blocks. To refresh all the ICs, then, it is only necessary to entirely chip-enable the block currently being addressed and the segment which is not involved in the interleaved addresses. The signals which enable the segment refresh are called "Bn REFR" and occur at the end of each character row (every sixth or seventh raster line).

The bottom half of figure A3 contains three subparts concerned with the access of the CPU to memory: (1) the address-space decoder for the various memories to which the CPU has access, (2) the conflict resolution logic which prevents simultaneous access to the large RAM by both the CPU and the display processor, and (3) the timing chain which generates timing signals for CPU accesses to the large RAM. The CPU timing for the large RAM differs from that of the display processor since it may be a write cycle and hence be longer than 650 ns.

Note that the conflict resolution logic can be in either of two modes, selectable by the CPU. In normal or non-"quickmode", the CPU is allowed access to the RAM only during horizontal or vertical retrace, so that the display processor is never prevented from retrieving data needed for the display. In "quickmode" (used, for example, for rapidly clearing all 48K bytes) the CPU is allowed access at the end of any display-processor RAM cycle, which causes the display to get incorrect data for at least one display cycle (650ns). If done repeatedly with the screen not blanked, the interference appears as "snow" covering the displayed image.

3D. Character Generators

Figure A4 describes the path of data from the large RAM to the video monitor. In text mode, the most significant bit of the byte read is used to select a character generator, and the other seven bits are used as the most significant bits of the address to the generator, i.e. the address of the character bit mask. (For this purpose the 4K bytes of static writable character generator memory can be considered to be two 128-character generators). The low-order address to the character generator (the address of the line within the character to be displayed) is provided by the four-bit 74161 counter. This counter, which can be preloaded by the CPU during vertical retrace, either counts to thirteen or sixteen depending on the "16-line mode" signal set by the CPU. It also accounts for the fact that the video display is fully interleaved so that only half the raster lines of each character row are displayed during each frame. For example, if the lines of a character are numbered from 1 to 13, the following sequence of addresses are generated by the character row counter:

First Frame	Second Frame
1	2
3	4
5	6
7	8
9	10
11	12
13	1
2	3
4	5
6	7
8	9
10	11
12	13
1	2

(etc.)

Note that the counter alternately divides by six and seven. The CPU preloads the counter during vertical retrace to specify the starting line; this "defines" the frame for the current display as well as allowing the entire display to be shifted by any number of raster lines compared to a previous frame.

The visible part of the raster line on a standard U.S. 525-line TV monitor is displayed in 52.8 microseconds. Since 81 characters are to be displayed on each raster line, a new character must be provided to the shift registers every 652 nanoseconds. Since the data to be displayed must be extracted from the large RAM (450 ns access time) and perhaps used as an index into a character generator (500 ns access time) some form of pipelining is required to maintain a steady flow of characters for the video display. During the time that a raster line is being generated there are three characters simulataneously being processed: (1) the bit mask for the character currently being displayed is in the 8-bit parallel-load shift register and is being serially shifted into the video stream, (2) the character to be displayed next is latched in the RAM output latch and is propagating through a character generator, and (3) the address of the next character has been latched into the address latches of the RAM and is being fetched.

The serial data from the shift registers is multiplexed into the video data stream by the 74153 multiplexor. In GRAPH mode, the source is always the shift register which was loaded directly from the RAM data latch. In TEXT mode, however, either the shift register attached to the ROM character generator or the shift register attached to the RAM character generator supplies the data, depending on the high-order bit of the original character and the "NO ROM CHARS" signal set by the CPU. Since the decision is made at the second level of the pipeline but used at the third level, the result must be propagated with the pipelined data; it is the function of the 7474 D-latch whose output controls the A input of the bit-stream multiplexer to record the decision made one cycle earlier.

The 4K RAMS used for the writable character generator (EMM SEMI 4200) have static data storage but, unlike the ROM character generator, dynamic addressing logic. For that reason the chip-enable signal used by the large RAM ("VID CE") is used to derive chip-select for the 4K RAMs, and the JK flipflop is used to guarantee that CPU accesses to the 4K RAMs will also obey the requirements for minimum chip-select recovery time.

3E. Timing

Figure A5 shows the logic necessary to generate the timing signals in the VGT, and Figure A6 displays the waveforms for a typical raster line ("H-TIMING") and, at larger scale, for even and odd video frames ("V-TIMING"). The source for almost all timing signals is the 18.4275 MHz crystal which is used with the 8224 CPU Clock Generator to

provide a master clock frequency. The Clock Generator divides the master clock by 9, producing a 2.0475 MHz clock which is used both by the CPU and by the TV Sync Generator (National MM5320). The master clock is also multiplied by 2 (using a delayed input to an exclusive-OR gate) and divided by 3 to produce a 12.285 MHz bit clock. The fact that 648 bit clock pulses fit precisely into the non-blanked part of the raster line as defined by the Sync Generator is the reason that there are 81 character positions per line.

In addition to providing the composite sync used for the TV monitor ("¬SYNC"), the Sync Generator provides HDRIVE, VDRIVE, and BLANK used to derive other needed control signals. VTIME is the time during which the screen is blanked for vertical retrace. PICTURE TIME is the time during which information is being displayed, and excludes VTIME as well as the time for the half-raster line at the top of the screen during every other frame. PHDRIVE is the HDRIVE signal that occurs only when information is being displayed.

In order to begin displaying valid data at the start of a raster line, the first two positions of the pipeline must be prefilled during the last part of the horizontal retrace time. This prefetch of two characters is started when HDRIVE ends during horizontal retrace, and the "PIPE FILL" flipflop is used to record completion of the pipeline filling. No more bit clocks are generated from then until the display is unblanked, at which time normal operation of the pipeline occurs until the next HDRIVE. The last two characters fetched (the 82nd and 83rd) are never displayed.

In GRAPH mode the individual data bits displayed must exactly fill the character space, so the normal BIT CLOCK is used as the shift register clock. However in TEXT mode a more pleasing character proportion is obtained by shifting the data bits somewhat faster than the normal bit rate, which results in narrower characters and a larger space between them. A free-running RC oscillator formed from NAND gates and synchronized with the BIT CLOCK at the left edge of the character space provides the faster clock (approximately 16 MHz).

3F. Serial I/O and Status Logic

Figure A7 contains various CPU peripheral devices. Two four-bit latches are used for the miscellaneous mode-setting signals over which the CPU has control. Note that the CPU cannot read the state of the latches and must maintain an

internal copy in order to be able to make single-bit changes. The 74367 hex buffers are used to put keyboard and interrupt status data onto the CPU data bus when requested. The keyboard is not latched and does not cause interrupts since the CPU can conveniently poll it at the 60 Hz frame rate. In order to prevent race conditions from the keyboard strobe signal, a flipflop is used to record changes in the strobe which occur between poll intervals.

The transmitter and receiver clock to the USART can each be supplied either from the local baud rate generator or from the external world via the RS232 connector; the decision is made by two CPU mode control bits. The baud rate generator is the COM5016 [4] which contains a four-bit latch for each clock and generates its master frequency using an exernal 5.0688 MHz crystal.

5G. The Large RAM

Figure A8 contains the address, chip select, and chip enable drivers for the large RAM. There is one such circuit for each of the three memory boards. (Note that the drivers are INTEL 3210's, which have been discontinued and withdrawn from distributor stock). The ability to chip-select only one row while chip-enabling all four is a requirement of the memory refreshing scheme.

3H. Optional Bells and Whistles

Figure A9 is an example of an analog-to-digital subsystem which might be used for joystick or handheld mouse control input. The AD7570J A/D converter [5] is convenient in that it contains tri-state bus drivers and hence can be put directly onto the CPU data bus. The bus drivers are slow enough, however, to require that a wait cycle be added to the CPU input instruction.

Figure A10 is an example of a parallel interface for external devices. The eight output data bits can be set by the CPU, and changes are indicated by a 1 microsecond clock pulse. The eight input bits can be strobed by an input clock, and are polled by the CPU. We have connected this interface to a Versatec Matrix Printer to get hardcopy in both TEXT and GRAPHICS mode.

THE SOFTWARE

The interface between the software and the hardware consists of two parts: the 48K byte RAM used for the display image, and the I/O ports. Figure A11 contains a list of the I/O ports defined by the hardware shown on the preceeding logic diagrams, and a map of the address space of the CPU.

A complete listing of a version of the VGT software has been included on microfiche as Appendix C. The software is entirely written in assembler language for the 8080. The assembler we use is one written at SLAC and the mnemonics and operand formats differ somewhat from what is normally used for the 8080. Documentation for the SLAC assembler is available from the Computation Research Group [6], but the two-page summary in Appendix B should provide sufficient information for reading the program listing.

The software is not a model of a carefully structured and constructed program. By way of apology I offer the following excuses: (1) major sections of it were written in very short time, (2) it is a direct descendent of the original VGT program which dictated the overall structure but did not anticipate the complexity of the problem, (3) a large part of the graphics software was stolen from a graphics program for the 8008 and therefore looks like silly code for an 8080, and (4) parts of the code were written by two programmers. (After such an introduction he may have preferred anonymity, but Ed Frank's excellent contributions to the software should not go unacknowledged). The software works fairly well, however, and contains a number of very useful and somewhat sophisticated features.

About the I/O definitions: The software in Appendix C is that which was running (at some point) in the prototype VGT. The I/O port numbers for the prototype differ from those shown on the logic diagrams, but the names should be sufficient to indicate the correct correspondence.

At power-up time, or when the console reset button is pressed, the CPU begins execution at location zero. Hardware and software initialization is performed, which sets defaults for all local options. The display memory is cleared and a screen is displayed which contains the standard character set, information about the current software version, and a summary of local command functions.

After initialization, the program is logically divided into a foreground or interrupt-disabled section, and a background or interrupt-enabled section. Interrupts cause a CALL to location X'38' and may orignate either from the USART serial interface or from the display processor indicating that a frame has been completed. After interrupt processing the background program is resumed. If there are no background tasks to do, the CPU executes the HALT (really a WAIT FOR INTERRUPT) instruction. The pin on the CPU which indicates the WAIT state is used to drive a "CPU BUSY" LED on the front panel.

The first responsibility of the CPU at frame-interrupt time is to load the hardware registers which control what is to be displayed for the next frame. In TEXT mode the address of the first character to be displayed and the top raster line of the first character row are loaded from the local RAM and output to the appropriate hardware registers. The values are stored into the local RAM as a result of scrolling commands or characters received from the USART. The top raster line used depends on whether an odd frame or an even frame is to be displayed, and the CPU reads the FRAME status bit to make that determination. In GRAPHICS mode, the even raster lines and odd raster lines of the image have been stored in separate parts of the large RAM, and the CPU uses the FRAME bit to determine which is to be displayed.

Since frame interrupts occur at exactly 1/60 second intervals, the CPU uses that interrupt for a variety of timing purposes. A real-time clock is maintained and can be displayed by keyboard command. The keyboard itself is interrogated only at frame interrupts. The cursor wink rate and the scrolling speeds are controlled by counters that are decremented at the time the frame interrupt occurs.

The COMMAND key is a keyboard shift key which is not part of the 7-bit ASCII code. When it is depressed the other keys are interpreted to be local commands to the VGT software. Most of the COMMAND functions, such as text scrolling or changing the baud rate, are processed immediately, that is, during the frame interrupt at which they were first recognized. A few of the COMMANDS and all keys depressed without the COMMAND shift cause a character to be put in a buffer for processing by the background task.

A USART interrupt can either indicate that a new character has been received or that the transmitter register has been emptied. New characters are put into the "receive

buffer" for later background processing. If the transmitter register is empty, the "transmit buffer" is checked for characters waiting to be sent; if so another character transmission is initiated.

The system for which the VGT was primarily developed is an IBM 370 running the text editor WYLBUR [8]. One of the more obnoxious aspects of almost all IBM-based terminal systems is that full duplex is not supported; you may type to the computer only when it is in a receptive state, or else suffer loss of characters. On terminals which don't have mechanically lockable keyboards the terminal must be watched for the "prompt string" which indicates that you may type. To mitigate this difficulty the VGT has a "type-ahead" mode in which it takes the responsiblity to insure that nothing is sent to the host computer in its non-receptive state. This is made possible since the line protocol implemented by the host system requires that such a state be started by a carriage return, control-D, or break signal from the terminal, and ended by a DC1 sent from the computer.

The background processing task can be in either of two major states corresponding to TEXT mode and GRAPHICS mode. In TEXT mode characters are removed from the receive buffer and inserted into the large RAM at the cursor position. Editing characters (space, backspace, linefeed, carriage return, horizontal tab and vertical tab) are used to position the cursor in the appropriate manner. If the cursor is on the screen and is about to be moved off the screen by such a positioning command, the display is scrolled (by changing the local RAM variable that will be used by the frame interrupt routine) so that the cursor remains on the screen. In the normal case that means that a linefeed which occurs on the last display line causes the text to scroll up one line.

In GRAPHICS mode the VGT is the closest we could come to being a complete Tektronix 4013 simulator [7]. It enters GRAPHICS mode in response to an "erase page" command (ESC-FF) or a "start vector" command (GS), and the large RAM (except for the buffer space) is cleared. The RAM is now considered to be divided into two parts corresponding to the even and odd raster lines of the display.

Within GRAPHICS mode there are two sub-modes: ALPHA mode for receiving text and VECTOR mode (called, unfortunately, GRAPH mode by Tektronix documentation) for drawing vectors. In ALPHA mode the characters to be displayed are used as an index into the character generator,

and the bit mask for the character is transferred into the large RAM at the cursor position. Cursor positioning characters act as in TEXT mode, except that the 4013 two-column wraparound format is used instead of scrolling.

In VECTOR mode the incoming ASCII characters are interpreted as the endpoints of vectors to be drawn on the screen according to the 4013 convention. The points of the vector to be drawn are computed (without multiplications or divisions) with an algorithm which may be represented as follows:

```
Let (PX,PY)  be the current position.
Let (X,Y) be the amount by which PX and PY is to change.

Let INCX and INCY be procedure variables which can have
    the values '+PX', '-PX', '+PY', '-PY' and which
    increment or decrement PX or PY.

INCX:='+PX';
INCY:='+PY';
IF X<0 THEN [ X:=-X; INCX:='-PX'; ]  "make X positive"
IF Y<0 THEN [ Y:=-Y; INCY:='-PY'; ]  "make Y positive"
IF Y>X THEN [  "make X not less than Y"
            X,Y := Y,X;  "exchange deltas"
            INCX,INCY := INCY,INCX;  "exchange directions"
            ]
D:=-X/2;  "initialize pseudo-dividend"
FOR N:=1 to X [
    D:=D+Y;
    IF D>0 THEN [ "nearest Y is one larger"
                D:=D-X;
                INCY;
                ]
    INCX;
    PUTDOT AT (PX,PY);
    ]
```

This procedure generates the dots which are closest to the ideal line connecting (PX,PY) and (PX+X,PY+Y). The actual implementation is somewhat optimized for small vectors and will do large vectors in as many as four separate segments.

EXTENSIONS

The software whose listing is included in Appendix C is to be viewed as a preliminary version. The following list includes modifications we are certain to make in the future, as well as other ideas which are less certain to be implemented.

1. The software is not yet aware of the wraparound hardware for display processor addressing, hence there is an annoying discontinuity in line scrolling at the memory boundary.

2. Tabs do not work for positions greater than 80, or in graph mode.

3. In this version there is no software to load the writable character set, or to enter characters into the memory with the most significant bit set. The standard SI/SO convention for alternate character sets will be used.

4. Software which loads the character set will also be able to load programs into the same memory, which can then be executed locally. There will be interference with the displayed image in TEXT mode, but not in GRAPHICS mode. Here's your chance, SPACEWAR freaks!

5. Macros are to be implemented, so that strings of text can be sent to the host computer with a single keystroke. The macros will be definable either locally or via host computer commmands.

6. Some users have objected to the line scrolling while characters are being received because it moves the text too often for comfortable reading. There will be a page-scroll mode, wherein text is written starting at the top of the screen, and when it gets near the bottom a "page up" scroll command is executed.

7. A reverse horizontal tab is a nice way to avoid long waits with the backspace key depressed.

8. There should be an optional bell which sounds at a user-defined column position during keyboard input, much like on mechanical typewriters.

9. For the system with which we use the VGT, a reverse break from the remote computer causes characters that have been sent to the computer since the last carriage return to be ignored. It would be delightful if the VGT were to automatically resend those characters when the message accompanying the break has been sent.

10. It is awkward that characters typed when the type-ahead feature is being used are not displayed until they are sent. One solution is to reserve a single line on the screen to be used for type-ahead text. A more sophisticated approach would display all typed but unsent lines at the bottom of the display area and separated from the text which is normally interleaved with computer responses. As the responses are received, the type-ahead text lines are moved up to the interleaved text area as they are sent. There should perhaps be two cursors on the screen, one for keyboard input and one for text sent from the computer.

REFERENCES

[1] Baskett, F. and Shustek, L., "The Design of a Low Cost Video Graphics Terminal", Proc. Third Annual Conf. on Computer Graphics, July 14-16 1976, University of Pennsylvania. Also available from the Stanford Linear Accelerator Center as SLAC PUB-1715.

[2] Bell, G. and Strecher, W., "Computer Structures: What Have We Learned From the PDP-11", Proc. Third Symp. Computer Architecture, IEEE/ACM, November 1975.

[3] Electronic Industries Association, "Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange", RS-232-C, August 1969.

[4] SMC Microsystems Corporation, "COM 5016 Dual Baud Rate Generator", Preliminary Specifications, Hauppauge, New York

[5] Analog Devices, "AD7570 CMOS 10-bit Monolithic A/D Converter", Data Sheet, Norwood, Mass., August 1975.

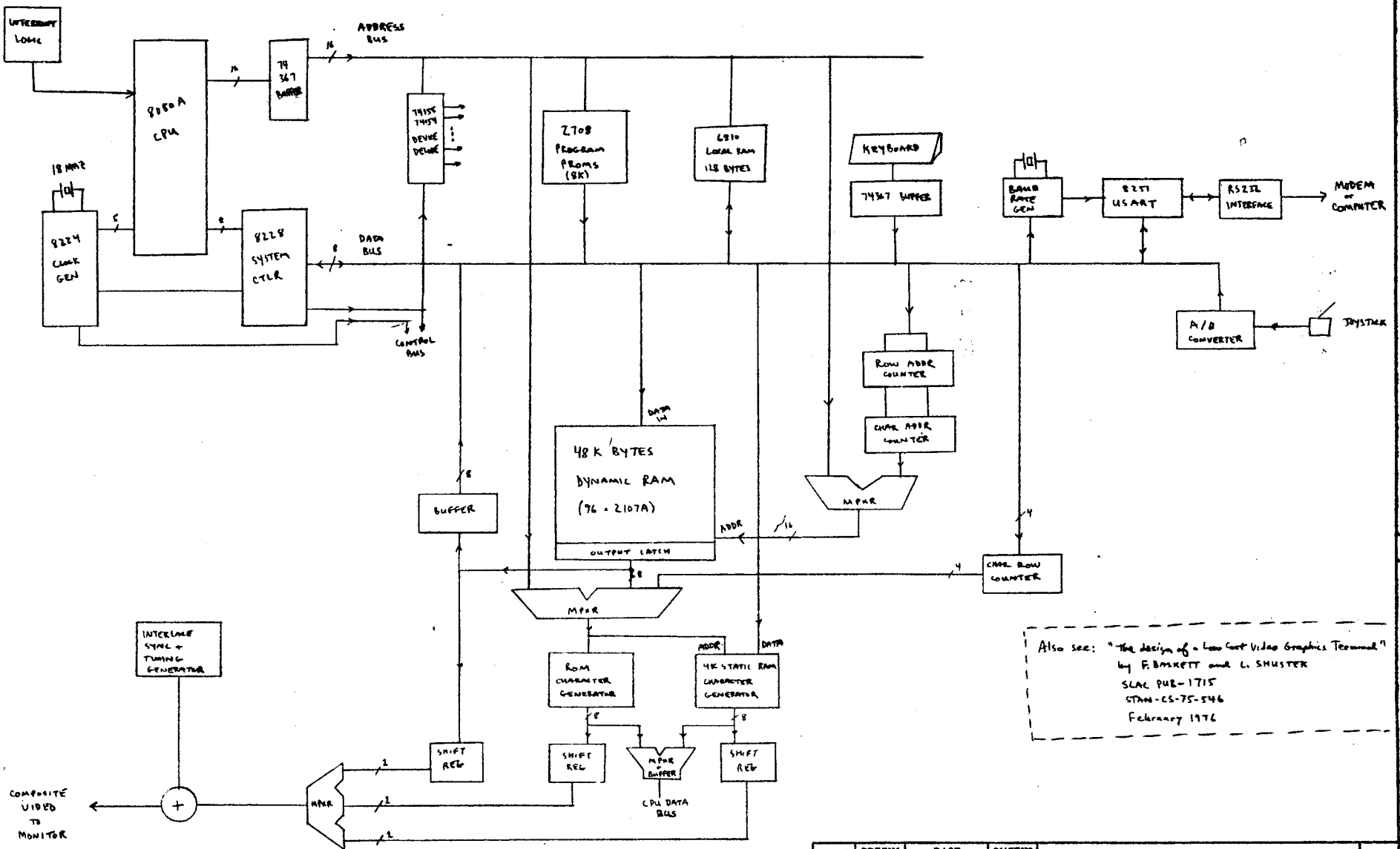
[6] Shustek, L., "Microcomputer Assemblers for the Intel 8080 and the Motorola 6800", Stanford Linear Accelerator Center, Computation Group Technical Memo CGTM-174, May 1976.

[7] Tektronix Corporation, "4013 Computer Display Terminal Users Instruction Manual", Beaverton, Oregon

[8] WYLBUR/370 Reference Manual, Stanford Center for Information Processing, Stanford, Ca. November 1975.

APPENDIX A - LOGIC DIAGRAMS

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE



Also see: "The design of a Low Cost Video Graphics Terminal"
by F. Baskett and L. Shuster
SLAC PUB-1715
STAN-CS-75-546
February 1976

ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: DO NOT SCALE DRAWING NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
STANFORD UNIVERSITY STANFORD CALIFORNIA

VGT VIDEO GRAPHICS TERMINAL

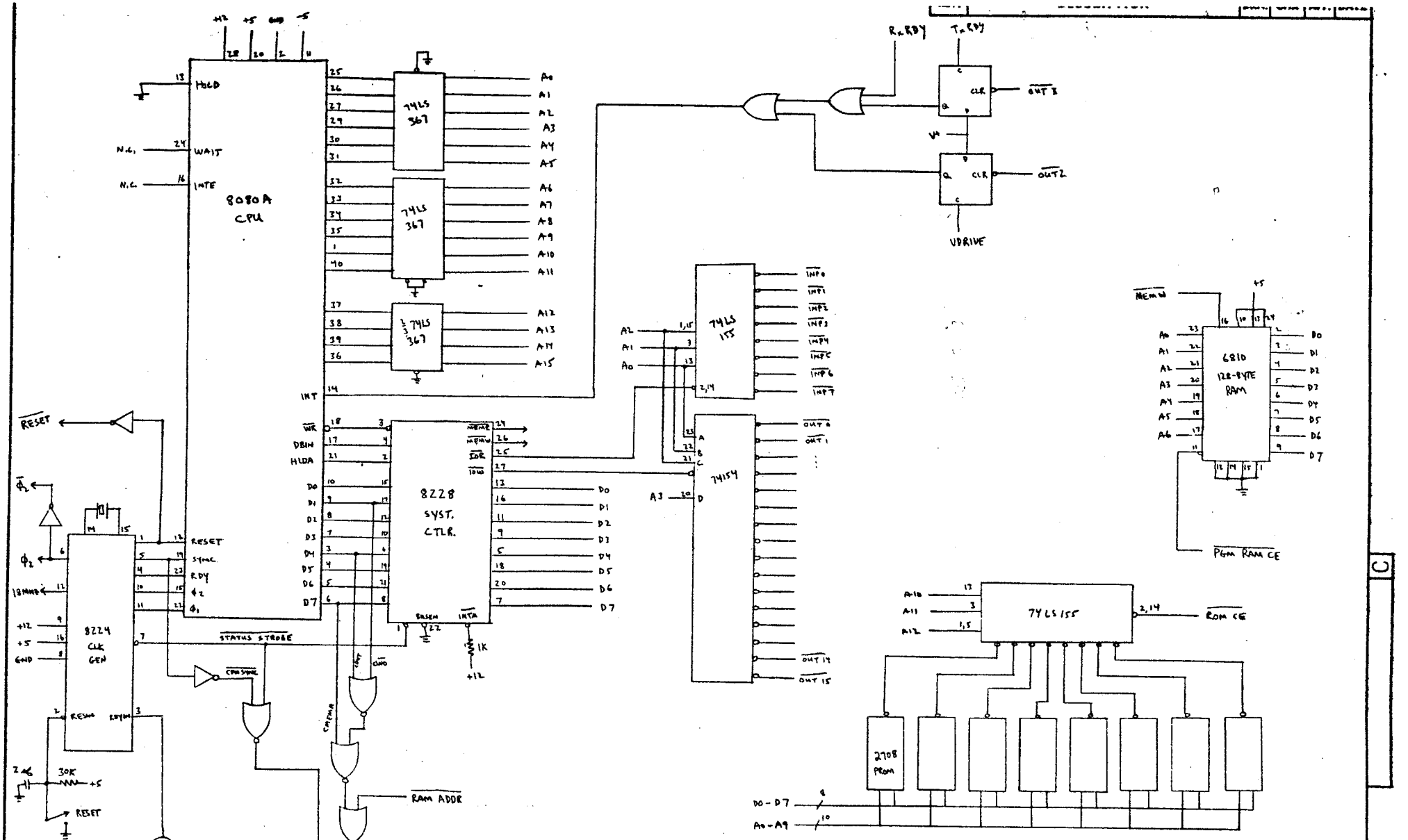
OVERVIEW

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ± ANGLES ± ALL SURFS. ✓

ENGR. SHUSTEK 8/1/75
DFTS.
CHK'R.

APPROVALS

FIGURE A1 REV 1 C



ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: DO NOT SCALE DRAWING NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY STANFORD, CALIFORNIA

VGT

CPU & ROMS

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ± ANGLES ± ALL SURFS. ✓

ENGR. SHUSTEK DATE
DFTS. SHUSTEK 8/1/75, 12/20/76
CHK'D.

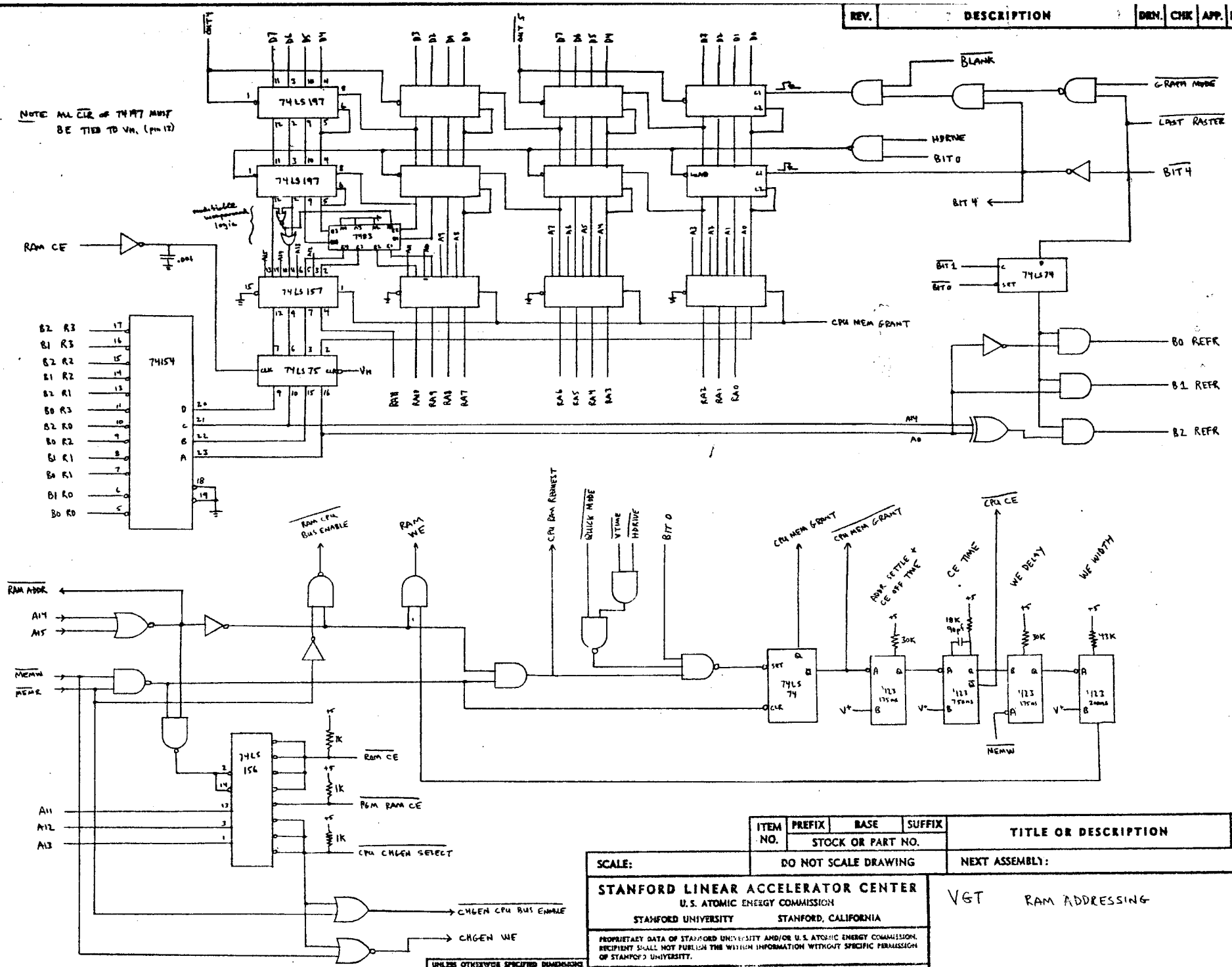
APPROVALS

FIGURE A2

REV 1

C

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE



ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: DO NOT SCALE DRAWING — NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY STANFORD, CALIFORNIA

VGT RAM ADDRESSING

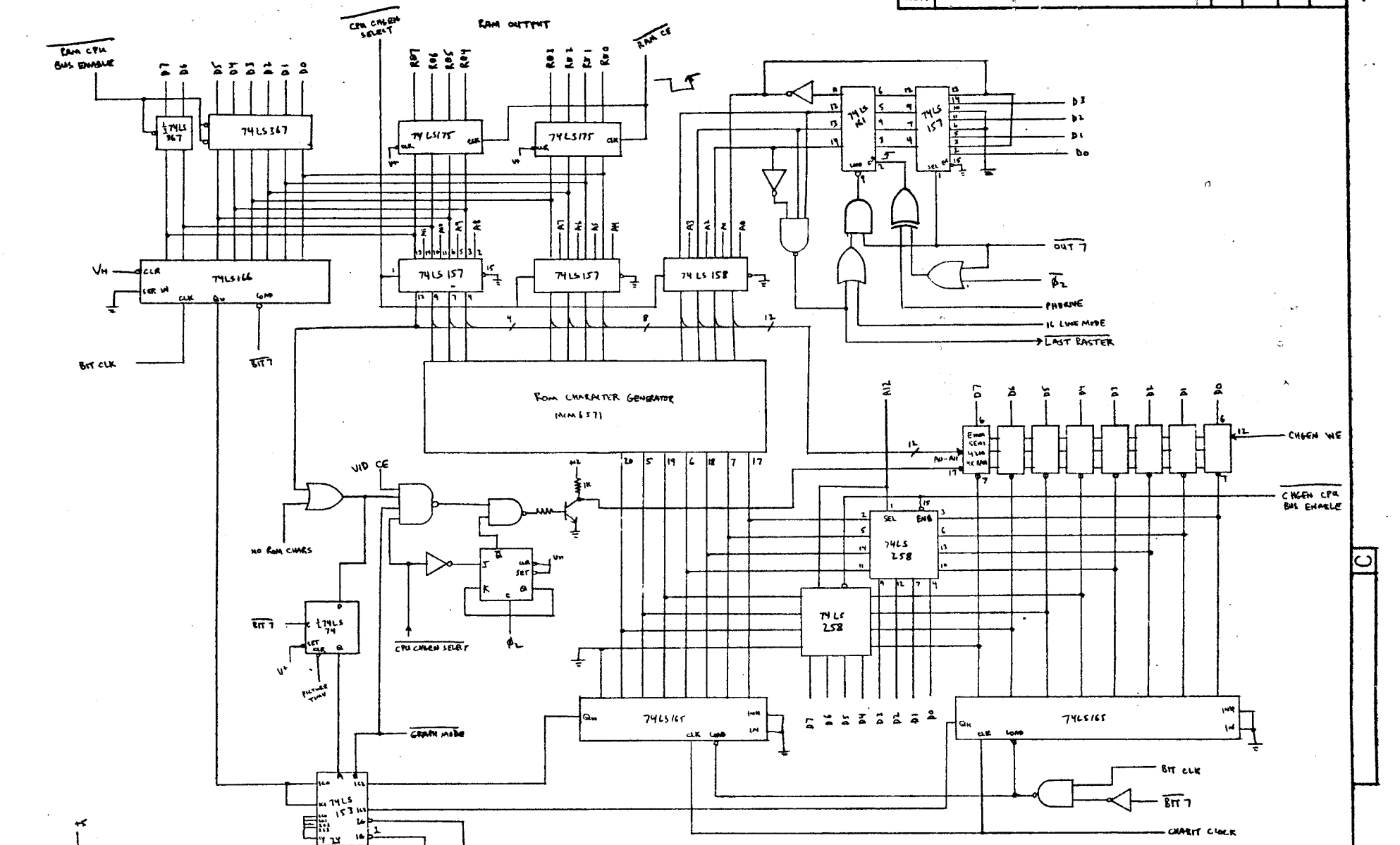
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ±
ANGLES ± ALL SURFS. ✓

ENGR. L. Smutek DATE
DFTS. L. Smutek 8/27, 9/27/74
CHK'R.

APPROVALS

FIGURE A3 C

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE



ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: DO NOT SCALE DRAWING

NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY STANFORD, CALIFORNIA

VGT VIDEO GENERATION

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ± ANGLES ± ALL SURFS. ✓

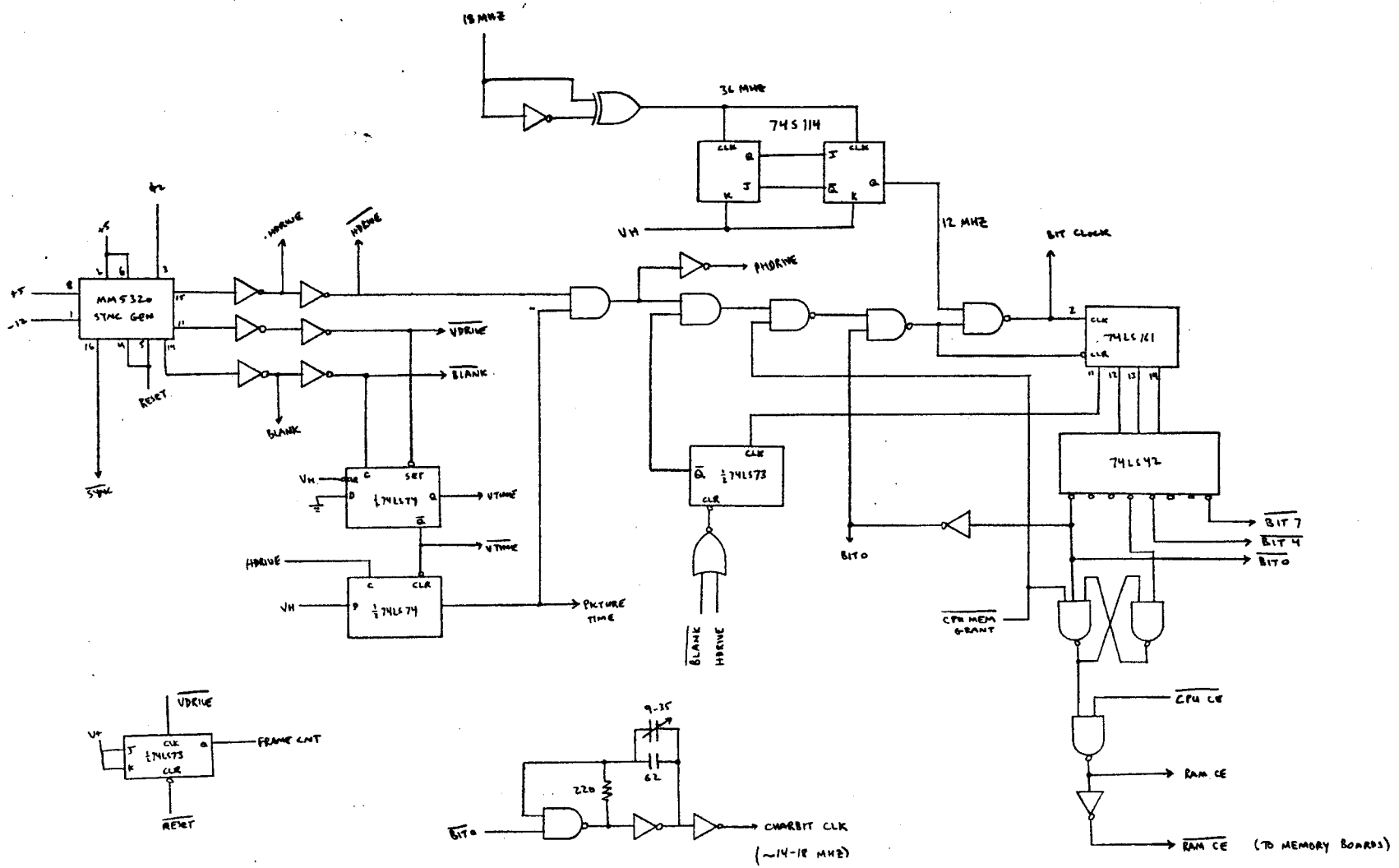
ENGR. L. SHUSTER DATE
DETL. L. SHUSTER 9/15, 3/1/76
CHK'D.

APPROVALS

FIGURE A4

C

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE



ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: DO NOT SCALE DRAWING — NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY STANFORD, CALIFORNIA

VGT

TIMING LOGIC

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ± ALL SURFS. ✓
ANGLES ±

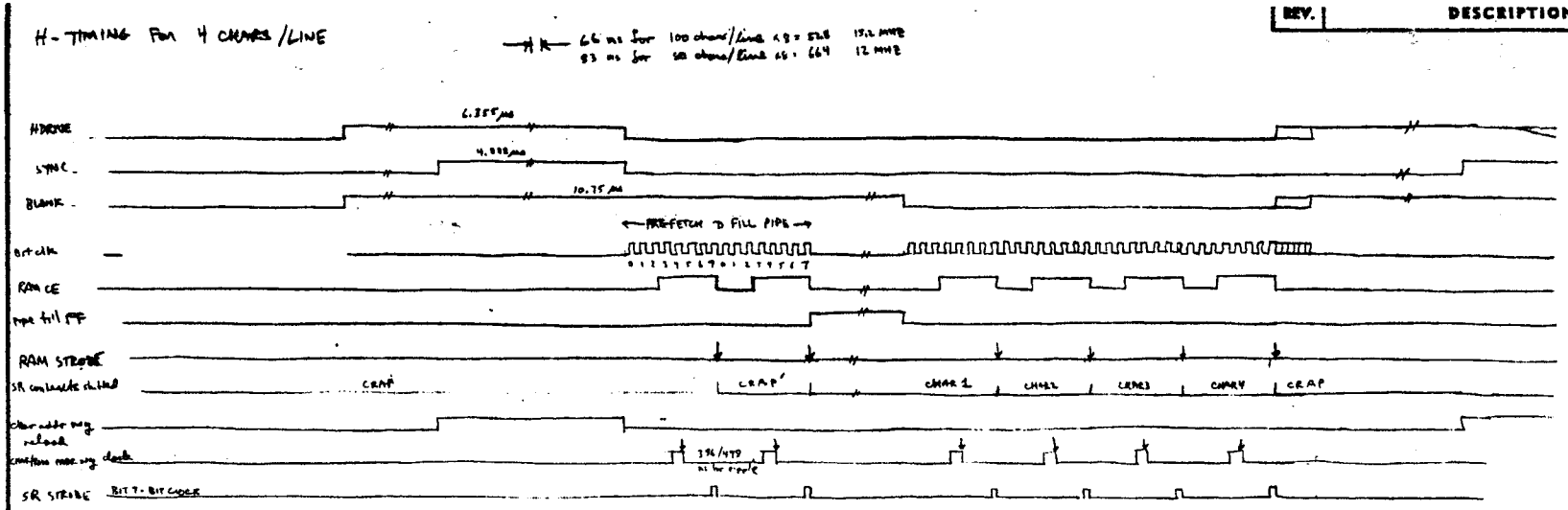
ENGR. L. SHUSTER DATE
DFTS. L. SHUSTER 8/9/75 8/1/76
CHK'D.

APPROVALS

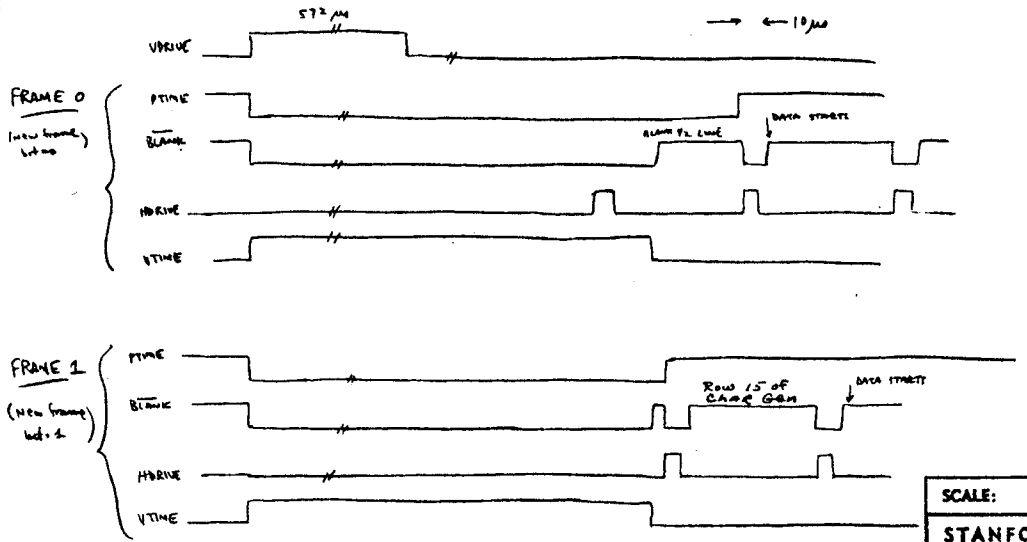
FIGURE A5 C

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE

H-TIMING FOR 4 CHARS/LINE



V-TIMING



ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: DO NOT SCALE DRAWING NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY STANFORD, CALIFORNIA

VGT
TIMING

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGE .005-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ±
ANGLES ± ALL SURFS. ✓

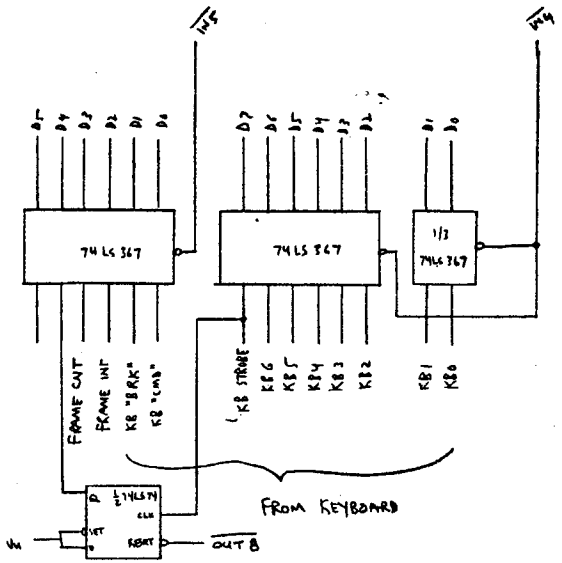
DRWN. SHUSTEK 8/1/75 DATE
DFTS.
CHK'D.

APPROVALS

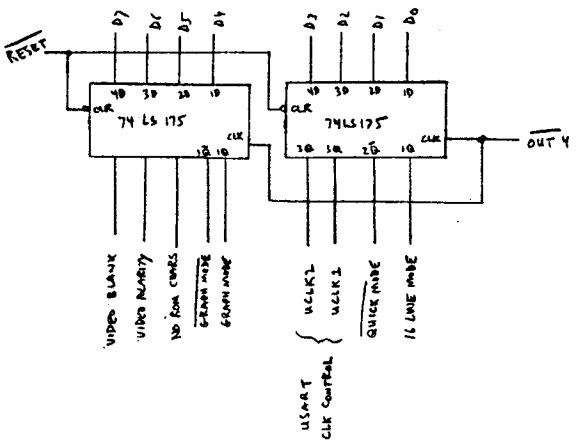
FIGURE A6 C

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE

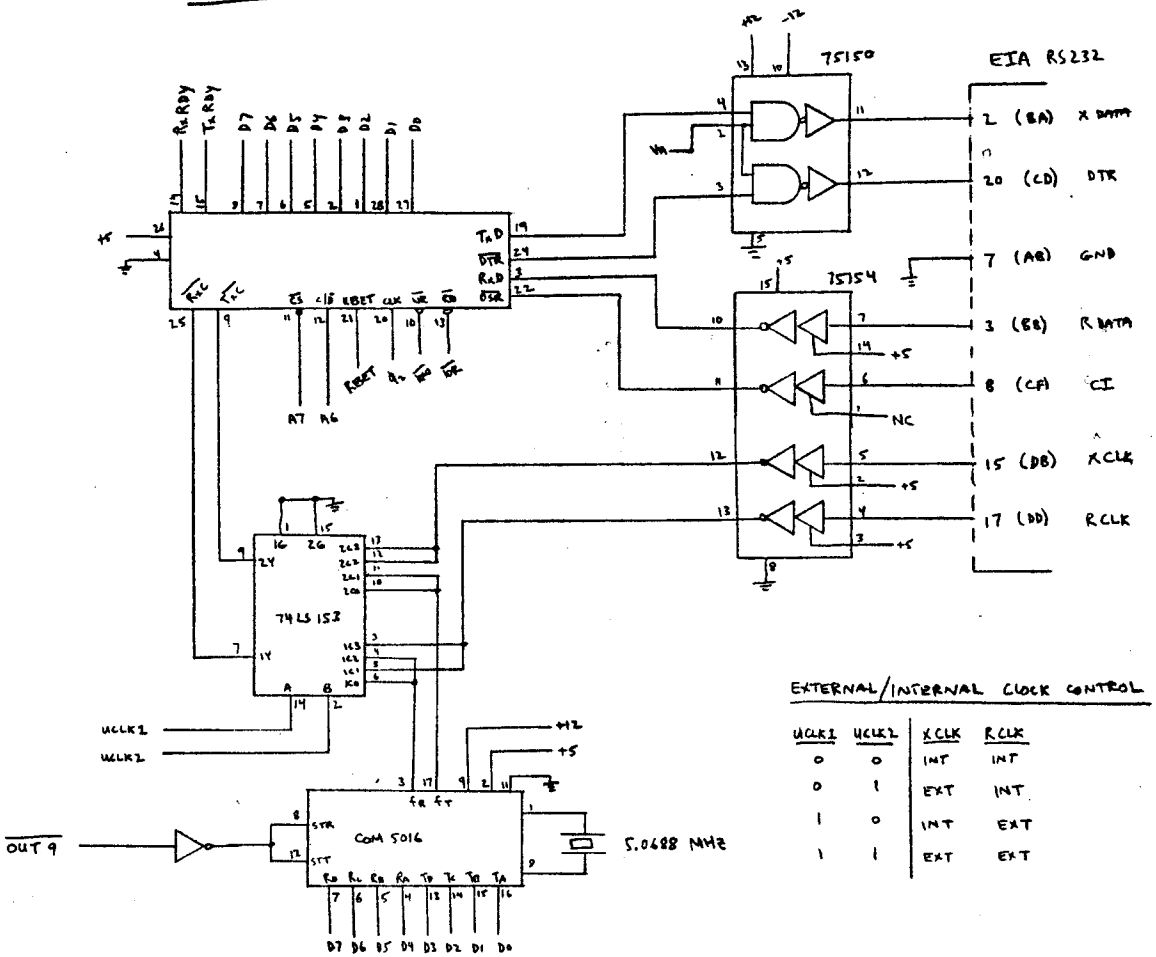
KEYBOARD AND STATUS INPUTS



CONTROL OUTPUTS



USART



EXTERNAL/INTERNAL CLOCK CONTROL

UCLK1	UCLK2	XCLK	RCLK
0	0	INT	INT
0	1	EXT	INT
1	0	INT	EXT
1	1	EXT	EXT

ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: DO NOT SCALE DRAWING — NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY — STANFORD, CALIFORNIA

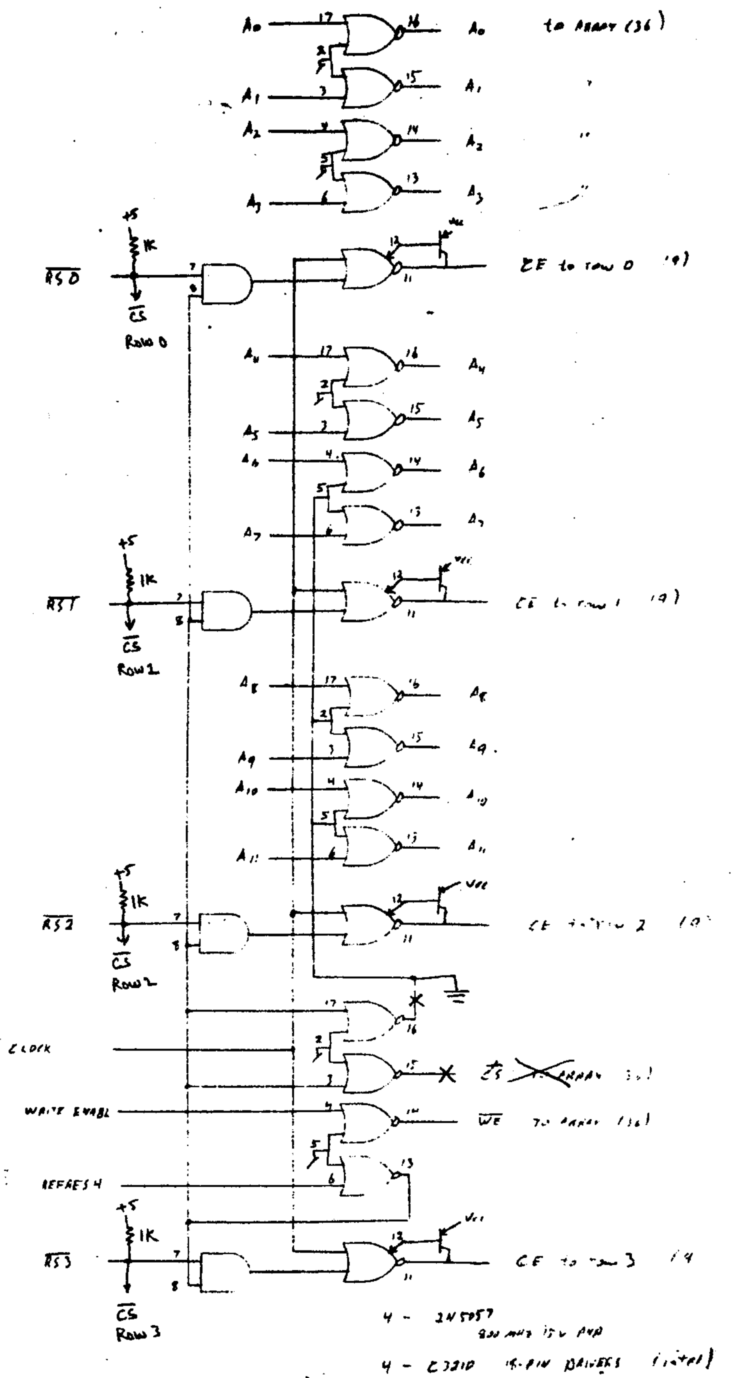
VGT — KEYBOARD USART CONTROL BITS

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± — INT. COR. .015 R. MAX.
DEC. ± — ANGLES ± — ALL SURFS. ✓

ENGR. L. SHUSTEK — DATE
DFTS. L. SHUSTEK 8/9/75, 3/29/76
CHK'D.

APPROVALS

FIGURE A7 — C



1	A0 address	A	data in 0
2	$\overline{CE}$ clock	B	out 0
3	A1	C	in 1
4	A2	D	out 1
5	A3	E	in 2
6	$\overline{Row\ 0}$ select	F	out 2
7	A4	H	in 3
8	A5	J	out 3
9	A6	K	in 4
10	A7	L	out 4
11	$\overline{Row\ 1}$ select	M	in 5
12	A8	N	out 5
13	A9	P	in 6
14	A10	R	out 6
15	A11	S	in 7
16	$\overline{Row\ 2}$ select	T	out 7
17	write enable	U	in 8 (parity)
18	Refresh	V	out 8
19	$\overline{Row\ 3}$ select	W	VBB (-5v)
20	Gnd	X	Ground
21	+5	Y	+5v
22	+12	Z	+12v

Drivers for 36 (4k x 1) 2107A (Intel) Dynamic RAMS (4K x 1)

Forming 16 kilobytes + parity

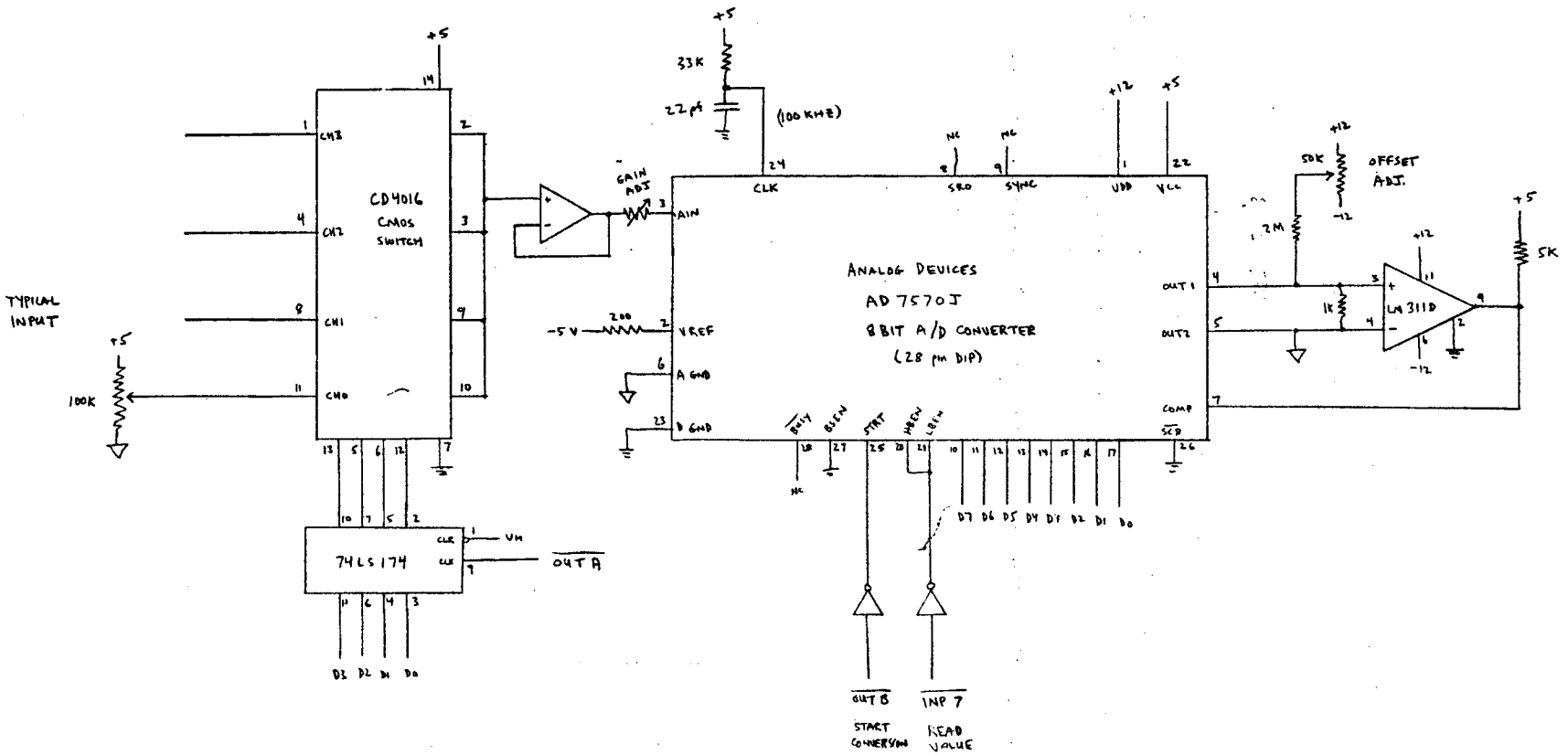
RAM ARRAY DRIVERS

David B Gustavson

DEC '74

FIGURE A8

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE



ANALOG GROUND

DIGITAL GROUND

ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
		STOCK OR PART NO.			

SCALE: DO NOT SCALE DRAWING NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U. S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY STANFORD, CALIFORNIA

VGT

ANALOG INPUTS

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± INT. COR. .015 & MAX.
DEC. ±
ANGLES ± ALL SURFS. ✓

ENGR. L. SORENSEN 3/15/76 DATE
DFTS.
CHK'D.

APPROVALS

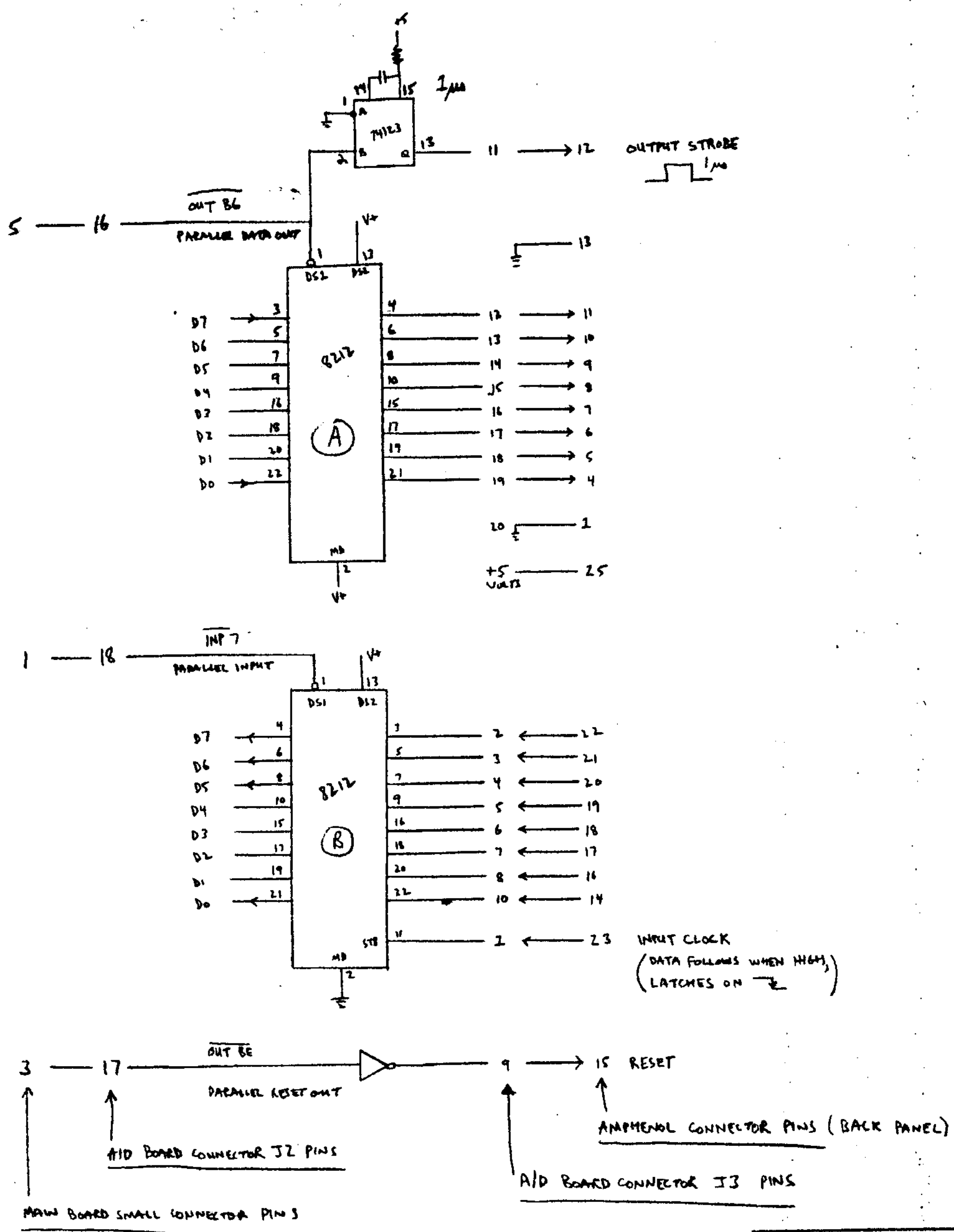
FIGURE A9

C

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE

CABLE TO VERATEC MATRIX 200 PRINTER:

1. PARALLEL DATA FROM AMP. PINS 4-11
2. CLEAR FROM AMP. PIN 15
3. PRINTER READY to AMP PIN 14 (LSB)
4. INPUT CLOCK (AMP 23) CONNECTED to +5 (AMP 25) IN CABLE CONNECTOR



ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: | DO NOT SCALE DRAWING | NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
STANFORD UNIVERSITY STANFORD CALIFORNIA

VGT DIGITAL I/O
(on A/D Board)

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .005-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ±
ANGLES ± ALL SURFS. √

ENGR. SHUSTEK 5/21/76 DATE
DFTR.
CHK'D.
APPROVALS

FIGURE A10

C

REV.	DESCRIPTION	DRN.	CHK	APP.	DATE

OUTPUT PORTS

X'80' MODE BITS
- A0 = 1 16 raster/char mode
- A1 = 1 QUICK MODE
- A2, A3 } USART CLOCK CONTROL (see TABLE)
- A4 = 1 GRAPH MODE
- A5 = 1 NO ROM CHARS MODE
- A6 = 1 REVERSE VIDEO POLARITY
- A7 — UNUSED —

X'01' TRANSMIT USART CHAR IN A

X'41' SET USART CONTROL REG FROM A

X'82' RESET FRAME INTERRUPT

X'83' RESET USART INTERRUPT

X'84' SET HIGH-ORDER DISPLAY ADDRESS FROM A

X'85' SET LOW-ORDER DISPLAY ADDRESS FROM A

X'86' RING BELL

X'87' SET 1ST CHAR ROW RASTER LINE BY A3-A0

X'88' RESET KEYBOARD STROBE LATCH

X'89' SET INTERNAL USART CLOCK SPEEDS A7-A4 (RCVR)
A3-A0 (XMTR)

X'8A' SELECT ANALOG INPUTS FROM A3-A0

X'8B' START A/D CONVERSION (20 μs)

INPUT PORTS

X'01' INPUT USART CHAR

X'41' INPUT USART STATUS BITS

X'84' KEYBOARD A6-A0 ASCII DATA BITS
A7 "KEY PRESSED" STROBE

X'85' STATUS BITS
- A0 KB "CMD" KEY PRESSED
- A1 KB "BRK" KEY PRESSED
- A2 FRAME INTERRUPT
- A3 FRAME COUNT BIT
- A4 KB STROBE LATCH

X'86'

X'87' INPUT A/D VALUE

NOTES:
1. A7-A0 INDICATE A-REGISTER BITS, NOT ADDRESS LINES
2. HIGH ORDER PORT ADDRESS BIT CONTROLS USART SELECTION, IF PA = PORT ADDRESS, THEN PA7=0 AND (PA3-PA0)=1111 SELECTS USART PA6=0 for DATA, 1 for CONTROL/STATUS

MEMORY MAP (IN HEX)

			SIZE (DECIMAL)
0000	to 1FFF	PROM for program	8192
2000	to 207F	LOCAL RAM for CPU	128
2080	to 27FF	— UNUSED —	1920
2800	to 2FFF	ROM CHARACTER GENERATOR	2048
3000	to 3FFF	RAM CHARACTER GENERATOR	4096
4000	to FFFF	Large RAM for display	49152
			65536

ITEM NO.	PREFIX	BASE	SUFFIX	TITLE OR DESCRIPTION	QTY.
	STOCK OR PART NO.				

SCALE: | DO NOT SCALE DRAWING | NEXT ASSEMBLY:

STANFORD LINEAR ACCELERATOR CENTER
U.S. ATOMIC ENERGY COMMISSION
STANFORD UNIVERSITY STANFORD, CALIFORNIA

VGT

I/O PORTS AND MEMORY MAP

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES.
TOLERANCES: BREAK EDGES .015-.015
FRACT. ± INT. COR. .015 R. MAX.
DEC. ± ANGLES ± ALL SURFS. ✓

	NAME	DATE
ENGR.	L. SHUSTER	
DFTS.	L. SHUSTER	5/21/76
CHK'D.		

APPROVALS

FIGURE A11 | C

APPENDIX B - SUMMARY OF ASSEMBLER FORMAT

Notation: Upper case letters must be written as shown.
Lower case letters indicate variable fields to be substituted; see the KEY at the end.
{x,y,z} means x or y or z
[x] means x is optional

Opcode	Operands	Description
LOD	rrr,rrr	Register-to-register loads
LOD	SP,HL	
LODI	r,data	Load register from immediate data
LODI	r,addr,{<,>}	
LODI	rp¹,addr	
PUSH	rp²	Push onto stack pointed to by SP
POP	rp²	Pop from stack pointed to by SP
LD	A,gaddr	Load A from memory
ST	A,gaddr	Store A into memory
LD	HL,addr	Load HL from memory
ST	HL,addr	Store HL into memory
XCH	HL,DE	Exchange HL with DE
XCH	HL,(SP)	Exchange HL with the top-of-stack
opr	r	register-to-A arithmetic
ADD	HL,rp¹	16-bit addition to HL
oprI	data	Immediate data to A arithmetic
oprI	addr,{<,>}	
ROT	rop[,n]	Accumlator rotate/shift
INC	{r,rp¹}[,n]	Increment register or register pair
DEC	{r,rp¹}[,n]	Decrement register or register pair
JMP	[cc,]addr	Jump
JMP	(HL)	Jump indirect
CALL	[cc,]addr	Call subroutine
RET	[cc]	Return from subroutine
RST	m	Restart (CALL 8*m, 0≤m≤7)
IN	dev	Input to A from I/O device
OUT	dev	Output from A to I/O device
NOP	[n]	Null operation n times
HLT		Halt (actually Wait-for-Interrupt)

STC	Set carry bit on
CMC	Complement carry bit
CMA	Complement A
DAA	Decimal adjust A
EI	Enable Interrupts
DI	Disable Interrupts

KEY

rrr — one or more registers {A,B,C,D,E,H,L,M}
LOD B,C means "load B from C"
LOD BC,HL is equivalent to LOD B,H then LOD C,L
M represents the memory location pointed to by HL.

r — one register {A,B,C,D,E,H,L,M}

rp[1] — a register pair {BC,DE,HL,SP}

rp[2] — one or more register pairs {BC,DE,HL,AF,PSW} separated by commas. (AF = PSW = A reg and flags)
PUSH BC,DE is equivalent to PUSH BC then PUSH DE
POP BC,DE is equivalent to POP BC then POP DE

data — is an 8-bit constant or expression

addr — is a 16-bit constant or expression

{<,>} — is the character < if the high-order 8 bits of the address are to be used, or the character > if the low-order 8 bits of the address are to be used. Mnemonic: view them as left- or right-pointing arrows.

gaddr — a "generalized" address {addr,(BC),(DE),(HL)}

opr — an arithmetic operation
{ADD,ADC,SUB,SBB,ANA,XRA,ORA,CMP}

rop — a rotate operation {R,L} for 8-bit rotates
{RC,LC} for 9-bit rotates

cc — a condition code {C,NC,Z,NZ,P,M,NS,S,PE,PO,U}
(S=M=sign bit on, NS=P=sign bit off, U=unconditionally)

n — a positive integer exression or constant indicating how many such instructions should be generated.

dev — is an 8-bit I/O device number or expression.

APPENDIX C - SOFTWARE LISTING