21959

THE APPLE II® CIRCUIT DESCRIPTION

BY
WINSTON D. GAYLER

The Apple II® Circuit Description

by Winston Gayler

Howard W. Sams & Co., Inc.
4300 WEST 62ND ST. INDIANAPOLIS, INDIANA 46268 USA

FIRST EDITION
FIRST PRINTING—1983

International Standard Book Number: 0-672-21959-X
Library of Congress Catalog Card Number: 82-61966

Edited by: *Welborn Associates*
Illustrated by: *D. B. Clemons*

Printed in the United States of America.

Preface

This book is a detailed circuit description of the Apple II® computer. Specifically, it covers the main logic board including all revisions from the earliest through the latest (Rev. 0 through RFI Rev. D). Also covered are the current two-piece keyboard and the older single-piece keyboard.

The intended audience includes engineers, technicians, students, and hobbyists. An attempt has been made to appeal to most skill levels and backgrounds. This is done by dividing each chapter into two sections: Overview and Detailed Circuit Analysis. You may choose to read one, the other, or both.

The book consists of eight chapters. Chapter 1 discusses nomenclature and symbols used throughout the text and figures. This chapter also contains a glossary of terms. Chapter 2 presents a description of the Apple II computer at the block diagram level.

The detailed descriptions start in Chapter 3. Chapters 3 and 4 cover the system clocks and part of the video circuitry. Chapter 5 explains the memory system and Chapter 6 examines the 6502 microprocessor and the system bus. Chapters 5 and 6 contain overview sections dedicated to the 4116 RAM and the 6502 microprocessor. As a result, the reader need not have prior knowledge of these devices.

The keyboard and other on-board I/O are discussed in Chapter 7. The video display (graphics and text) is the subject of Chapter 8. Much of the Apple's circuitry serves to generate a video signal. Due to the importance of this topic, Appendix A is included to provide an introduction to video techniques.

Appendix B is a gathering of all known circuit revisions of the Apple II mother board and keyboard. While the body of the book describes the latest Apple II revision, Appendix B describes differences between this latest Apple and all earlier revisions. Appendix C contains a set of mother board and keyboard schematics. These schematics cover all revisions. The book concludes with a list of references.

WINSTON D. GAYLER

EDITOR'S NOTE: In order to present the illustrations as large as possible many have been placed on the foldout pages at the back of the book. These illustrations are indicated with an asterisk(*) immediately following the figure number in the text, e.g., Fig. 3-4* indicates that Fig. 3-4 is located at the back of the book.

Acknowledgments

I wish to thank fellow Apple enthusiast Dr. James Alinsky for his many suggestions and general inspiration during the research and writing of this book. I am very appreciative for his review, from the reader's viewpoint, of the complete manuscript.

Thanks also go to the numerous employees of Apple Computer, Inc., who assisted by providing documentation and answering questions. I appreciate the assistance of Apple's engineering department in coordinating much of this activity.

Finally, I wish to acknowledge Stephen Wozniak who designed the Apple II, making it all possible.

Contents

CHAPTER 1

CHAPTER 2

CHAPTER 3

CHAPTER 4

CHAPTER 5

CHAPTER 6

CHAPTER 7

CHAPTER 8

APPENDIX A

APPENDIX B

APPENDIX C

APPENDIX D

CHAPTER 1

Introduction

Have you ever wanted to know the detailed circuit operation of your Apple II®† computer? Perhaps you were designing a peripheral or making a modification. Maybe you were repairing an Apple. You may have just been curious about how it works.

This book started as an exercise in understanding the Apple II hardware. The initial goal was to evaluate or design circuit modifications. It soon became apparent that the information obtained would be useful to others. Thus, the plans for a book developed.

The result is a detailed circuit description and analysis of the main circuit board and keyboard of the Apple II. In this introductory chapter, we present the organization of the book and explain some of the terms and symbols used.

THE AUDIENCE

This book is intended for engineers, technicians, students, and serious hobbyists. The engineer and hobbyist can use the descriptions and timing diagrams as a preparatory step to designing peripherals or modifications. The service technician can use the timing diagrams and schematics as aids to troubleshooting. The waveform drawings are particularly handy for troubleshooting with an oscilloscope. The student can use the Apple II for examples of practical circuit design. In many cases the reasons behind the design are presented here. All readers can use the descriptions to better understand how the Apple II works.

CHAPTER ORGANIZATION

Chapter 2 is a block diagram description of the Apple II mother board. There we introduce the names such as "address multiplexer" and "video address generator" given to various functional circuit blocks. Chapter 2 contains the discussion of the power supply, a simplified circuit description.

Chapters 3 through 8 comprise the body of the book. Each of these chapters takes a functional part of the circuit and examines it in detail. These chapters are

†Apple and Apple II are registered trademarks of Apple Computer, Inc.

each divided into two sections: Overview and Detailed Circuit Analysis. The Overview presents the circuit concepts and often contains block diagrams and simplified timing diagrams. If the material is new to you, you may want to read only the Overviews and save the Detailed Circuit Analyses until you need specific details. On the other hand, you may already be familiar with Apple II hardware. In that case, you may want to jump directly into the Detailed Circuit Analyses and skip the Overviews.

Chapter 3 presents the master oscillator, clock generator, and horizontal portion of the video address generator. Clocks are always important in a digital circuit, and they are especially important in the Apple II due to their interplay with the video circuitry.

Chapter 4 completes the video address generator by presenting its vertical portion. The chapter also covers video sync, blanking, and color burst.

The random access memory in the Apple II is shared by the microprocessor and the video generator. Chapter 5 covers this shared access scheme. The chapter also contains an introduction to the type 4116 dynamic RAM (random access memory).

Chapter 6 starts with an introduction to 6502 microprocessor hardware. The chapter then describes all the 6502 cycle types that are used in the Apple. Included are read cycles, write cycles, RAM cycles, ROM cycles, I/O cycles, keyboard cycles, interrupts, and DMA (direct memory access).

Chapter 7 describes the Apple II on-board I/O devices, such as cassette I/O, game I/O, and speaker. This chapter also contains the circuit description for the current *two-piece* keyboard.

The video generator is described in Chapter 8. There you will learn how text, LORES, and HIRES are generated by the hardware under software control.

Appendix A is an introduction to video signal techniques. If you are not familiar with video signals, such as sync, blanking, and color burst, then you may want to read Appendix A. Doing so may increase your appreciation of Chapters 3, 4, and 8.

Appendix B covers the topic of Apple II circuit revisions. The most recent circuit available (RFI mother board, Rev. D) is covered in the body of the book. Circuit variations dating back to the earliest Apple II (Rev. 0) are then covered in Appendix B. The appendix also contains modified waveform drawings for signals that differ in earlier revisions. The circuit description for the old *single-piece* keyboard is contained in Appendix B.

Appendix C contains schematic diagrams for all revisions of the Apple II.

A list of references follows the appendices—it is arranged by chapter.

TRADEMARKS, PATENTS, AND COPYRIGHTS

WHAT YOU SHOULD KNOW

As a reader of this book, you should be familiar with TTL (transistor-transistor logic), such as gates, flip-flops, shift registers, and multiplexers. While reading, you

CHAPTER 1

Introduction

Have you ever wanted to know the detailed circuit operation of your Apple II®† computer? Perhaps you were designing a peripheral or making a modification. Maybe you were repairing an Apple. You may have just been curious about how it works.

This book started as an exercise in understanding the Apple II hardware. The initial goal was to evaluate or design circuit modifications. It soon became apparent that the information obtained would be useful to others. Thus, the plans for a book developed.

The result is a detailed circuit description and analysis of the main circuit board and keyboard of the Apple II. In this introductory chapter, we present the organization of the book and explain some of the terms and symbols used.

THE AUDIENCE

This book is intended for engineers, technicians, students, and serious hobbyists. The engineer and hobbyist can use the descriptions and timing diagrams as a preparatory step to designing peripherals or modifications. The service technician can use the timing diagrams and schematics as aids to troubleshooting. The waveform drawings are particularly handy for troubleshooting with an oscilloscope. The student can use the Apple II for examples of practical circuit design. In many cases the reasons behind the design are presented here. All readers can use the descriptions to better understand how the Apple II works.

CHAPTER ORGANIZATION

Chapter 2 is a block diagram description of the Apple II mother board. There we introduce the names such as "address multiplexer" and "video address generator" given to various functional circuit blocks. Chapter 2 contains the discussion of the power supply, a simplified circuit description.

Chapters 3 through 8 comprise the body of the book. Each of these chapters takes a functional part of the circuit and examines it in detail. These chapters are

†Apple and Apple II are registered trademarks of Apple Computer, Inc.

each divided into two sections: Overview and Detailed Circuit Analysis. The Overview presents the circuit concepts and often contains block diagrams and simplified timing diagrams. If the material is new to you, you may want to read only the Overviews and save the Detailed Circuit Analyses until you need specific details. On the other hand, you may already be familiar with Apple II hardware. In that case, you may want to jump directly into the Detailed Circuit Analyses and skip the Overviews.

Chapter 3 presents the master oscillator, clock generator, and horizontal portion of the video address generator. Clocks are always important in a digital circuit, and they are especially important in the Apple II due to their interplay with the video circuitry.

Chapter 4 completes the video address generator by presenting its vertical portion. The chapter also covers video sync, blanking, and color burst.

The random access memory in the Apple II is shared by the microprocessor and the video generator. Chapter 5 covers this shared access scheme. The chapter also contains an introduction to the type 4116 dynamic RAM (random access memory).

Chapter 6 starts with an introduction to 6502 microprocessor hardware. The chapter then describes all the 6502 cycle types that are used in the Apple. Included are read cycles, write cycles, RAM cycles, ROM cycles, I/O cycles, keyboard cycles, interrupts, and DMA (direct memory access).

Chapter 7 describes the Apple II on-board I/O devices, such as cassette I/O, game I/O, and speaker. This chapter also contains the circuit description for the current *two-piece* keyboard.

The video generator is described in Chapter 8. There you will learn how text, LORES, and HIRES are generated by the hardware under software control.

Appendix A is an introduction to video signal techniques. If you are not familiar with video signals, such as sync, blanking, and color burst, then you may want to read Appendix A. Doing so may increase your appreciation of Chapters 3, 4, and 8.

Appendix B covers the topic of Apple II circuit revisions. The most recent circuit available (RFI mother board, Rev. D) is covered in the body of the book. Circuit variations dating back to the earliest Apple II (Rev. 0) are then covered in Appendix B. The appendix also contains modified waveform drawings for signals that differ in earlier revisions. The circuit description for the old *single-piece* keyboard is contained in Appendix B.

Appendix C contains schematic diagrams for all revisions of the Apple II.

A list of references follows the appendices—it is arranged by chapter.

TRADEMARKS, PATENTS, AND COPYRIGHTS

WHAT YOU SHOULD KNOW

As a reader of this book, you should be familiar with TTL (transistor-transistor logic), such as gates, flip-flops, shift registers, and multiplexers. While reading, you

may want to have a copy of a TTL data book, such as Reference 1.2. You need not be familiar with the 4116 RAM or the 6502 microprocessor. Special sections in Chapters 5 and 6 will cover these devices. However, you should have a basic knowledge of microprocessor and microcomputer architecture. And of course you should be familiar with the binary and hexadecimal number systems.

Concerning your Apple II background, you should be familiar with the *Apple II Reference Manual* (Reference 1.1).

REVISIONS

There have been several circuit revisions to the Apple II since its introduction in 1977. The revisions are discussed in detail in Appendix B. Here in Chapter 1 we simply summarize the changes and establish a revision nomenclature for use throughout the book.

There are two categories of the Apple II mother board: *Non-RFI* (the early mother boards) and *RFI* (recent mother boards that have been designed to reduce radio-frequency interference).

Non-RFI mother boards have the part number 820-0001-XX where XX is the revision. The first non-RFI mother board was Revision 0; we will call it simply *Rev. 0*. It had only four HIRES colors. Revision 0 also lacked color killer and power-on reset circuits.

Revision 1 came next and added two more HIRES colors (for the current total of six). Revision 1 also added color killer and power-on reset circuits and made other small changes. Revisions 2, 3, and 4 had the same circuit as Rev. 1. In this book, we include them with Rev. 1.

The next significant change occurred at revision 7—the memory jumper blocks were removed and the character generator IC was changed.

The next significant change occurred with the switch to the RFI mother board. This board has the part number 820-0044-XX where XX is the revision. All revisions of this board to date (through Rev. D) have the same functional circuit. We refer to it simply as *RFI*.

The part numbers are found either along the left edge of the mother board or under the 6502 IC. See Appendix B for more details. In summary, the mapping from mother board part numbers to our nomenclature is shown in Table 1-1.

Table 1-1. Mapping From Mother Board Part Numbers

	Part No.	Revision
Non-RFI	820-0001-00	Rev. 0
	820-0001-01 820-0001-02 820-0001-03 820-0001-04	Rev. 1
	820-0001-07 & up	Rev. 7
RFI	820-0044-01 820-0044-C 820-0044-D	RFI

IC AND SIGNAL NOMENCLATURE

The ICs on the mother board are designated by their location on an X-Y grid. The grid coordinates consist of the letters A through K along the left edge of the board

and the numbers 1 through 14 along the front edge of the board (Fig. 1-1*). Within an IC, the individual gates or sections are designated by the pin number of the gate output. For example, a reference in the text to "flip-flop B10-9" refers to a flip-flop in the IC located at coordinates B10. The specific flip-flop is the one whose Q output is on pin 9.

Signals use a similar nomenclature. For example, signal "C11-4" is the signal at pin 4 of IC C11. Some signals have already been given names on the Apple schematics—"LD194" is an example. When a signal name has a bar over it (such as $\overline{CAS}$), it means that the signal is active low. All signal names are printed using uppercase letters.

WAVEFORMS

When we say a digital signal is *low,* we mean it is about 0 volts. When a digital signal is *high,* we mean it is about 4 or 5 volts. The exact voltage level varies with logic family, load, and power supply voltage. For typical 74LSXX logic, a low *output* is less than 0.5 volt and a high *output* is greater than 2.7 volts. The same logic family will accept an *input* of less than 0.8 volt as a low, and greater than 2.0 volts as a high. Input levels between 0.8 and 2.0 volts result in an indeterminate state.

When we say a digital IC's output signal is *high impedance,* we mean that the IC neither drives nor appreciably loads the signal line. The high-impedance state is the third state of *three-state* logic: low, high, and high impedance. We sometimes refer to the high-impedance state as the *off* state. The off state of a three-state IC allows other ICs to turn *on* and drive a common signal line.

Digital waveforms are not drawn to vertical scale. Instead, the three possible states are depicted as shown at the top of Fig. 1-2. The rest of this figure shows the symbols used for transitions between states.

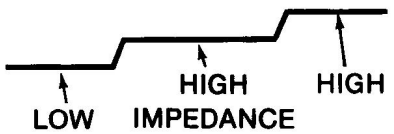
RESEARCH

The research for this book consisted of two major steps: paper analysis and laboratory verification. In the first step, the schematics were analyzed to ascertain the circuit operation and the timing waveforms. In the second step, the waveform drawings were taken into the laboratory and verified using equipment, such as the frequency counter, oscilloscope, logic analyzer, and light pen recorder.

The two research steps were first performed on a Revision 3 non-RFI Apple II. Later, Rev. 0, Rev. 7, and RFI Apples were obtained so that waveforms unique to these revisions could be verified. Operation and waveforms for both the single-piece and two-piece keyboards were also confirmed. Thus, the complete set of Apple II waveforms presented in this book have been laboratory verified.

GLOSSARY OF TERMS

ac—Alternating current.
access time—The time from accessing a memory IC (with an address or clock) until the data becomes stable at the output.
active high—A high level represents a logic 1.
active low—A low level represents a logic 1.
AN—Annunciator.
architecture—Block diagram.
ASCII—American Standard Code for Information Interchange. A common 7- or 8-bit code used by computers and peripherals.

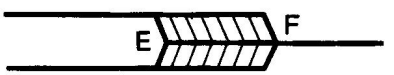


THE THREE LOGIC STATES

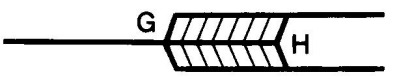
INDETERMINATE STATE (i.e.,GARBAGE)

SIGNAL GOES LOW SOMETIME BETWEEN A AND B

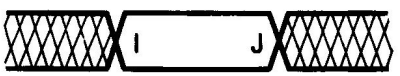
SIGNAL GOES HIGH SOMETIME BETWEEN C AND D



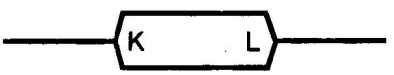
SIGNAL GOES FROM STABLE TO HIGH IMPEDANCE SOMETIME BETWEEN E AND F



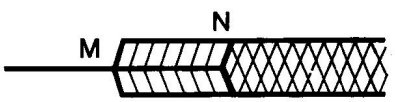
SIGNAL GOES FROM HIGH IMPEDANCE TO STABLE SOMETIME BETWEEN G AND H



SIGNAL IS STABLE BETWEEN I AND J, INDETERMINATE ELSEWHERE



SIGNAL IS STABLE BETWEEN K AND L, HIGH IMPEDANCE ELSEWHERE



SIGNAL GOES FROM HIGH IMPEDANCE TO INDETERMINATE SOMETIME BETWEEN M AND N

SIGNAL GOES FROM HIGH IMPEDANCE TO HIGH SOMETIME BETWEEN O AND P

POSITIVE-GOING GLITCH

Fig. 1-2. Digital waveform symbols.

blanking—The part of a video signal that turns off the scanning electron beam during retrace.
bow tie—A PC board foil pattern in the shape of a bow tie. Meant to be cut to break the circuit.
buffer—A simple logic element used to increase the quantity of gates that a signal can drive,
burst—(See color burst).
bus—A group of signal lines that connect in common to several circuit elements.
byte—Eight bits.
CAS—Column Address Strobe. (Strobes column address into 4116 RAM.)
clock—An often repetitive digital signal whose edges are used to advance the outputs of counters and flip-flops.
CLR—Clear.
color burst—About nine cycles of a 3.579545 MHz signal that appear in a composite video waveform just following the horizontal sync pulse; used to synchronize the color circuitry in a tv receiver.
color killer—A circuit in a color tv set that defeats the color circuitry when a black and white transmission is received. Its purpose is to eliminate color noise and tinting from black and white pictures. Also, a circuit in the Apple II that removes the color burst in text mode. This is to allow the color killer in the associated tv set to function.
combinatorial logic—Logic consisting of just gates.
complement—Opposite state in two-state logic.
composite video—A video signal containing sync and blanking information in addition to picture information.
CRT—Cathode ray tube; usually a monitor or terminal containing a crt.
CTRL—Control.
DeMorgan's Theorem—States that $\overline{A \cdot B} = \overline{A} + \overline{B}$, and that $\overline{A+B} = \overline{A} \cdot \overline{B}$.
dc—Direct current.
DMA—Direct memory access; the ability of peripherals to directly access the main memory of a system without going through the microprocessor.
don't care—A signal whose logic state will not affect circuit operation.
dynamic—Depending on continuous clocking or sequencing for proper operation.
EPROM—Erasable programmable read-only memory.
equalizing interval—The portion of a video waveform just prior to and just following the vertical sync pulse.
falling edge—Signal transition from high to low.
FCC—Federal Communications Commission.
ferrite bead—A toroid or cylinder of magnetic material that is threaded with a wire to make an inductor.
fetch—To read from memory.
field—One complete scan of a crt face by an electron beam.
firmware—Software executing in ROM.
flag—A bit or signal that stores a binary state, such as on or off, ready or not-ready, set or cleared.
frame—One complete picture displayed on the face of a crt by a scanning electron beam; may consist of more than one field.
garbage—Data that is in an indeterminate or unstable state.
glitch—A usually short and usually undesirable level change in a logic signal.
high—A digital signal voltage of about 4 volts.

high-order—Bits representing the larger place values of a binary number.
HIRES—High resolution.
hold time—The time after a clock edge during which the input data to a flip-flop or other clocked IC must remain stable.
hue—Color tint (red, blue, etc.).
Hz—Hertz (cycles per second).
IC—Integrated circuit.
INH—Inhibit.
interlace—The process by which the lines of two or more fields are interleaved on the face of a crt to create a video frame.
I/O—Input/output.
IRQ—Interrupt request.
K or k—Kilo. (× 1000 when dealing with ohms, hertz, etc., × 1024 when dealing with memory locations, etc.)
KBD—Keyboard.
LORES—Low resolution.
low—A digital signal voltage of about 0.
low-order—Bits representing the smaller place values of a binary number.
LSB—Least significant bit (the lowest order bit).
luminance—The brightness or black and white portion of a color video signal.
M—Mega (× 1,000,000).
mask—To cause to ignore.
mother board—The main logic board into which peripheral boards plug.
mS—Millisecond (0.001 second).
MSB—Most significant bit (the highest order bit).
negative true—A low level represents a logic 1.
NMI—Non-maskable interrupt.
non-interlaced—The video technique where each frame consists of one field; that is, the field and frame are identical (see interlaced).
non-maskable—Incapable of being ignored.
nS—Nanosecond (10^{-9} second).
off—High-impedance state of three-state logic.
on—Low-impedance state (either 0 or 1) of three-state logic.
op code—Operation code; the first byte of an instruction.
open collector—A logic output with two states: low (about 0 volt) and high impedance.
overscan—The loss of picture information caused by the electron beam scanning beyond the edges of a crt.
PC—Printed circuit.
PDL—Paddle.
period—Reciprocal of frequency.
pixel—Picture element.
positive true—A high signal represents a logic 1.
PROM—Programmable read-only memory.
RAM—Random access memory.
RAS—Row address strobe (strobes row address into 4116 RAM).
RDY—Ready.
refresh—The process by which the data contents of a dynamic RAM are maintained at their correct values by continuously clocking the IC.
REPT—Repeat.

RES—Reset.
retrace—The return of the scanning electron beam to the left of a crt after displaying a line. Also the beam's return to the top after displaying a field.
RF—Radio frequency.
RFI—Radio-frequency interference.
rising edge—Signal transition from low to high.
ROM—Read-only memory.
R/W—Read/write.
saturation—1. The intensity of color in a video signal (red is more saturated than pink).
2. The state of a linear device (such as an operational amplifier) that is operating outside its linear range.
serration—One of several narrow pulses within the vertical sync pulse. Serrations serve to maintain horizontal synchronization during the vertical sync pulse.
setup time—The time prior to a clock edge during which the data input to a flip-flop or other clocked IC must be stable.
Soft 5—A pull-up to a high TTL level.
Soft Switch—A register that can be set or reset under software control. The register then acts as a switch to control a hardware function.
solder pad—A PC board foil pattern to which wire jumpers are soldered.
STB—Strobe.
subcarrier—A carrier that modulates a main carrier. The subcarrier itself is modulated by information to be transmitted (such as the color information in a video signal).
SW—Switch.
sync—Synchronization.
transceiver—A bidirectional buffer.
TTL—Transistor-transistor logic.
UART—Universal Asynchronous Receiver/Transmitter.
V—Volts.
V_{be}—The voltage between the base and emitter of a transister; about 0.6 volt for a forward-biased silicon transistor.
wait state—An extra clock cycle inserted into the normal memory cycle of a microprocessor; used to accommodate peripherals with long access times.
$—Indicates a hexadecimal number, for example: $C0FF.
μS—Microsecond (10^{-6} second).
ϕ—Phase.
+—Logical OR.
•—Logical AND.
−—Logical NOT ($\overline{A}$ = NOT A).

CHAPTER 2

The Apple II® Block Diagram

In this chapter we describe the Apple II at the block diagram level. The material will probably not be new to most readers. It may be useful, however, to read the chapter for review. And of course if you read this material, we can then be assured of speaking the same language in later chapters.

We will discuss each of the blocks in the diagram and follow the signal paths for several major computer functions through the Apple. The Apple II block diagram is shown in Fig. 2-1.*

BASIC ARCHITECTURE AND BUSES

At first glance the Apple II's basic architecture appears to be standard for a single board microcomputer. There are several significantly uncommon features, however, and we will point them out as we go.

Microprocessor

At the heart of the Apple II is the 6502 microprocessor (A in Fig. 2-1*). The 6502 is an 8-bit processor. This means that it operates on data in chunks of eight bits, or one byte. The 6502 can directly address 64K bytes of memory. Thus, it outputs a 16-bit address.

Input/output (I/O) operations in the 6502 are memory mapped. This means that I/O or peripheral devices share the same 64K address space with the memory. There is no separate I/O address space as provided in some microprocessors, such as the 8080. In addition to the 8 data lines and 16 address lines, there are various clock and control lines connected to the 6502. These will be described shortly.

Buses

There are three major buses in the Apple II: the 16-bit address bus, the 8-bit data bus, and the control bus. These buses run throughout the computer and appear at the eight peripheral I/O connectors.

Address Bus—The 16 address lines from the 6502 are buffered by a three-state driver (B in Fig. 2-1*) which then drives the address bus. This driver can be turned

off (switched to the high impedance state) by signal $\overline{\text{DMA}}$ from the control bus. The function of $\overline{\text{DMA}}$ will be described shortly.

Data Bus—On write cycles, the eight data lines from the 6502 are buffered by transceiver C which then drives the data bus. On read cycles, signal R/W (read/write) reverses the drive direction of transceiver C. This allows data from the data bus to pass to the 6502.

Control Bus—The major control bus signals are the interrupt, ready, reset, read/write, $\overline{\text{DMA}}$, and clock lines. There are two interrupt lines that allow peripherals to signal the 6502 that they need its immediate attention. The first of these lines is $\overline{\text{IRQ}}$ (interrupt request). It can be selectively ignored (masked) by the 6502. The second interrupt line is $\overline{\text{NMI}}$ (non-maskable interrupt). It cannot be ignored; the 6502 always responds to $\overline{\text{NMI}}$.

The ready line allows "slow" peripherals to momentarily stop the 6502 while they fetch their data and put it on the bus. The reset line allows any device connected to it to reset (initialize) all other devices connected to the reset line.

We have already mentioned the read/write line. Its function is to control the direction of data transfers on the data bus. Data is *read from memory* or I/O devices into the 6502. Data is *written to memory* or I/O devices from the 6502.

Direct memory access (DMA) refers to the ability of peripheral devices to exchange data directly with the system's memory without the need of first sending data through the microprocessor. During a DMA cycle, the $\overline{\text{DMA}}$ line turns off bus driver B so that the DMA device can put its own address on the bus. Also during DMA cycles, transceiver C does not drive the data bus. This frees that bus for data transfers between the DMA device and the system memory.

Clocks—The system clocks are the key to controlling the timing of data and address transfers on the buses. The clocks are also used throughout the Apple for such functions as video generation. The clocks have their origin in crystal oscillator D. Its output of about 14 MHz is used by clock generator T to produce the system clocks. One of these clocks (at about 1 MHz) is supplied to the 6502. The 6502 uses this 1 MHz clock to time its accesses to the buses. Memory and I/O devices use the same clock to synchronize their bus accesses with the 6502.

A read or a write operation can take place in one period of the 1 MHz clock. While the 6502 is running a program, it is executing individual instructions of that program. Each instruction executes in an integer number of clock cycles. On each clock cycle, the 6502 either writes to the bus, reads from the bus, or performs an internal operation. Each instruction consists of a mixture of these cycle types. The shortest instruction is two clock cycles and the longest is seven.

The 6502, the buses, and such topics as interrupts and DMA will be discussed in Chapter 6. The clock generator will be discussed in Chapter 3.

MEMORY

Two types of memory are provided on the Apple II mother board: ROM (read-only memory) and RAM (random access memory). The mother board can contain up to 12K bytes of ROM and up to 48K bytes of RAM.

ROM

Up to six 2K byte ROMs (E) can be installed on the mother board. Address decoder F decodes the high-order address bits to provide individual chip select lines for the six ROMs. The low-order address bits connect directly to the ROMs. On ROM read cy-

cles, the ROM's data output is placed directly on the data bus. Fig. 2-2* shows the main address and data paths through the Apple for a ROM read cycle. ROM cycles will be discussed in detail in Chapter 6.

RAM

Up to 48K bytes of RAM (G) can be installed on the mother board. Individual RAM locations are selected for read/write operations by addresses on the address bus. The address passes through address multiplexer H on its way to the RAM. The data input (DI) of the RAM connects directly to the data bus. This is the source of data for RAM write cycles. Fig. 2-3* shows the main address and data paths through the Apple for RAM write cycles.

On RAM read cycles, the data output (DO) of the RAM is stored in latch I. The read data then passes through data multiplexer J to the data bus. Fig. 2-4* shows the main address and data paths through the Apple for RAM read cycles. The RAM cycles will be discussed in detail in Chapters 5 and 6.

INPUT/OUTPUT

On-Board I/O

A computer may have a large memory and a fast processor, but it is of no use unless the computer can communicate with humans or other machines. That communication is the purpose of the computer's I/O (input/output) facilities. In the Apple II, some of the most frequently used I/O devices are located on the mother board. These on-board I/O devices include the speaker, cassette I/O, game I/O, and keyboard (K). Although the use of on-board I/O by the Apple is not unique, it is a distinguishing feature of the Apple II.

Write to I/O—Individual address select lines for the on-board I/O are provided by address decoder F. The 6502 processor writes to on-board I/O devices using these address lines; it does not use the data bus. For example, the 6502 may address one location to turn an I/O function *on*. The 6502 may then address another location to turn that same function *off*. It is the act of addressing specific locations that performs the functions. The data bus is not used.

Read from I/O—When the 6502 reads from an on-board I/O device, the data bus *is* used. Most on-board I/O devices provide just one bit, bit 7. This single data bit (D7) connects directly to the data bus as shown in Fig. 2-1*. The keyboard is an exception since it provides seven data bits. Fig. 2-5* shows the address and data paths for a keyboard read cycle. When the 6502 reads the keyboard, the keyboard data passes through multiplexer J. Keyboard cycles are described in Chapter 6.

Soft Switches—The soft switches are output locations that let the Apple configure its own hardware. The soft switches are set under software control to configure the video circuitry for various display modes. Their names of TEXT MODE, MIX MODE, PAGE 2, and HIRES MODE suggest this application.

On-board I/O is discussed in detail in Chapter 7. The soft switches are also discussed in Chapters 5 and 8.

Peripheral I/O

There is a need for flexibility and expansion beyond the on-board I/O. To meet this need, the Apple II mother board is provided with eight 50-pin peripheral connectors (L in Fig. 2-1*). The address, data, and control buses appear at each connector. Each

connector is also provided with individual select lines from address decoder F. The select lines reduce the amount of address decoding circuitry needed on each peripheral card. This feature of the Apple II will be discussed in detail in Chapter 6.

VIDEO

The video output of the Apple contains text and graphics information for display on a monitor, or tv set. Video signals require high-frequency components that are too fast to be generated directly by the 6502 microprocessor. (If you are not familiar with video signals, you may wish to review Appendix A at this time.) As a result of the high frequencies, dedicated hardware is provided to generate the Apple's video output. Of course, software-control gives the 6502 command over this hardware to generate specific text and graphics patterns.

The patterns that are to be displayed on the screen are generated by the 6502 and stored as data in the RAM in the Apple. The video circuitry then reads the data from the RAM, converts it to a video format, and sends it to the video output for display on a monitor. The accesses to RAM by the 6502 and the video circuitry are time-shared as described here. During the first half of a 6502 cycle, the video circuitry reads from RAM. During the second half of the cycle, the 6502 reads from RAM or writes to RAM.

In the sections that follow, we will examine the blocks in Fig. 2-1* that comprise the video circuitry.

Video Address Generator

Using the system clock as a reference, video address generator M creates a 15-bit *video address*. The video address consists of six horizontal bits and nine vertical bits. The address is continuously running through an incrementing sequence that repeats about 60 times a second. During each pass through the sequence, the vertical address has 262 different values. For each value of vertical address, the horizontal address increments through 65 counts.

Together, the horizontal and vertical addresses can select a location anywhere on the screen. Each screen location has a height of one scan line, and a width of about one microsecond. The video address also includes locations that do not appear on the screen since they occur during horizontal or vertical blanking.

Each location on the screen corresponds to a unique video address. As we will see shortly, the scanning electron beam in the crt is synchronized to the video address. One function of the video address is to "tell" other hardware sections the current screen location of the beam. The other hardware sections then fetch the appropriate character or graphics symbol for display at that location.

Fig. 2-1* shows a feedback path from the video address generator to the clock generator. This feedback signal delays the system clock by about 140 nS every horizontal scan line. The purpose of this unique arrangement is to simplify the generation of color graphics—it will be described in Chapter 3.

Memory Mapper

The video address is not a bit-for-bit equal of the memory address used to store the corresponding screen data. As a result, the video address must be converted to a memory address in order to fetch the correct screen data from memory. Memory mapper N performs this function. Screen locations go into the mapper, and the corresponding memory location comes out.

The memory mapper will map to different memory address blocks under control of the soft switches. In HIRES mode, for example, the video address maps to a different memory range than it would in text mode.

Fig. 2-6* shows the path through the Apple for a video cycle. The video address goes into the memory mapper and the memory address comes out. During video cycles, address multiplexer H connects the output of the mapper to the address input of the RAM. Data out of the RAM is then stored in latch I. The latched data then appears as an input to video generator P.

Video Generator

The video generator configures the latched data into video data that it sends to mixer Q. In the mixer, the video data is combined with the sync and color burst signals to become the composite video output.

The conversion process in the video generator is controlled by the soft switches. In LORES mode, for example, the video generator acts on the data in a manner different from the way it would act on the same data in text mode.

Sync Generator

Sync generator R uses the video address to generate the sync, color burst, and blanking signals. SYNC and COLOR BURST are mixed with the video in mixer Q. The sync component of the composite video output allows the scanning electron beam in the display crt to synchronize with the video address. The color burst component allows the color circuitry in the display device to synchronize with the internal color reference clock of the Apple. BLANKING connects to the video generator where it forces the video signal to go black during the horizontal and vertical blanking intervals.

POWER SUPPLY

The Apple II is equipped with a *switching power supply* that provides +5 V, +12 V, −5 V and −12 V to the mother board, keyboard, and peripheral I/O connectors. Switching power supplies are noted for their higher efficiency and lower bulk than more conventional designs. The power supply design of the Apple eliminates the need for a heavy line transformer. The ac line input is rectified, then converted to a high-frequency ac. This high-frequency ac is then coupled to the secondary of the supply via a small transformer.

Basic Operation

Fig. 2-7 is a simplified diagram of the power supply. Bridge rectifier CR1 rectifies the ac line input to provide a dc potential that is filtered by C1. This dc potential causes a current to flow through a primary winding of transformer TR1 when Q3 is on. Transistor Q3 switches on and off at a high frequency. On each cycle, energy is stored in TR1 while Q3 is on. When Q3 turns off, this energy is coupled to the secondary windings and causes current to flow in the output load.

Circuitry is provided to control the conduction of Q3 via its base. This control circuitry derives its operating potential from a second primary winding of TR1.

At the secondary, rectifier diodes and filter capacitors are used to obtain the four dc output potentials.

Regulation—Transistor Q4 is wired as a comparator that senses the voltage on the +5 V output. Resistor R15 and zener diode CR19 derive a reference voltage from

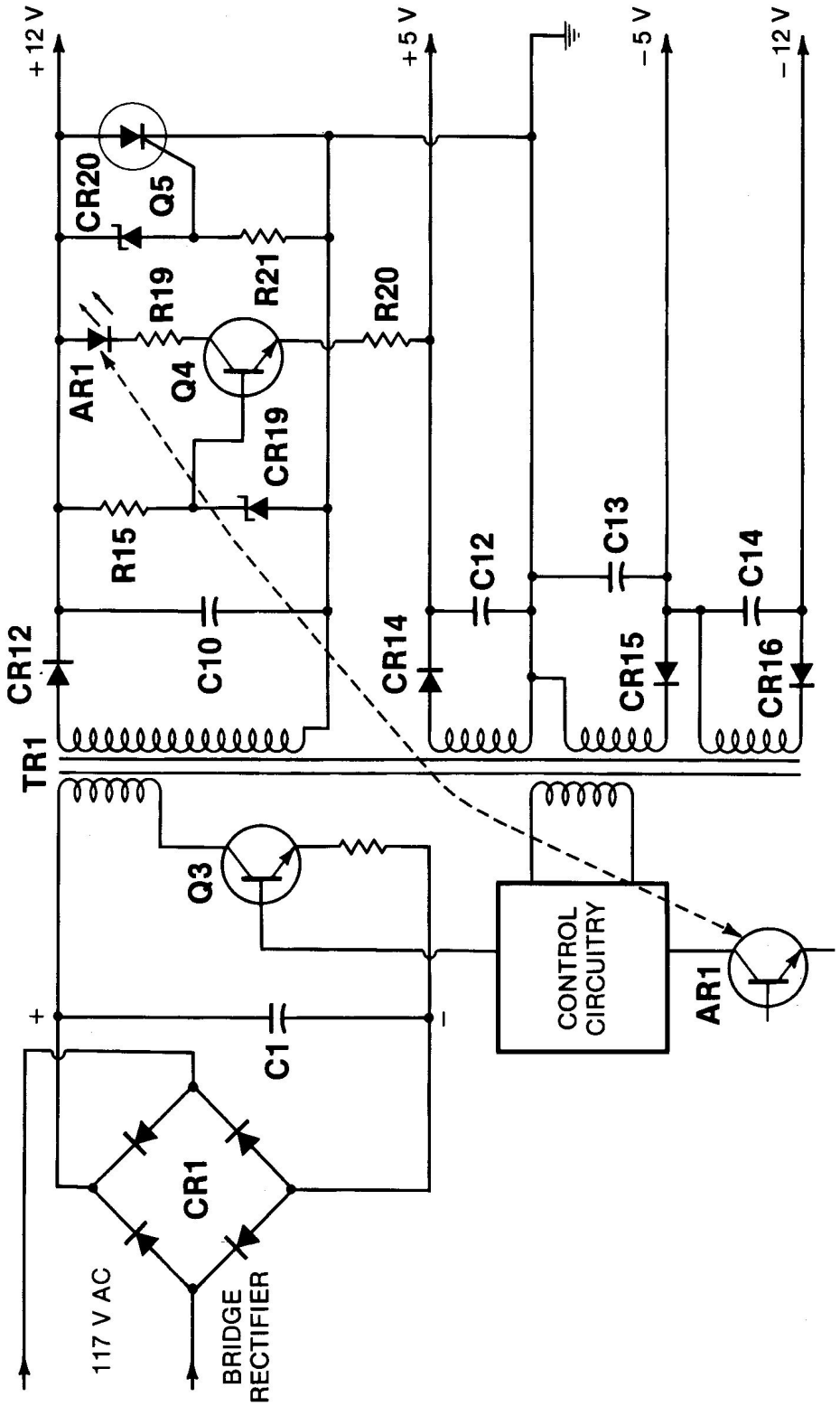


Fig. 2-7. Power supply—simplified diagram.

the +12 V output. This reference is applied to the base of Q4. The emitter of Q4 is connected through R20 to the +5 V output. When Q4 conducts, current flows through the LED (light emitting diode) that is part of optical coupler AR1. The amount of light emitted is a function of the voltage on the +5 V output. If the +5 V output decreases, for example, Q4 conducts more heavily and the LED emits more light. This light is coupled to the phototransistor portion of AR1. Increasing light causes the phototransistor to conduct more heavily. Via the control circuitry, this feedback causes Q3 to deliver more energy to TR1. This will in turn increase the voltage on the +5 V output. The other three voltages will track the regulation of the +5 V output.

Overvoltage—In the event of a fault that causes an overvoltage, the increased potential on the +12 V output will cause zener diode CR20 to conduct. This makes the gate of silicon controlled rectifier Q5 go positive—Q5 conducts, shorting the +12 V output to ground. This action will shut down the whole supply.

The complete power supply schematic is shown in Fig. C-23*. **CAUTION:** This schematic is reprinted directly from the *Apple II Reference Manual.* It has not been verified for accuracy with respect to any product shipped by Apple Computer. You are also cautioned against attempting to repair or modify your power supply. Much of the circuitry is not isolated from the ac line input and thus contains hazardous voltages. If you wish to read additional material on the power supply, obtain a copy of U.S. Patent No. 4,130,862.

SUMMARY

In this chapter we have presented a brief overview of the Apple II computer. At the heart of the Apple II is a basic single-board microcomputer consisting of microprocessor, buses, bus drivers, bus transceivers, ROM, and RAM. This basic architecture is enhanced by the inclusion of on-board I/O for many of the frequently used I/O functions. Expandability is provided by eight peripheral I/O connectors. The use of these connectors is simplified by an address decoding scheme that provides individual select lines for each connector.

A large part of the Apple II hardware consists of the built-in video text and graphics capability. Since the video screen memory resides in the address space of the microprocessor, the microprocessor can quickly output data to the screen with a simple memory write operation. Access to the screen memory is time-shared by the microprocessor and the video circuitry. This is an efficient arrangement that does not slow the processor.

In the chapters that follow, we will examine in detail the elements that make up the Apple II block diagram. We start in the next chapter with the clock generator.

CHAPTER 3

Clock Generator and Horizontal Timing

We begin our discussion of the Apple II with signals that are essential for the operation of the computer, the clocks. While it might not be very exciting, a knowledge of the clocks will provide a basis for the explanation of many of the functions of the Apple. In this chapter we will derive the clocks and horizontal timing and make some interesting discoveries about the processing speed of the 6502.

Schematic reference: Figs. C-2* and C-3*.

OVERVIEW

Clocks

The Apple II circuit design is based on clocked logic. There are thus many flip-flops, counters, and shift registers. These circuit elements all perform their tasks upon receiving a signal transition (or edge) at their clock inputs. For example, a 74LS161 counter will count and update its output pins upon receiving a rising edge at its clock input. The counter then sits idle until the next clock edge is received. Clearly the faster the clock, the higher the throughput. If the clock is too fast, however, the circuitry may not have time to respond between clock edges. The 74LS161, for example, may not respond to clock frequencies higher than 25 MHz.

Other devices in the Apple also use clocks. The 6502 microprocessor advances through a program one step at a time as it receives its clock input. The 6502 used in the Apple II is limited to a clock frequency of 1 MHz.

The type 4116 dynamic RAMs are also clocked devices. They load their address inputs upon receiving a clock edge. The RAMs in the Apple II are accessed at a 2 MHz rate.

The composite video output of the Apple contains an accurate 3.579545 MHz color reference signal. This signal is derived in the clock generator.

As you can see, there is a requirement in the Apple II for several clocks of different frequencies. By deriving the various clocks from a common high frequency master oscillator, all sections of the Apple are made to operate synchronously with each other. Synchronous operation is a highly desirable trait in a digital system.

The master oscillator is selected to be an integer multiple of the individual clock

frequency that is the most critical. In the Apple, the color reference is the most critical frequency, and the multiplication factor is 4. Thus, the master oscillator frequency is 4 × 3.579545 MHz = 14.31818 MHz. The Apple makes use of counters and shift registers to divide the master oscillator frequency into the various lower frequencies required.

Clock Generator Block Diagram

Fig. 3-1 is a block diagram of the clock generator. The ICs that comprise each block are noted in the figure. The master oscillator frequency is divided by 2 to generate a complementary pair of 7.2 MHz clocks, 7M and $\overline{7M}$. Signal 7M is divided by 2 to create COLOR REF. Signals 7M, $\overline{7M}$, COLOR REF, and 14M are used in the video generator.

Signal 14M is divided by 7 to generate $\overline{RAS}$, AX, $\overline{CAS}$, and Q3 (all at 2 MHz). Row address strobe ($\overline{RAS}$), AX, and $\overline{CAS}$ are used by the RAM and memory address multiplexer. Clock Q3 is a general purpose clock made available to the peripheral I/O connectors. Clock Q3 is divided by 2 to create a complementary pair of 1-MHz clocks, $\phi 0$ and $\phi 1$. These two clocks are used by the 6502 and all other devices that read from or write to the system data bus.

Combinatorial logic then derives the LD194 and $\overline{LDPS}$ clocks from the other

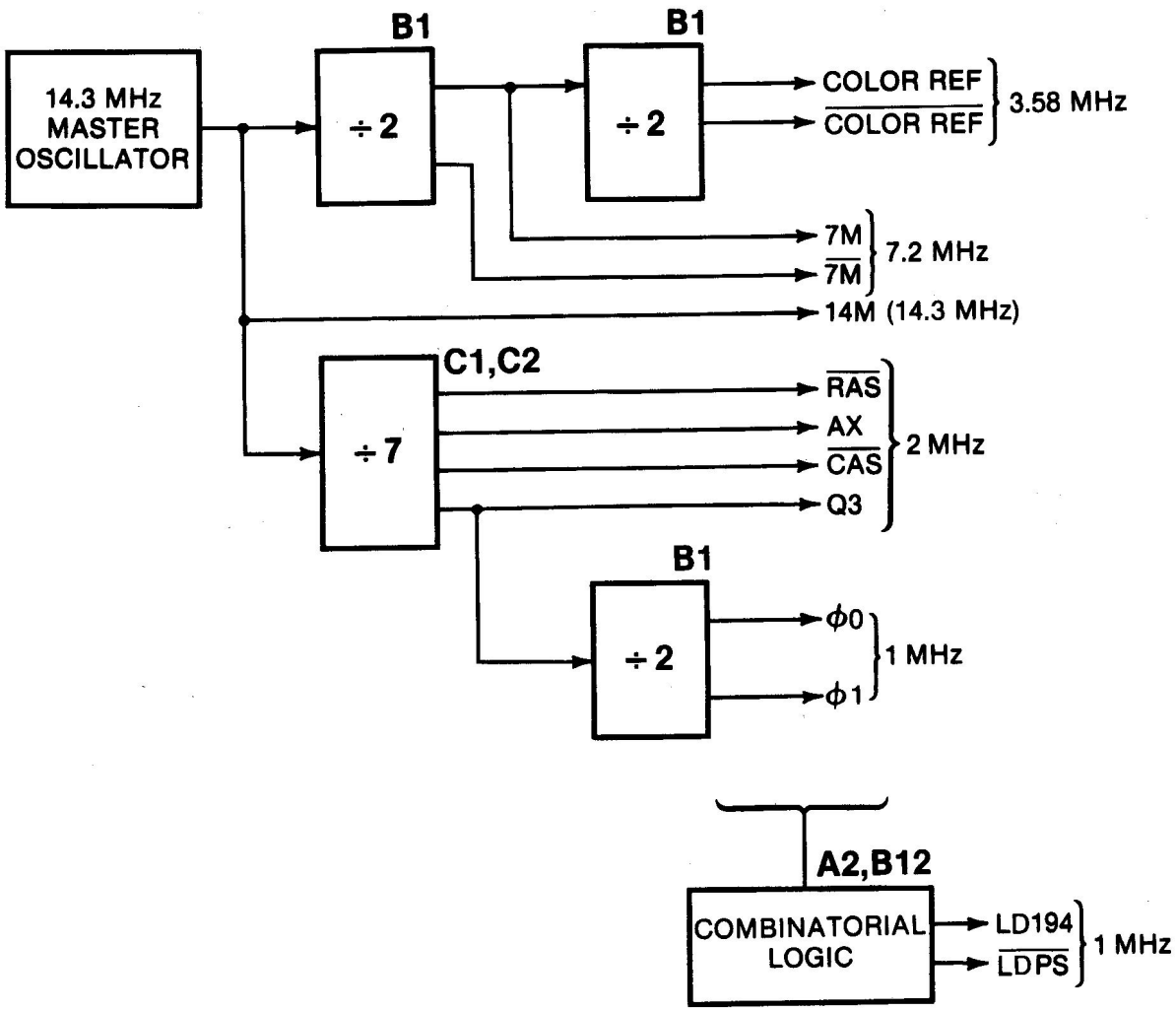


Fig. 3-1. Clock generator block diagram.

clocks previously described. LD194 is used in the video generator and $\overline{\text{LDPS}}$ is used in the video address generator.

Video Address Generator

The video address generator (Fig. C-3*) consists of a string of counters. These counters increment every microsecond to generate a new video address. The video address represents the instantaneous screen location of the scanning electron beam in the monitor connected to the Apple. The video address is used to fetch the video data that is to be displayed at each screen location. The video address is comprised of horizontal and vertical parts. The six bits of the horizontal part (H0 through H5) select one of 40 visible screen locations along a scan line. Later chapters will cover the use of the video address. In this chapter, however, we must examine the horizontal video address in detail. This is due to a feedback path from the video address generator to the clock generator. This feedback path plays a key role in the operation of the clock generator.

DETAILED CIRCUIT ANALYSIS

Clock Generator

Basic Clocks—Transistors Q1, Q2, and associated components (Fig. C-2*) form a crystal oscillator running at 14.31818 MHz. The oscillator output is buffered by B2-8 to become signal 14M, Fig. 3-2. The first section of quad flip-flop B1 (Fig. C-2*) is arranged to divide by 2, producing 7.15909 MHz at B1-15—this is signal 7M. Its complement $\overline{7M}$ appears at B1-14. The second section of B1 and B2-3 are arranged to divide 7M by 2, creating 3.579545 MHz at B1-2 and its complement COLOR REF at B1-3.

Shift Register C2—C2 is a 4-bit shift register arranged so that on each 14M clock rising pulse (pin 10) it either parallel loads (pin 9 low) or shifts (pin 9 high). When it shifts, it does so in the direction of Q0 toward Q3. Pins 2 and 3 are the serial inputs. Since they are grounded, a low shifts into Q0 on 14M rising while C2 is in the shift mode. In other words, on a shift, Q0 goes low, Q1 becomes Q0's previous value, Q2 becomes Q1's previous value, etc. Pin 9 (parallel enable) is tied to Q3, so after four shifts, the low at pins 2 and 3 shifts to Q3, putting C2 in the load mode. If we assume for now that D2-6 is always high, then the only external signal into C2 is the steady 14M clock. We can now determine the timing of C2's outputs.

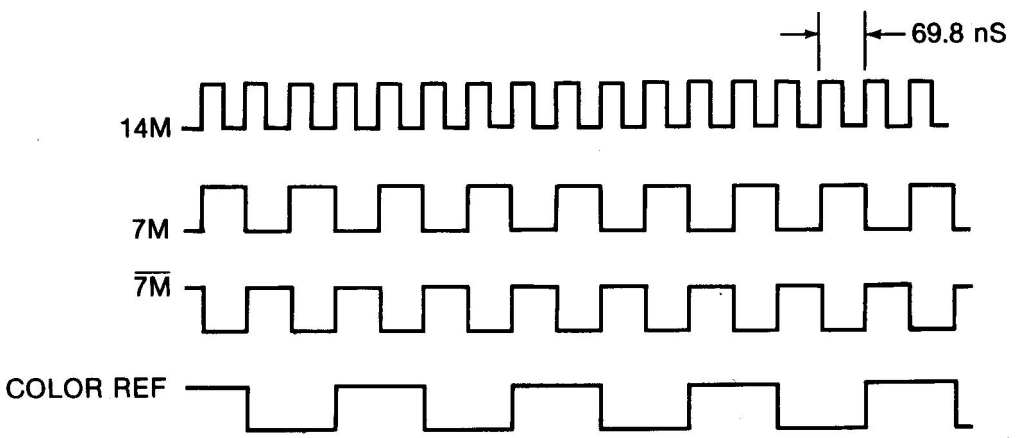


Fig. 3-2. Basic clock signals.

More Clocks—Referring to Fig. 3-3, let's start at the point where the low has just shifted into Q3 (point A). Shift register C2 is a synchronous load device, so after the next clock pulse the outputs will take on the values of the parallel inputs that existed before the clock pulse. Signal Q0 ($\overline{RAS}$) will remain low since PO (AX) was low. Signal Q1 (AX) will go high since P1 is assumed to be high for now. Signal Q2 ($\overline{CAS}$) will remain low since P2 (AX) was low. And finally, Q3 will stay low since P3 ($\overline{RAS}$) was low.

We are now at point B and still in the load mode. This time P0 and P2 (AX) are high, so on the next clock pulse, Q0 ($\overline{RAS}$) and Q2 ($\overline{CAS}$) will both go high. On the third load, Q3 goes high since P3 ($\overline{RAS}$) was high. We now stay in shift mode for four clock cycles before the low shifts into Q3 to put us back into the load mode. You can see the low's shifting along at points E, F, G, and H. The cycle begins again at point H which is the equivalent of point A. Since this cycle takes seven clock periods, $\overline{RAS}$, AX, $\overline{CAS}$, and Q3 all have a frequency of 14.31818 MHz ÷ 7 = 2.045454 MHz.

Microprocessor Clock—The circuit next divides by 2 to get the microprocessor's clock frequency of about 1 MHz. The third flip-flop in B1 (Fig. C-2) performs this function, aided by data selector C1-9. The 1 MHz signal that we are looking for will be at B1-10. The fourth B1 flip-flop delays B1-10 by one 14M clock pulse to generate $\phi 0$ at B1-7. Let's start with pins 10 and 7 of B1 low; this is in line with point A of Fig. 3-3. With both AX and $\phi 0$ low, select inputs S0 and S1 of C1 are low, and input pin 10 is gated to output pin 9. Thus, the low at B1-10 appears at B1-12. Both B1-10 and B1-7 remain low after the next clock edge, and we move to point B in Fig. 3-3.

At point B, AX is high, causing C1-9 (Fig. C-2*) to select input pin 11. Pin C1-11 connects to C2-11 and is high at this time. On the next clock pulse, B1-10 goes high and we move to point C (Fig. 3-3). The select inputs of C1 do not change, so on the next clock pulse, B1-10 stays high and B1-7 goes high to follow B1-10. We are now at point D. With B1-7 high, C1-9 selects input pin 13 which is high. Nothing changes on the following clocks until AX falls. Now C1-9 selects input pin 12 which is high. Still there is no change until Q3 falls at point H causing C1-9 to select input pin 10. Still no change until AX rises at point I. This selects C1-9 input pin 13 which is now low. On the next clock pulse, B1-10 falls and we move to point J. The select inputs of C1 do not change but B1-7 follows B1-10 to go low at point K. With B1-7 low, C1-9 selects input pin 11 which is low, so there is no change at point M. Signal AX is low at point N causing C1-9 to select input pin 10. Pin 10 is low, so there is no change at point O. The cycle begins again at point P which is the equivalent of point A.

Since this cycle takes 14 clock periods, signal $\phi 0$ has a frequency of 14.31818 ÷ 14 or about 1 MHz. Phase 1 at B1-6 is the complement of $\phi 0$. Phase $\phi 0$ and $\phi 1$ are the clocks for the 6502 microprocessor.

$\overline{LDPS}$ Clock—The signal $\overline{LDPS}$ is obtained by the combinatorial logic of gates B13-10 and A2-3 (Fig. C-2*). When AX is low *and* $\overline{CAS}$ is low *and* $\phi 0$ is high, then $\overline{LDPS}$ is low; see Fig. 3-3. Even though the waveforms in Figs. 3-2 and 3-3 are all derived from 14M, we have not combined them on one drawing yet since we do not know their relative phase. This will come later after we develop some of the video timing.

Horizontal Timing

In Fig. C-3* (Video Address Generator), D11, D12, D13, and D14 are a string of 4-bit binary counters chained together to provide the video address. The counters clock on the $\overline{LDPS}$ signal when it is rising. On each clock pulse, each counter will either

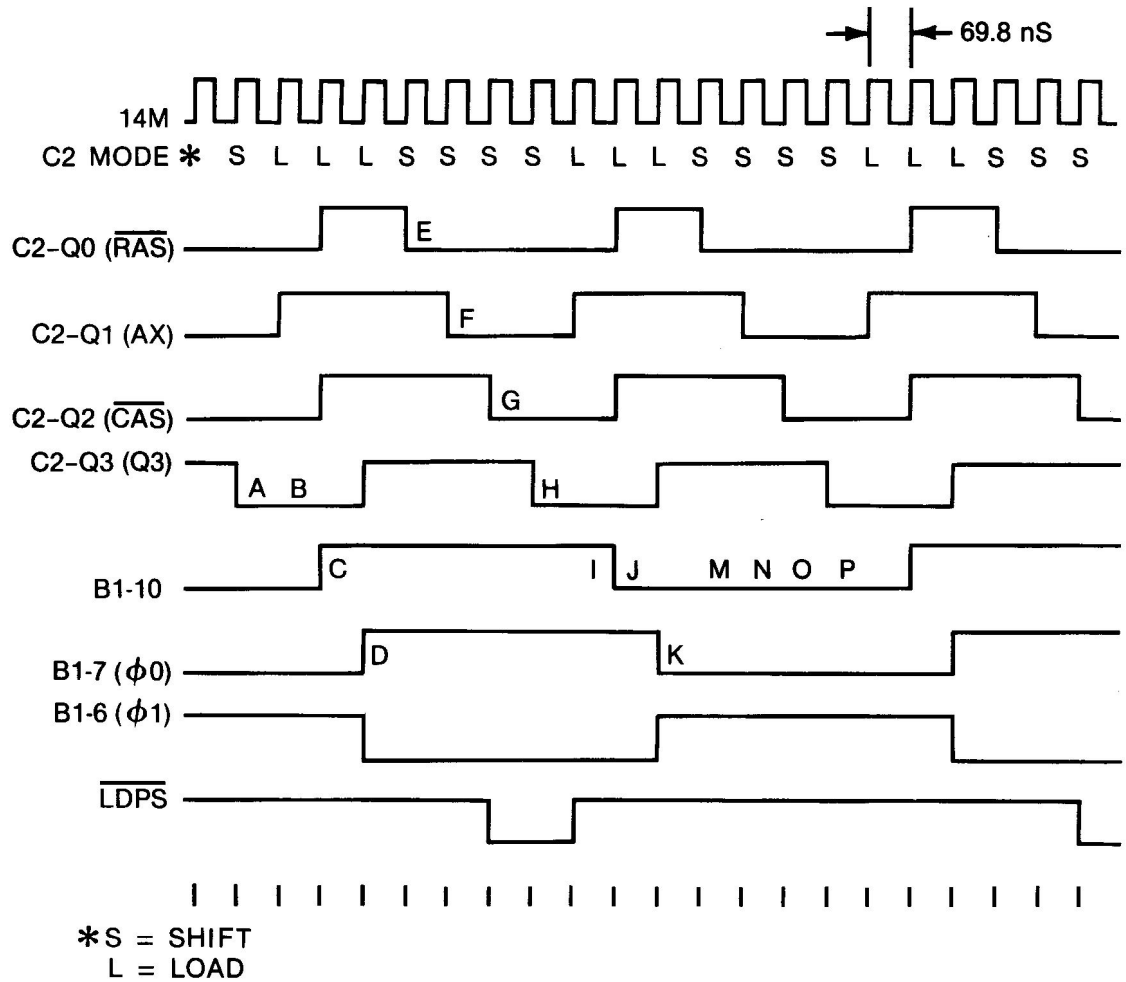


Fig. 3-3. Simplified system timing.

count (advance in binary sequence) or parallel load from its P inputs. The 74LS161 counts when pin 9 is high and loads when pin 9 is low.

The first seven counter bits (H0, H1, H2, H3, H4, H5, and $\overline{\text{HPE}}$) have the potential for 128 counts. Only 65 counts are used here. Let's start with all seven bits low, point A in Fig. 3-4*. Since $\overline{\text{HPE}}$ is low, D13 and D14 are in the load mode. On the next clock pulse ($\overline{\text{LDPS}}$ rising), the six low order bits load in zeros (so remain low), and $\overline{\text{HPE}}$ loads in a 1 (so goes high). We are now at point B and in the count mode. Counter D14 now counts up to 15 at which time its carry output (pin 15) goes high (point C in Fig. 3-4*). This enables D13 so that on the next clock D13 counts (H4 goes high). Counter D14 simply counts from 15 back to 0 and we are at point D. The enable input of D13 is removed (D14-15 low), so D13 does not count again until D14 reaches a count of 15 at E. Counter D13 counts at point F and again at points G and H. At point H, $\overline{\text{HPE}}$ is low again and the cycle restarts.

The decimal values of the bits H0 through H5 are shown in Fig. 3-4*. Note that there are two counts in the cycle with the value 0, and that there are 65 counts total per cycle. Counters D13 and D14 thus divide $\overline{\text{LDPS}}$ by 65. Each cycle (65 counts) through the waveforms of Fig. 3-4* corresponds to one horizontal line of the Apple video output. Each of the 65 counts corresponds to the width on the screen of one character in text mode (or one pixel in LORES mode). We know that the Apple

displays only 40 characters per line. It so happens that the other 25 counts occur during horizontal blanking. There will be much more on this in later chapters. First let's examine how $\overline{HPE}$ feeds back to the clock generator to affect shift register C2.

Note: The remainder of the waveforms in Fig. 3-4* will be discussed in Chapter 4.

The Extended Cycle

Fig. 3-5 is similar to Fig. 3-3, but some new signals have been added. Bit $\overline{HPE}$ will go low on an $\overline{LDPS}$ rising edge, let's say at point A. Bit $\overline{HPE}$ is inverted by B2-6 (Fig. C-2*) to put a high at D2-1. This meets one of the conditions necessary for D2-6 to go low. The others are AX low *and* $\overline{CAS}$ low *and* $\phi 0$ high *and* COLOR REF low (note that D2-2 is the complement of COLOR REF via B1). These conditions are the same as $\overline{LDPS}$ low *and* COLOR REF low. At this point we do not know the phase of COLOR REF relative to the other signals in Fig. 3-5. There are four possibilities shown as CR1, CR2, CR3, and CR4. The correct COLOR REF phase will turn out to

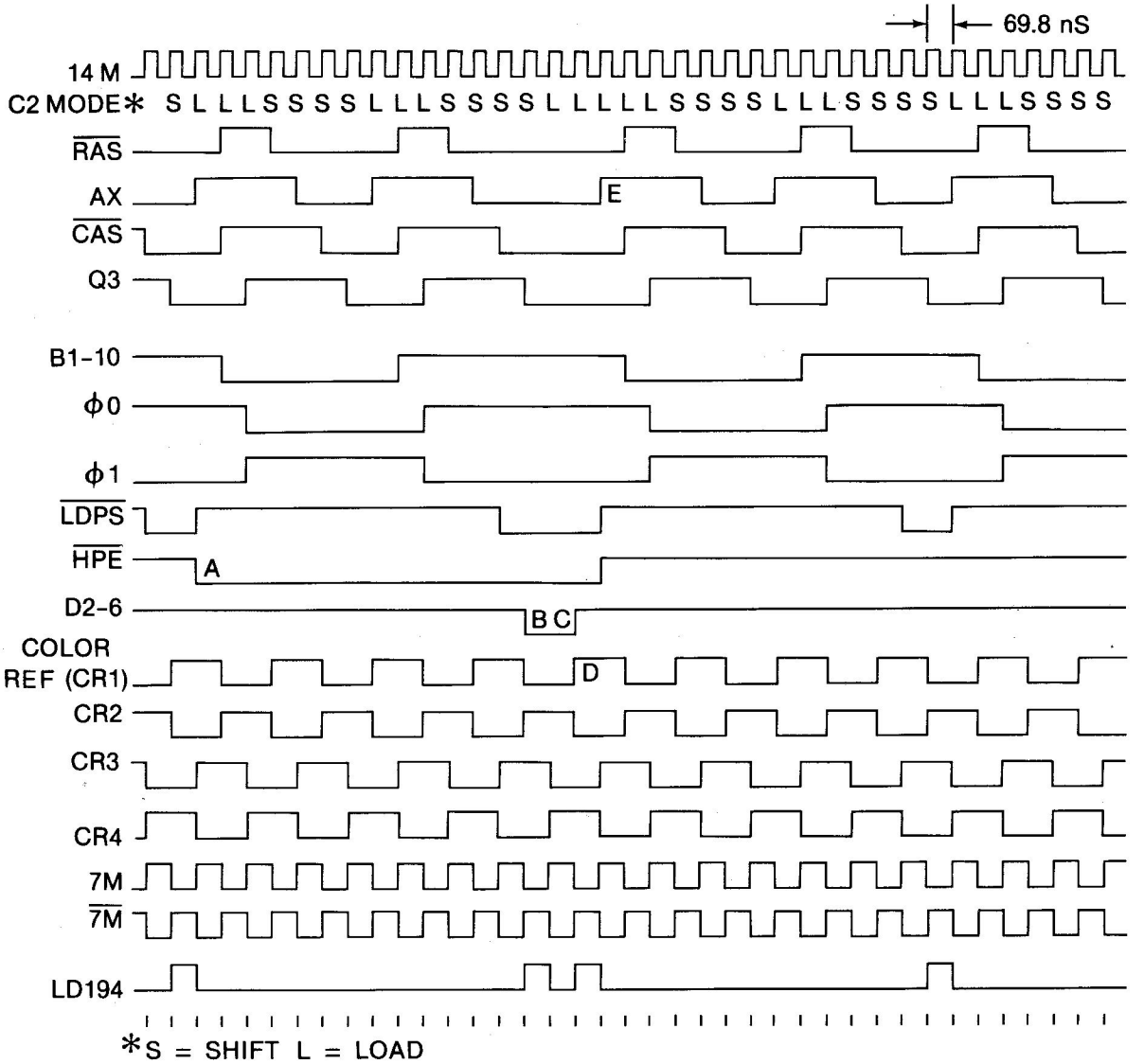


Fig. 3-5. System timing with extended cycles.

be CR1. Assume for now that our Apple powers up with COLOR REF shown as CR1; we will worry about the other three cases later.

Everything proceeds as previously described until point B when $\overline{\text{LDPS}}$ and COLOR REF are both low. Pulse D2-6 goes low, so that on the next clock cycle a low loads into AX instead of the usual high. We are now at point C and another low loads into AX. At point D, COLOR REF and D2-6 go high so that on the next clock pulse AX goes high (point E). The timing continues from this point as previously described in Fig. 3-3. The result of the low at $\overline{\text{HPE}}$ is to extend $\overline{\text{RAS}}$, AX, $\overline{\text{CAS}}$, Q3, $\phi 0$, $\phi 1$, and $\overline{\text{LDPS}}$ for two 14M clock cycles. The normal cycle is 14 ÷ 14.31818 MHz = 978 nS (nanoseconds). The extended cycle is 16 ÷ 14.31818 MHz = 1117 nS. In other words, most 6502 cycles are 978 nS long, but every 65th cycle is 1117 nS long. The average processor speed is then

$$14.31818 \text{ MHz} \times \frac{65}{(64 \times 14) + 16} = 1.020484 \text{ MHz}.$$

The horizontal video frequency is 1.020484 MHz ÷ 65 = 15.700 kHz.

Why the extended cycle every horizontal line? For the Apple to work with color tv sets, the frequency of COLOR REF must be very close to 3.579545 MHz. This has been achieved. Also, the horizontal frequency must be near (but need not be exactly) 15.734 kHz. Without the extended cycles, the circuit would divide 14M by (14 × 65). Since

$$\frac{14.31818 \text{ MHz}}{14 \times 65} = 15.734 \text{ kHz}$$

it would appear that the second requirement is easily met. It is, but there is a complication. Note that COLOR REF divided by this horizontal frequency is 3.579545 MHz ÷ 15.734 kHz = 227.5. Every horizontal line would contain 227 and *one-half* cycles of COLOR REF. COLOR REF would thus change phase by 180 degrees on each line relative to what it was on the previous line. There are two ways to deal with this:

1. Change the phase of the color data every line to compensate for the change in COLOR REF.
2. Choose another horizontal frequency such that one horizontal line would contain an integer number of cycles of COLOR REF.

The Apple II design uses Number 2. The 227.5 ratio is increased to 228 by extending each line by one-half period of COLOR REF. Note that one-half period of COLOR REF equals two full periods of 14M. The horizontal frequency now becomes 3.579545 MHz ÷ 228 = 15.700 kHz, a number derived previously. This horizontal frequency is close enough to 15.734 kHz for most tv sets. The extended cycle design of the Apple II is a major claim of U.S. Patent No. 4,136,359.

Synchronization of Clocks

What about the cases where COLOR REF and the other clocks do not start in correct phase at power up (Fig. 3-5)? We will take each of the alternative phases (CR2, CR3, and CR4) and examine what they do while $\overline{\text{HPE}}$ is low.

CR2–We start with CR2 in Fig. 3-6. At point A, D2-6 goes low. Register C2 shifts on the next clock pulse, so this low has no effect. D2-6 goes high again at point B. Since the cycle was not extended, CR2 will have an extra one-half period left over

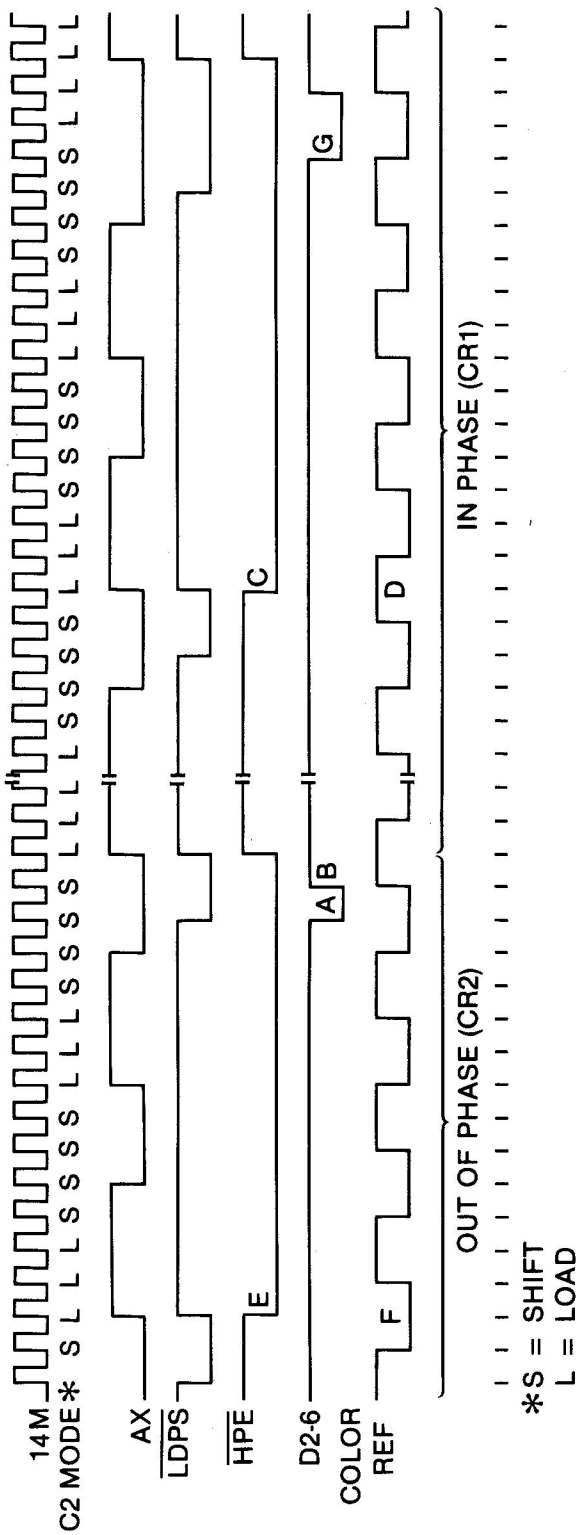


Fig. 3-6. Clock synchronization—CR2.

at the end of the line. The next time $\overline{HPE}$ is low (point C), COLOR REF will have shifted 180 degrees relative to $\overline{HPE}$. (Examine point C relative to point D, and point E relative to point F to see the 180 degree shift.) COLOR REF now has phase CR1. When D2-6 goes low at point G, the cycle is extended as previously described. The correct COLOR REF phase has now been established.

CR3—Next we examine CR3 in Fig. 3-7*. Phase CR3 is high (point A) when $\overline{LDPS}$ is low (point B). Thus, D2-6 does not go low, and there is no extended cycle. However, as in the case of CR2 previously, on the next $\overline{HPE}$ low, CR3 will have shifted 180 degrees. Phase CR3 has now become CR4. (To see the 180 degree shift, examine point C relative to point D, and point E relative to point F.)

CR4—We continue now with CR4. Signal D2-6 is low at point G and C2 shifts on the next clock pulse. Signal D2-6 is low again at point H and on the next clock

Table 3-1 Summary of Signals in Chapter 3

Signal	Description	Remarks	Frequency	Period
14M	14 MHz	Master Oscillator	14.31818 MHz	69.8 nS
7M	7 MHz		7.15909 MHz	140 nS
$\overline{7M}$	7 MHz	Complement of 7M	7.15909 MHz	140 nS
COLOR REF	Color Reference	Used for Color Burst	3.579545 MHz	279 nS
$\overline{RAS}$	Row Address Strobe	Used by RAM and Address Multiplexer	2.040968 MHz (Average)	489 nS or 628 nS
AX	Address Multiplex			
$\overline{CAS}$	Column Address Strobe			
Q3		General Purpose Clock		
ϕ0	Phase ϕ	6502 Clock	1.020484 MHz (Average)	978 nS or 1117 nS
ϕ1	Phase 1	Complement of ϕ0		
$\overline{LDPS}$	Load Parallel to Serial	Clock for Video Address Generator and Load for Text		
LD194	Load (74LS)194	Load for Graphics Data		
H0	Horizontal 0	Horizontal Video Address		
H1	Horizontal 1			
H2	Horizontal 2			
H3	Horizontal 3			
H4	Horizontal 4			
H5	Horizontal 5		15.700 kHz	63.7 μS
$\overline{HPE}$	Horizontal Parallel Enable	Load for Horizontal Video Address Counters		

pulse a low loads into AX. At point I, COLOR REF is high, so on the next clock pulse, AX loads a high (point J) and the timing continues as usual. The cycle has been extended by only one 14M period. The result of this extension is to convert COLOR REF from phase CR4 to phase CR1. (By examining point K relative to point M, and point N relative to point 0, you can see that the phase established immediately following the extension is indeed phase CR1.) Once phase CR1 has been established, there will be an extended cycle (point P) everytime $\overline{\text{HPE}}$ is low.

No matter which of the four phases COLOR REF takes at power up, it will quickly assume CR1. Phases CR2, CR3, and CR4 exist only momentarily at power up. It may avoid confusion to note that the *absolute* phase of COLOR REF does not really change after power up. The absolute phases of the other clocks *do* change as a result of extended cycles. References previously to a change in COLOR REF's phase mean a change *relative* to the other clocks.

SUMMARY

Now that the correct phase for COLOR REF has been established, 7M and $\overline{\text{7M}}$ can be transferred from Fig. 3-2 to Fig. 3-5. We can also add LD194 which is $\overline{\text{7M}}$ high *and* ϕ0 high *and* AX low *and* $\overline{\text{CAS}}$ low—LD194 is used later in the video generator. Table 3-1 is a summary of the signals presented so far.

In the next chapter we will complete the description of the video address generator. We will then examine the portion of the video generator that creates the video sync, video blanking, and color burst signals.

CHAPTER 4

Video Timing

This chapter continues the analysis of the video address generator that was started in Chapter 3. First we will finish the investigation of the D11, D12, D13, and D14 counter chain in order to develop the vertical timing. Then we will combine the horizontal and vertical timing to create the video sync, the video blanking, and the color burst signals.

Schematic reference: Figs. C-3*, C-16*, and C-20*.

OVERVIEW

Vertical Timing

Just as we needed a horizontal video address counter to define the horizontal position of the video data at any instant in time, we need a vertical address counter to define the vertical position of the video data. The horizontal and vertical address counters are not independent. The crt beam must trace a complete horizontal line before the vertical counter can advance to the next line (see Appendix A for an explanation of video techniques). The vertical counter advances once each time the horizontal counter runs through a complete count of 65. Each count of the vertical counter corresponds to a different horizontal line on the screen.

The Apple II video output consists of 262 lines. Of these lines, 70 are not visible on the screen since they occur during vertical blanking. The remaining 192 are used as follows:

1. TEXT Mode: TEXT mode characters are made of a 5 by 7 dot matrix in a 7 by 8 dot cell. The eight vertical dots per cell times 24 lines of text per screen = 192.
2. LORES: LORES pixels are 4 dots high by 7 dots wide. The 4 vertical dots per pixel times 48 pixels per screen vertically = 192.
3. HIRES: In HIRES, there are simply 192 dots vertically.

A counter with 262 possible states needs nine bits (eight bits would be too few since $2^8 = 256$). This 9-bit counter consists of two 4-bit counters (D11 and D12 (Fig. C-3*)) and the last stage of D13. The 9 bits are named VA, VB, VC, V0, V1, V2, V3, V4, and V5 in sequence from least to most significant. Their timing is shown in Fig. 4-1*.

In TEXT mode, the bits V0, V1, V2, V3, and V4 define one of 24 lines of text and the bits VA, VB, and VC define 1 of 8 vertical dots within the character cell. In HIRES mode, all 8 bits VA through V4 define 1 of 192 vertical dots. The ninth bit (V5) was needed to count beyond 256 to 262, but is not used elsewhere since 8 bits are sufficient to define 192 lines. Since only 8 of 9 bits are decoded, there are 6 addresses that repeat during each vertical scan. These addresses occur during vertical blanking and are not seen.

Video Sync

The Apple must provide sync signals in the video output for the external display. A horizontal sync pulse is provided between each horizontal line, and a vertical sync pulse is provided between each vertical scan. See Appendix A if you are not familiar with these concepts. The sync pulses are derived from the video address by combinatorial logic. The ICs involved are parts of A12, A14, B11, B14, C11, and C13 (Fig. C-20*). The sync output appears at C13-8. It is mixed with the video data and the color burst at transistor Q3. Transistor Q3 then provides the composite video output. The sync timing is shown in Figs. 4-1* and 4-2*.

Blanking

Blanking turns off the beam in the video display so it will be invisible during retrace. A blanking pulse is wider than the associated sync pulse which it straddles. The blanking signal is derived in gate C14-6 from signals already generated for SYNC. Its timing is shown in Figs. 4-1* and 4-2*. In standard tv, blanking has a unique voltage level in the composite signal. This level is slightly darker than black, but well above sync level. In the Apple II, blanking has the same level as black. In fact, the blanking signal is used in A9 (Fig. C-16*) to simply turn the video on and off. Off or low at A9-5 corresponds to black. More on this in the detailed analysis.

Color Burst

The color burst is a burst of 3.579545 MHz that appears in the composite video just following the horizontal sync pulse. A color display device uses the burst as a reference. COLOR BURST is obtained by gating $\overline{\text{COLOR}}$ $\overline{\text{REF}}$ with A14-1, B13-1, and B12-12 (Fig. C-20*). Its timing is shown in Figs. 4-2* and 4-3. Adjusting capacitor C3 (Fig. C-20*) trims the hue of the color display by delaying COLOR BURST a variable amount.

European TV

Signal V5, bow ties 12 and 16, and solder pads 13, 14 (Fig. C-20*), and 15 (Fig. C-3*) are used to configure the Apple II for use with European tv sets. The resulting operation is not covered in this book. See page 10 of the *Apple II Reference Manual* for more information.

DETAILED CIRCUIT ANALYSIS

VA

Signal VA is the vertical least significant bit (LSB) and toggles every horizontal line (Fig. 3-4*). Assume VA starts out low at point I. Counter D13 is in count mode and at a count of 7, so on $\overline{\text{LDPS}}$ rising, VA goes high (count 8). At J, $\overline{\text{HPE}}$ is low, loading VA into itself (D13-11 connected to D13-6 in Fig. C-3*). Signal VA stays high until

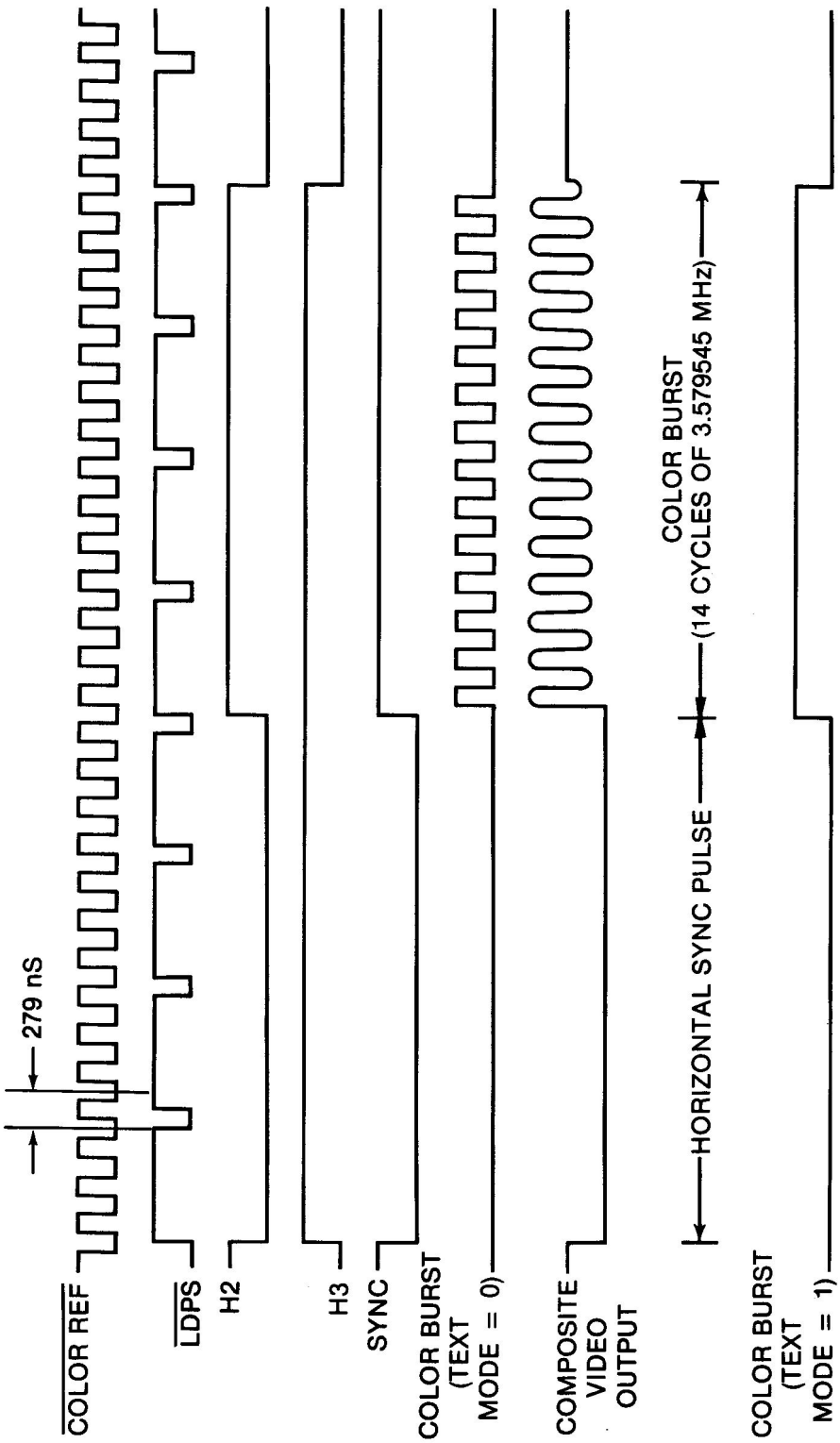


Fig. 4-3. Color burst signals.

the next time D13 is enabled (D14-15 high) *and* at a count of 15 (point K). On $\overline{\text{LDPS}}$ rising, D13 counts to 0 and VA goes low. Now $\overline{\text{HPE}}$ is low (point H), so VA loads into itself to stay low. This cycle repeats giving VA a period equal to two horizontal lines.

Before we leave Fig. 3-4*, let us examine the enable and load inputs of D12 and D11. Signal D13-15 goes high (point L) when D13 is enabled *and* at a count of 15. This enables D12. When D12 is enabled *and* at a count of 15 (D12-15 high), D11 is enabled (point M). When D11 is enabled *and* at a count of 15 (point N), C11-8 will go low (point P) to load D12 and D11. Note that D12 and D11 count (or load) on the clock *after* they are enabled, point Q.

The signals at points M, N, and P are drawn dotted to indicate that they are not active on every horizontal line. That is, on most lines these signals follow the solid trace. When the counters enable these signals, they follow the dotted trace. These signals are shown in Fig. 3-4* so that you may see their full width and timing relative to other signals. The signals will be shown again in later figures without ambiguity. These later figures have reduced time scales, however, and the signals will appear as mere "blips" with no width.

VB Through V5

Counters D12 and D11 are wired as an 8-bit counter. The counters load 125 decimal following a full count of 255 decimal. They count once per VA period (Fig. 3-4* point Q). This makes VB the next significant bit and V5 the most significant bit (MSB). Signals VA through V5 are the vertical video address bits and are shown in Fig. 4-1*. Let D12 and D11 start off with all bits high, point A. This makes C11-8 low when the counters are enabled, loading 125 decimal into the counters (point B). The two counters now count up to 255 decimal at point C where the cycle repeats. The vertical cycle consists of 255 − 125 + 1 = 131 counts of D12 and D11. Each of these counts consists of 2 states of the LSB (VA). Therefore, the video output of the Apple consists of 2 × 131 = 262 counts or lines per vertical cycle. We will find that 70 lines occur during vertical blanking, resulting in 192 lines visible on the screen.

The vertical frequency of the Apple is

$$\frac{15{,}700 \text{ Lines/Sec}}{262 \text{ Lines/Field}} = 59.92 \text{ fields/sec, or } 59.92 \text{ Hz}$$

Combinatorial Logic

The next step in our investigation requires picking a few intermediate points in the circuit for plotting in the figures. Analyzing gates B14-10, A12-4, and B14-13 we see that B14-13 = $\overline{\text{VC} + \text{V0} + \text{V1}}$. B14-13 will be low when any of VC, V0, or V1 are high. Examine B14-13 in Fig. 4-1* and you will see that this is so.

At this point we introduce Fig. 4-2* which combines portions of the horizontal and vertical timing in one figure. The left portion of the figure shows the last two visible lines at the bottom of the screen. This portion also shows the start of vertical blanking. The center portion shows the area near the vertical sync pulse. Finally, the right portion shows the end of vertical blanking and the first visible line at the top of the screen. These three separate portions are marked in Fig. 4-1* for reference.

The signals at the top of Fig. 4-2* (H2 through B14-13) have been derived previously. Continuing now with signal B14-1 by examining gates C11-10, B14-4, A14-4, A14-10, and B14-1 (Fig. C-20*) and using DeMorgan's theorem, we get B14-1 = V2 •

(H3 + H4 + H5). B14-1 is plotted in both Figs. 4-1* and 4-2*. Next, gates C13-C and B11-8 give us C13-C = B14-1 • B14-13 • V3 • V4. C13-C is plotted in both Figs. 4-1* and 4-2*. Note that signal C13-C appears internally to IC C13 only (see Fig. C-20*).

Blanking

Signal HBL (Horizontal Blanking) is derived in gate C13-6 (Fig. C-20*). HBL = $\overline{(\overline{H3} + \overline{H4})} \bullet \overline{H5}$, and is shown in Fig. 4-2*. BLANKING = HBL + (V3 • V4) and is shown in Figs. 4-1* and 4-2*. When BLANKING is high, the screen is blanked. In Fig. 4-1*, you can count the 192 visible lines between the top and bottom of the screen (points D to E). There are 70 lines in the vertical blanking interval (point E to point F). Note that vertical blanking extends quite far to either side of vertical sync. This is so no text or graphics will be lost at the top or bottom of the screen.

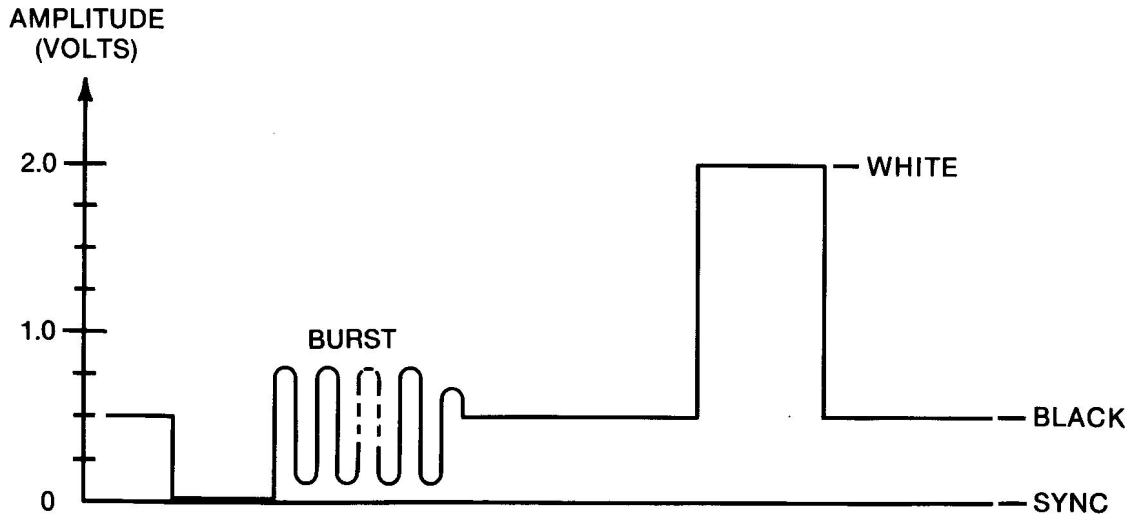
Selected portions of BLANKING are shown extended in Fig. 4-2*. The segment between points A and B is a horizontal blanking interval. (This segment is reduced to a thin line in Fig. 4-1*.) The interval between points B and C (Fig. 4-2*) is the unblanked or visible portion of the line; in this case the next to last line at the bottom of the screen. Note that this interval aligns with the video information in the COMPOSITE VIDEO OUTPUT for line 190. The screen is blanked again from points C to D and the segment from points D to E is the visible portion of line 191. Vertical blanking starts at point E. Horizontal blanking extends to either side of the horizontal sync pulses at points F and G. This is so no text or graphics are lost at the left or right sides of the screen. Vertical blanking ends at point H, and line 0 is at the top of the screen.

Let us deviate here to examine how BLANKING gets through A10, A9, and B10-5 to meet with SYNC and COLOR BURST at Q3 (Figs. C-16* and C-20*). Register A10 is a 4 bit shift register wired never to shift. It is used as a register with an *enable* (pins 9 and 10). It counts on 14M rising when LD194 is high (every 978 nS or one character width). Thus, BLANKING is delayed by one character going through A10 (in on pin 5, out on pin 13). Pin 13 of A10 is the strobe input for A9, a data selector. The data inputs of A9 contain the video from three sources: TEXT, LORES, and HIRES. The select inputs of A9 simply select the source. More on this in later chapters. For now, it is the blanking signal at the strobe input that turns output A9-5 on and off. When BLANKING is high, the strobe is off, and A9-5 is low corresponding to black. When BLANKING is low, the strobe is on, and the selected video passes to A9-5.

The video is delayed one 14M clock (70 nS) by B10-5 before reaching Q3. The point of all this is that BLANKING is delayed by the time it reaches Q3. The amount of the delay is 978 nS + 70 nS = 1.048 μS. This delay cannot be seen in Fig. 4-2* due to the compressed time scale of the figure.

Sync

Signal C13-D = $\overline{H2} \bullet H3 \bullet HBL$. It is shown in Fig. 4-2*. SYNC = $\overline{C13\text{-}C + C13\text{-}D}$ and is plotted in Figs. 4-1* and 4-2*. In Fig. 4-1*, each thin line is a horizontal sync pulse. The vertical sync pulse is four horizontal lines long and has three serrations. SYNC is shown expanded in Fig. 4-2* where you can see how the horizontal sync pulses at points F and G appear in the COMPOSITE VIDEO OUTPUT at points I and J. The vertical sync pulse starts at point K and extends to point L. One of the serrations is at point M.



APPLE II COMPOSITE VIDEO OUTPUT MEASURED OPEN CIRCUIT WITH R11 AT MAXIMUM.

SYNC LEVEL 0 V
BLACK LEVEL .5 V
WHITE LEVEL 2.0 V
BURST .7 V PEAK-PEAK CENTERED ABOUT .45 V

Fig. 4-4. Video levels.

Table 4-1. Summary of Signals in Chapter 4

Signal	Description	Frequency	Period	Remarks
VA	Vertical	7.85 kHz	127 μS	LSB
VB	Video	*		
VC	Address	*		
V0		*		
V1		*		
V2		*		
V3		*		
V4		59.92 Hz	16.7 mS	
V5		59.92 Hz	16.7 mS	MSB
BLANKING	C14-6	*		Combined Horizontal & Vertical Blanking
HBL	Horizontal Blanking	15.700 kHz	63.7 μS	Horizontal Blanking Only
SYNC	Synchronization	*		Combined Horizontal & Vertical Sync
COLOR BURST	B12-12	3.579545 MHz	279 nS	14 Cycles Every Horizontal Line
COMPOSITE VIDEO OUTPUT		H = 15.700 kHz V = 59.92 Hz	63.7 μS 16.7 mS	
VIDEO DATA				Video Without Sync or Burst

Complex waveforms, see Figs. 4-1 and 4-2*.

Color Burst

COLOR BURST = (TEXT MODE + $\overline{\text{COLOR}}$ $\overline{\text{REF}}$) • H2 • H3 • HBL (Fig. C-20*). We will assume for now that TEXT MODE = 0. COLOR BURST is plotted in Fig. 4-2* without attempting to show the individual cycles of COLOR REF. Color burst occurs immediately following each horizontal sync pulse and can be seen in the COMPOSITE VIDEO OUTPUT. There is no burst during the vertical sync pulse (point N for example) because COLOR BURST is essentially shorted out at Q3 when SYNC is low. In Fig. 4-3, COLOR BURST, COMPOSITE VIDEO OUTPUT, and one horizontal sync pulse are shown expanded. There are 14 cycles in COLOR BURST.

Coil L1, capacitor C2, and capacitor C3 form a tunable network parallel resonant at about the color burst frequency (Fig. C-20*). For example, if C3 is in the center of its range (about 25 pF), then

$$f = \frac{1}{2\pi\sqrt{LC}} = 3.6 \text{ MHz}$$

As C3 is tuned, COLOR BURST will experience a variable delay passing through R5 and the LC network. This is so the color hue of the Apple may be adjusted. The LC network also filters COLOR BURST to make it look more like a sine wave (see Fig. 4-3).

In TEXT mode, A14-2 is high, holding A14-1 low. This prevents COLOR REF from reaching COLOR BURST. Without a color burst present, the receiving tv activates its color killer and displays only black and white. *With* a color burst present, most tv sets would display text characters with an annoying color tint or fringe. To see this effect, observe the four lines of text in a LORES or HIRES *mixed mode* display.

Transistor Q6 and resistor R27 in Fig. C-20* are remnants from previous revisions. They perform no function since gate A14-1 is present.

Composite Video

The three signals VIDEO DATA, SYNC, and COLOR BURST are combined in Q3 (Fig. C-20*) to produce the COMPOSITE VIDEO OUTPUT. Transistor Q3 is an emitter follower with summing inputs. The values of the three input resistors R6, R7, and R8 are selected to give the required relative levels of sync, black, white, and burst in the output. Resistor R11 is a level adjust and R10 protects Q3 from shorts at the output.

To see the effect of the input resistor values, let's take white level as an example. For white, VIDEO DATA is high so we have about 3.5 V going into R7. During the visible portion of a line, SYNC is high and COLOR BURST is low, so we have about 3.5 V going into R8 and R6 grounded (through L1). Some easy math gives 2.6 V at the base of Q3. Subtracting V_{be} we get 2.0 V at the emitter of Q3. This will be the white level output voltage into an open circuit with R11 adjusted all the way up. Into a 75-ohm load, the output drops about 30% due mostly to the drop across R10. Fig. 4-4 is a sketch (to vertical scale) of the measured output of an Apple II.

SUMMARY

Table 4-1 is a summary of the signals introduced in this chapter.

In the next chapter we will discuss the memory system and explain how it is accessed by both the 6502 and the video circuitry. Also covered will be memory refresh and the mapping from screen locations to memory locations.

CHAPTER 5

The Memory System

One of the more clever design features of the Apple II is the efficient memory system. Memory access is shared by the 6502 microprocessor and the video display. This allows the video access to automatically refresh the memory; separate refresh circuitry is not needed.

First we review the 4116 dynamic RAM. You may wish to skip that discussion if you are already familiar with this IC. Then we introduce the memory circuitry and explain the reasons behind the design. Finally we analyze the circuit and signal waveforms in detail.

Schematic Reference: Figs. C-5*, C-6*, C-7*, and C-13*.

THE 4116

Memory Cell

The 4116 is a dynamic random access memory (RAM). It is organized 1 bit wide by 16K bits deep. Each of the 16K memory cells consists of a small capacitor and a transistor. The state (0 or 1) of the cell is stored as the presence or absence of charge on the capacitor. When the cell is accessed, the transistor turns on to connect the capacitor to the internal circuit. On a write cycle, the circuit charges or discharges the capacitor as appropriate for the state to be stored. On a read cycle, the circuit senses the presence or absence of charge as a 1 or a 0.

Refresh

The cell capacitor is so small that its charge is greatly reduced by the circuit that reads it. To compensate, the circuit restores the capacitor to its original charge at the end of the read cycle. Thus, cells that are read frequently are kept "refreshed." The charge on nonaccessed cells, however, will quickly leak away. To prevent this, cells that are not read with sufficient frequency must receive refresh cycles. A refresh cycle is nothing more than a dummy read cycle. There must be continuous read or refresh activity if the 4116 is to retain the correct data. This is why it is labeled "dynamic."

Multiplexed Address

The 16K cell locations inside the 4116 are selected by a 14 bit address (2^{14}= 16K). In order to conserve package size, these 14 bits are multiplexed onto 7 pins. Multiplexing means that the 14 bits are divided into two groups of seven each. Each time the memory is accessed, the first group of seven bits is placed on the address pins and strobed, then the second group of seven bits is placed on the address pins and strobed. After the second strobe, all 14 bits will be in the RAM, and a unique location will be selected.

The cells are arranged logically into 128 rows by 128 columns (128 × 128 = 16,384). The first seven address bits are named the *row address* since they select the row. The second set of seven bits is the *column address* and selects the column. The two strobes or clocks are named the $\overline{RAS}$ (*row address strobe*) and the $\overline{CAS}$ (*column address strobe*). The strobes are active low, and the address is clocked into the 4116 on the falling edge of the strobe.

When a cell is read, all cells in the selected row are refreshed. This means the complete IC can be refreshed by reading a cell in each row. Stated another way, to refresh all cells, perform 128 read cycles using all 128 different row addresses—the column address is a "don't-care." You can also omit the $\overline{CAS}$. This turns what was a read cycle into a refresh-only cycle.

Again, if the RAM is read often enough, the normal read cycles will keep it refreshed, and refresh-only cycles will not be needed. By often, we mean all 128 rows every 2 milliseconds.

If any 7 of the 14 address bits constantly sequence through all 128 states in less than 2 milliseconds, you should assign these 7 bits to the row address. This will give you free refreshes. Apart from this consideration, the 14 bits may be arbitrarily assigned to the rows and columns.

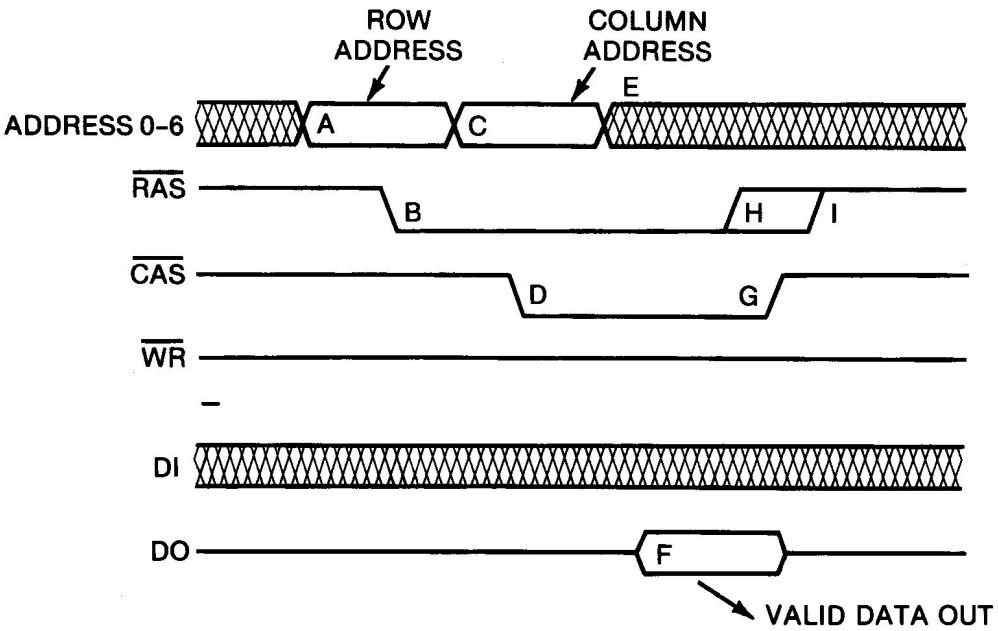
Read Cycle

Fig. 5-1A shows a typical 4116 read cycle. We start at point A, the time you place the row address on the address pins. After waiting the needed setup time, take $\overline{RAS}$ low to strobe the row address, point B. After a *hold time*, you may multiplex (change) the address to the column address, point C. After a *setup time*, take $\overline{CAS}$ low to strobe the column address, point D. After a hold time, you may change the address (it has become a don't-care), point E. Write ($\overline{WR}$) is active low and stays high during a read cycle. Data in (DI) is a don't-care on a read cycle. After the *access time* from $\overline{RAS}$ or $\overline{CAS}$ (whichever comes later), DO (data out) will change from high impedance to the value stored in the addressed cell, point F. Data out will remain valid until $\overline{CAS}$ goes high, point G. At that time, DO returns to high impedance. The $\overline{RAS}$ may return high either before or after $\overline{CAS}$, points H or I.

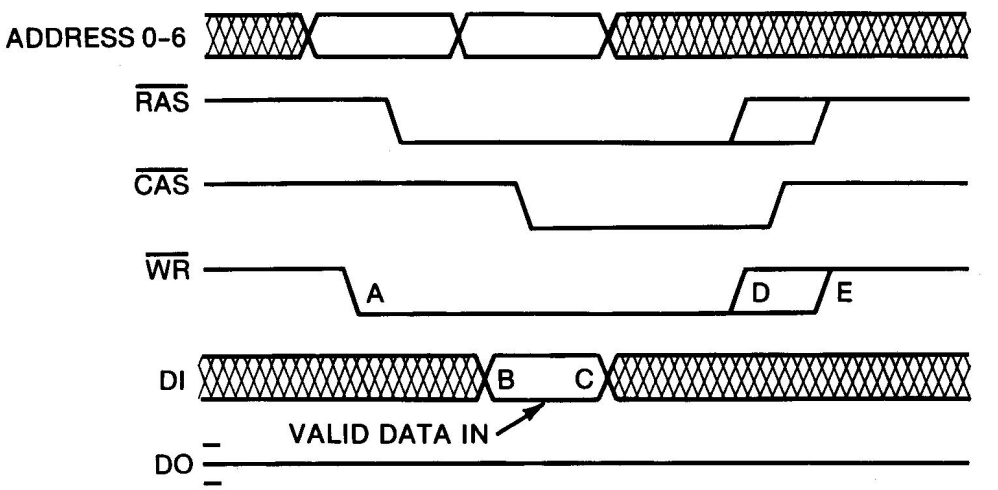
The setup, hold, and access times mentioned previously vary depending on the specified speed of the RAM. There are also minimum $\overline{RAS}$ and $\overline{CAS}$ low times and minimum times allowed between cycles. A common specification for data access time from $\overline{RAS}$ is 200 nanoseconds. All of the parameters may be found in a 4116 data sheet.

Write Cycle

Two different write cycles are possible with the 4116, *early-write* and *read-modify-write*. The Apple II uses an early-write cycle, shown in Fig. 5-1B. The sequence for address, $\overline{RAS}$, and $\overline{CAS}$ is the same in the write cycle as it was in the read cycle. In



(A) Read cycle.



(B) Early-write cycle.

Fig. 5-1. 4116 RAM cycles.

the early-write cycle, however, you take $\overline{WR}$ low (point A) prior to taking $\overline{CAS}$ low. You also set up valid data on DI (point B) prior to taking $\overline{CAS}$ low. On $\overline{CAS}$ falling, the data is clocked into the RAM. The DI must remain valid until point C to allow a hold time after $\overline{CAS}$ falls. Data out remains high impedance during an early-write cycle. At the end of the cycle, $\overline{WR}$ may return high either before or after $\overline{CAS}$, points D or E. Other cycle types are available with the 4116, but only the two shown in Fig. 5-1 are used in the Apple II.

Data stored in the RAM is not inverted. If you write a 1 (high), you read a 1 (high).

So far we have been discussing a single IC—a RAM one bit wide. In a typical application, the 4116 is used in groups of 8 ICs to give an 8-bit-wide byte. In this example, the eight ICs would give you 16K bytes of memory. Deeper memories (more addresses) are obtained by using additional groups of eight ICs.

OVERVIEW

Memory Array

The memory array of the Apple is eight ICs wide by three ICs deep (Fig. C-7*). This organization provides a memory space 8 bits wide by 48K deep. You may plug the RAMs into the mother board in groups of eight, giving you a 16K, 32K, or 48K byte capacity. Each group of eight ICs has a separate $\overline{CAS}$ line. Only one $\overline{CAS}$ line is activated at a time, selecting a unique 16K range.

The $\overline{RAS}$, $\overline{WR}$, and address lines of all RAMs are tied in common. The DI pins for each bit from all three groups are tied in common and connected directly to the data bus. The DO pins for each bit from all three groups are tied in common and connected to registers B5 and B8. From there the data finds its way back to the data bus via multiplexers B6 and B7 (Fig. C-13*).

Address Multiplexing

Fig. 5-2A is a simplified diagram of the RAM address multiplexing scheme. We mentioned that the RAM is shared by the 6502 and the video display. Multiplexer A provides this sharing function. It selects bits coming from either:

1. The 6502 via the address bus, or
2. The video address.

The selection is under control of $\phi 0$, one of the 6502 clocks. The 14 low order address bits (AD0–AD13) are divided into two groups of 7 bits each. Multiplexer B selects one or the other of these groups under control of signal AX (address multiplex). Similarly, 14 bits of the video address are divided into two groups for selection by multiplexer C. This selection is also under control of AX. Multiplexers B and C reduce 14 bit addresses to 7-bit row and column addresses for the 4116s.

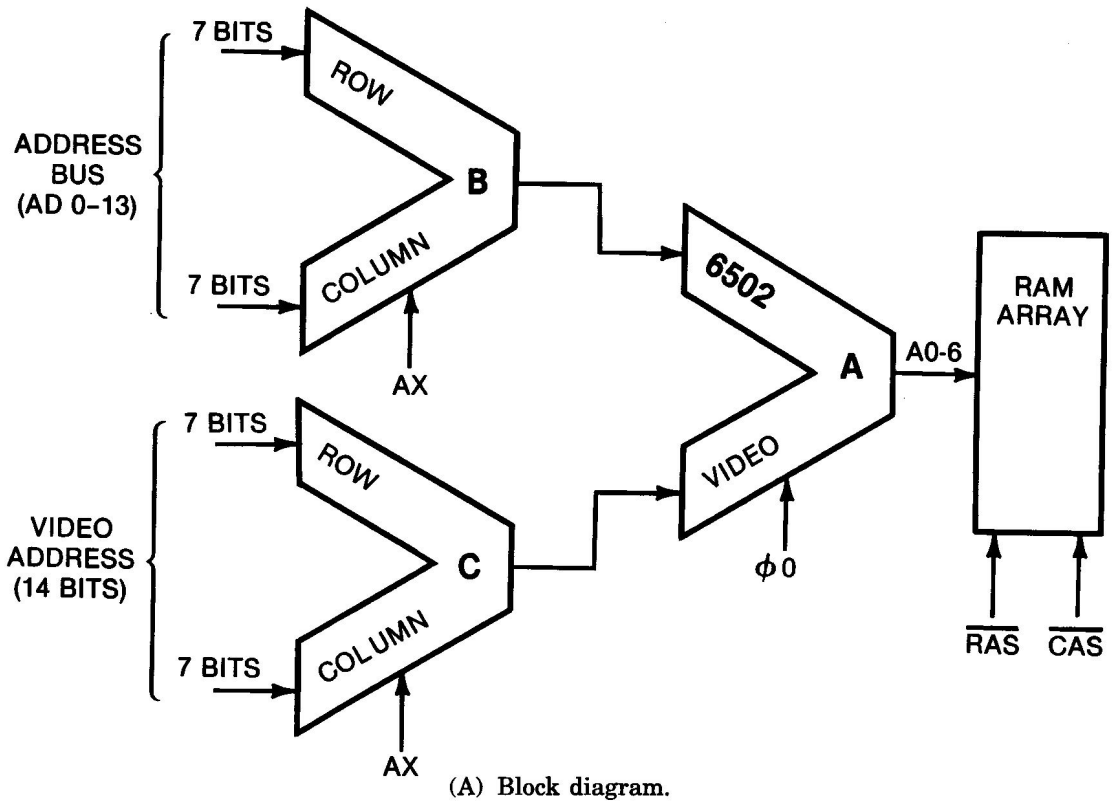
Do not confuse "rows and columns" as used in this section with rows and columns on the video screen or rows and columns of ICs on the mother board.

Fig. 5-2B shows the timing associated with the address multiplexing. The signals shown were derived in Chapter 3 and presented in Fig. 3-3. One period of $\phi 0$ (points A to C) corresponds to one 6502 cycle. We will learn more about the 6502 in Chapter 6, but for now we can say that the 6502 reads and writes memory while $\phi 0$ is high (points B to C). The video circuitry reads the memory while $\phi 0$ is low (points A to B).

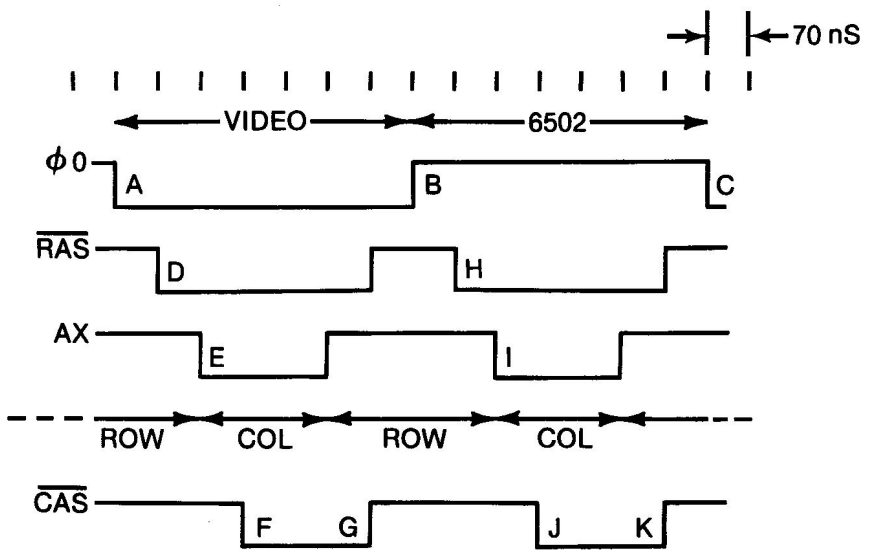
Let's follow the sequence. At point A, $\phi 0$ is low causing multiplexer A to select the video address. Signal AX is high causing multiplexer C to select the video row address. The $\overline{RAS}$ strobes the video row address at point D. At point E, AX goes low and multiplexer C selects the video column address. The $\overline{CAS}$ strobes the video address at point F. The video data is available from the RAM at about point G. Signals $\overline{RAS}$, AX, and $\overline{CAS}$ return high before point B. At point B, $\phi 0$ goes high causing multiplexer A to select the 6502 address bus. Signal AX is high causing multiplexer B to select the 6502 row address. Signal $\overline{RAS}$ strobes the 6502 row address into RAM at point H. Signal AX goes low at point I causing multiplexer B to select the 6502 column address. Signal $\overline{CAS}$ strobes the 6502 column address into RAM at point J. Data is available from the RAM at about point K. This sequence repeats every $\phi 0$ period, interleaving the video and 6502 accesses.

The multiplexers of Fig. 5-2A are implemented by ICs C1, E11, E12, and E13 (Figs. C-5* and C-6*).

In the previous discussion, we said that the 6502 puts the address on the bus.



(A) Block diagram.



(B) Timing diagram.

Fig. 5-2. RAM address multiplex.

This is not always true. If a peripheral device has DMA capability, then it may put the address on the bus. The address multiplexing scheme, however, will be the same. We will cover DMA in Chapter 6.

Video Memory Mapping

The video address consists of the 14 bits H0, H1, H2, H3, H4, H5, VA, VB, VC, V0, V1, V2, V3, and V4. Fourteen bits can address 16384 locations. The HIRES screen

needs only 7680 locations (192 × 40). The video addresses that occur during blanking account for the difference. If the 14 video address bits mapped directly to memory, we would need 16K of RAM for one HIRES screen. Over half of this RAM would be addressed during blanking and thus wasted.

We could map the 14 video address bits to 13 bits and use those 13 bits to address 8192 memory locations (2^{13}). Then only 512 bytes are wasted (8192 − 7680). Integrated circuit E14 (a 4-bit adder) does this by mapping 5 bits (H3, H4, H5, V3, and V4) to 4 bits. Thus, one HIRES screen uses 8192 or 8K bytes of RAM.

In HIRES mode, one byte stores only seven dots—horizontally. Additional dots are specified in vertical groups of eight by signals VA, VB, and VC. Compare this with TEXT mode where each byte stores a complete character, or with LORES mode where each byte stores a pair of pixels. In TEXT and LORES modes, VA, VB, and VC are not needed to address RAM. When these 3 bits are omitted, the remaining 10 bits address 1024 bytes. Thus, one TEXT or LORES screen uses 1024 or 1K bytes of RAM.

Part of the circuitry switches the 3 bits (VA, VB, and VC) in or out so we can address either 8K of memory for HIRES or 1K for TEXT (or LORES). To see how this works, we must introduce the signal HIRES which is generated in the video section. HIRES is low for TEXT and LORES modes, and it is high for HIRES mode. In mixed TEXT and HIRES mode, HIRES is high while the top of the screen is scanned, but goes low while the four lines of text at the bottom are scanned. When HIRES graphics are displayed, HIRES causes multiplexer C12 to connect VA, VB, and VC to the memory. When TEXT or LORES graphics are displayed, HIRES is low, turning off VA, VB, and VC at C12 and replacing them with the appropriate address to select one of the two text pages.

HIRES should not be confused with HIRES MODE, a different signal.

Signal PAGE 2 (F14-6 in Fig. C-12*) is the page 2 soft switch. PAGE 2 at C12 (Fig. C-6*) selects page 1 or page 2 TEXT (or LORES). PAGE 2 at J1 (via H1-8 in Fig. C-5*) selects page 1 or page 2 HIRES.

Refresh

It turns out that seven of the video address bits switch through all 128 combinations in about 2 milliseconds. These seven bits are assigned to the row addresses at multiplexer C of Fig. 5-2A. Thus, the memory refreshes are obtained without additional circuitry.

$\overline{\text{CAS}}$

Separate 16K blocks of RAM are selected by individual $\overline{\text{CAS}}$ lines. Multiplexer J1 (Fig. C-5*) selects two of its inputs to be AD15 and AD14 (6502 access) or ground and HIRES • PAGE 2 (video access). Decoder F2 decodes these two signals into one of three $\overline{\text{CAS}}$ lines.

Data Path

Input—The RAM data inputs simply connect to the data bus.

Output—The RAM data outputs are latched in registers B5 and B8 where they remain stable for a complete memory cycle. The video circuitry uses the latched data directly. However, for the 6502 to read the data, it must be put on the data bus. Multiplexers B6 and B7 do this, selecting either the latched RAM data or the keyboard output.

DETAILED CIRCUIT ANALYSIS

Memory Array and Data Path

We start our detailed analysis with the 24 IC memory array (Table 5-1).

Table 5-1. Physical and Logical IC Assignments

	Data Bits							
Address Range	0	1	2	3	4	5	6	7
32K – 48K	E3	E4	E5	E6	E7	E8	E9	E10
16K – 32K	D3	D4	D5	D6	D7	D8	D9	D10
0 – 16K	C3	C4	C5	C6	C7	C8	C9	C10

Front — Right — Top View

The $\overline{RAS}$, $\overline{WR}$, and address lines of all 24 ICs are tied in common. These inputs are largely capacitive and this can lead to reflections and ringing in a large array. Resistors RA02, RA03 (Fig. C-6*), R31, and R32 (Fig. C-5*) form terminations to alleviate this problem. The six low order memory address lines are driven by multiplexers E11, E12, and E13. The high order memory address line is driven by multiplexer C1-7.

$\overline{RAS}$—The $\overline{RAS}$ line for the array is the same line that was discussed in Chapter 3 and shown in Fig. 3-5. The $\overline{WR}$ line for the array is derived from the R/W (read/write) line on the system bus. This line is in turn driven by either the 6502 or by a peripheral with DMA capability.

$\overline{CAS}$—The $\overline{CAS}$ pins for each IC in a row of RAMs are tied in common. The $\overline{CAS}$ for each row is kept separate, however, since this is how the Apple selects unique 16K ranges. The driving device is one-of-four decoder F2 (Fig. C-5*). The decoder will take only one $\overline{CAS}$ low at a time, selecting one of three rows of RAMs. When the range 48K–64K is addressed, F2 will not activate any $\overline{CAS}$.

Data—The DI pin for each bit in the array is tied in common (Fig. C-7*). These eight lines then connect directly to the system data bus, D0 – D7. They are driven by either the 6502 or by a peripheral with DMA capability. The DO for each bit in the array is also tied in common. These eight lines are latched in registers B5 and B8 on $\overline{RAS}$ rising at the end of each read cycle. On video cycles, the latched data (signals DL0 – DL7) is used directly by the video generator. On 6502 or DMA cycles, the latched data is put on the data bus so that the 6502 or DMA device can read it. Multiplexers B6 and B7 (Fig. C-13*) perform this function under control of leads $\overline{RAM\ SEL}$ and $\overline{KBD}$. On a memory read in the range 0 to 48K, F2 (Fig. C-5*) will take low one of its output pins 10, 11, or 12. These lines pass to gate D2-8. Any input low at D2-8 will cause A2-5 to be high. Read/write is high (this is a read cycle) so $\overline{RAM\ SEL}$ goes low. The $\overline{RAM\ SEL}$ turns on B6 and B7 via their enable inputs. Then B6 and B7 drive the data bus. This is not a keyboard cycle, so $\overline{KBD}$ will be high at the B6 and B7 select inputs. This causes them to select the latched data to put on the data bus. The other function of B6 and B7 is to select the keyboard output when $\overline{KBD}$ is low. When neither the RAM nor the keyboard is selected, the B6 and B7 outputs remain high impedance.

Read Cycle—Fig. 5-3 shows the timing of memory read cycles. Note how video and 6502 cycles are interleaved. Signal $\overline{LDPS}$ rising at point A advances the video address at point B. With $\phi 0$ low and AX high, the row bits of the video address are

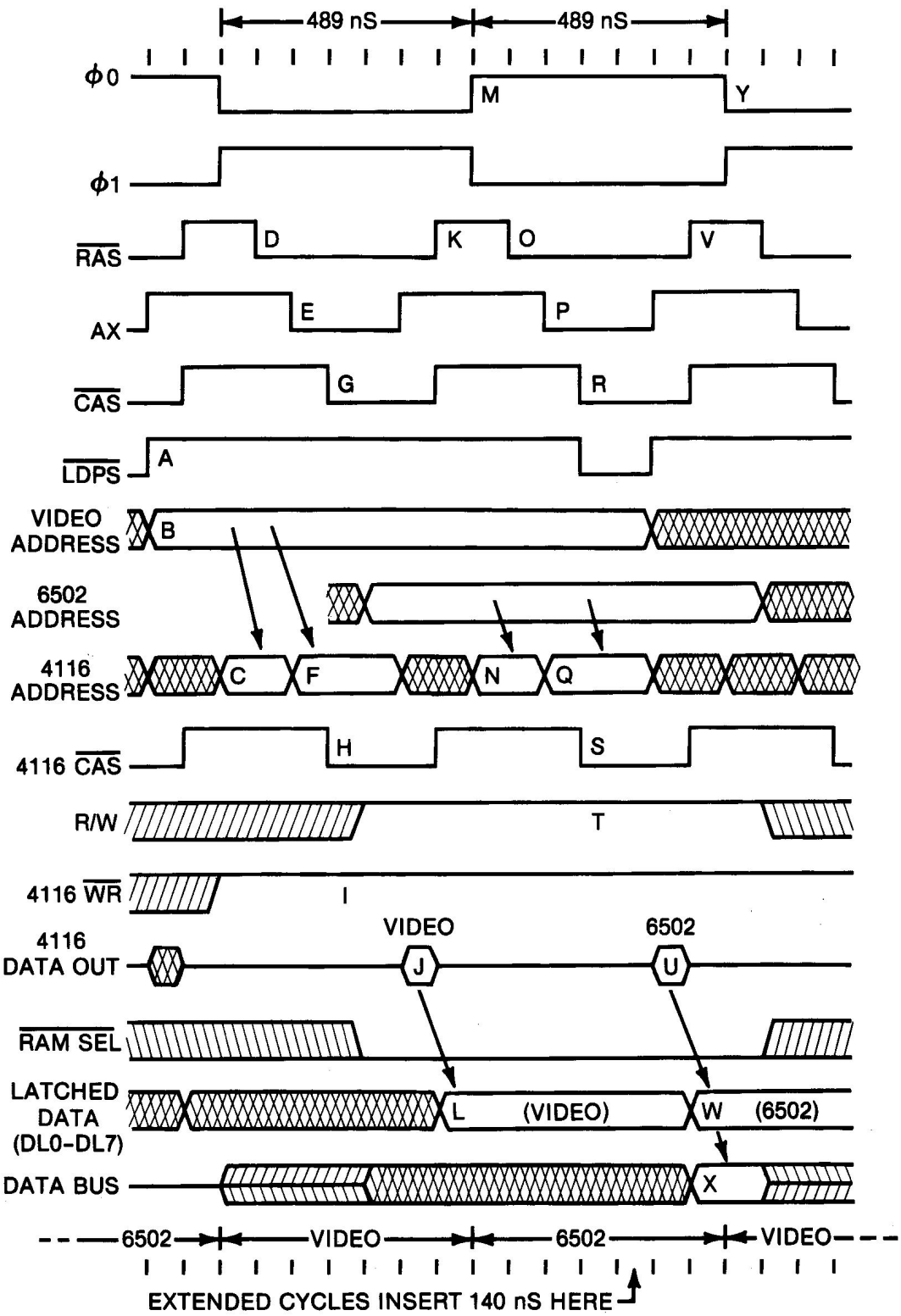


Fig. 5-3. RAM read cycle.

sent to the RAM at point C. They are strobed when $\overline{RAS}$ goes low at point D. AX goes low at point E to send the video column address bits to the RAM at point F. Meanwhile, the two high order address bits at decoder F2 prepare it to select one of the three 4116 $\overline{CAS}$ lines when $\overline{CAS}$ (F2-1) goes low at point G. When this happens, the selected 4116 $\overline{CAS}$ goes low (point H) to strobe the column address bits. When $\phi 1$ is high, 4116 $\overline{WR}$ is high via gate C14-11 (Fig. C-7*). Thus, 4116 $\overline{WR}$ is high at point I (Fig. 5-3) to start a 4116 read cycle when $\overline{CAS}$ goes low. After the 4116 access time from $\overline{CAS}$, the 4116 outputs go low impedance and contain the addressed data, point J. On $\overline{RAS}$ rising (point K), the 4116 data is latched into registers B5 and B8, point L. The latched data is now available for the video generator.

Now comes the 6502 read cycle. Thus, $\phi 0$ goes high at point M and AX is already high to send the 6502 row address bits to the 4116s at point N. They are strobed when $\overline{RAS}$ goes low at point O. Now AX goes low at point P to send the 6502 column address bits to the 4116s at point Q. Meanwhile, address bits AD14 and AD15 (Fig. C-5*) prepare decoder F2 to select one of the three 4116 $\overline{CAS}$ lines when $\overline{CAS}$ (F2-1) goes low at point R (Fig. 5-3). When this happens, the selected 4116 $\overline{CAS}$ goes low (point S) to strobe the 6502 column address. Note that the 4116 $\overline{CAS}$ shown in Fig. 5-3 represents all three 4116 $\overline{CAS}$ lines. Thus, the strobes at points H and S will not occur on the same 4116 $\overline{CAS}$ line unless the two cycles shown address the same 16K range.

We are discussing a 6502 read cycle, so R/W and 4116 $\overline{WR}$ are both high at point T. Thus, a 4116 read cycle starts when $\overline{CAS}$ falls. After the access time, the 4116 data outputs go low impedance and contain the addressed data, point U. On $\overline{RAS}$ rising (point V), the 4116 data is latched (point W). The $\overline{RAM\ SEL}$ is low at this point, so multiplexers B6 and B7 drive the latched data onto the data bus (point X). The 6502 reads the data from the bus on $\phi 0$ falling, point Y. This concludes the cycle.

Recall from the clock discussion in Chapter 3 that every 65th $\phi 0$ period is extended by 140 nS. We have drawn Fig. 5-3 to represent a RAM read cycle without this extension. At the bottom of the figure we have indicated the location for the extension. All the waveforms from Fig. 5-3 are redrawn in Fig. 5-4 with the 140 nS extension added. Even though the timing is altered, the extended cycle of Fig. 5-4 functions just like the normal cycle of Fig. 5-3.

Write Cycle—Fig. 5-5 shows a RAM write cycle. It is interleaved between two video cycles just like the 6502 read cycle of Fig. 5-3. The video cycles are the same as described previously. (Note that video cycles always *read* from RAM.) On a write cycle, the address is strobed into RAM the same as on a read cycle. The address is divided into row and column bits at points A and B (Fig. 5-5) and these are strobed at points C and D. Since this is a write cycle, the 6502 takes R/W low at about point E. When $\phi 1$ falls at point F, 4116 $\overline{WR}$ goes low (point G). Since $\overline{WR}$ goes low before $\overline{CAS}$, the 4116 will enter an early-write cycle when $\overline{CAS}$ falls at point D. Meanwhile, the 6502 has output the data to be written at about point H. This data is driven onto the data bus and becomes stable at point I. The data bus is connected directly to the RAM inputs, so when $\overline{CAS}$ falls at point D, the data at point I is written into the RAM. Note that in a 4116 early-write cycle, the 4116 data out pin remains high impedance, point J.

Write cycles can be extended 140 nS just like read cycles. At the bottom of Fig. 5-5 we indicate where the extra 140 nS go.

Chapter 6 will discuss RAM read and write cycles again from the viewpoint of the 6502.

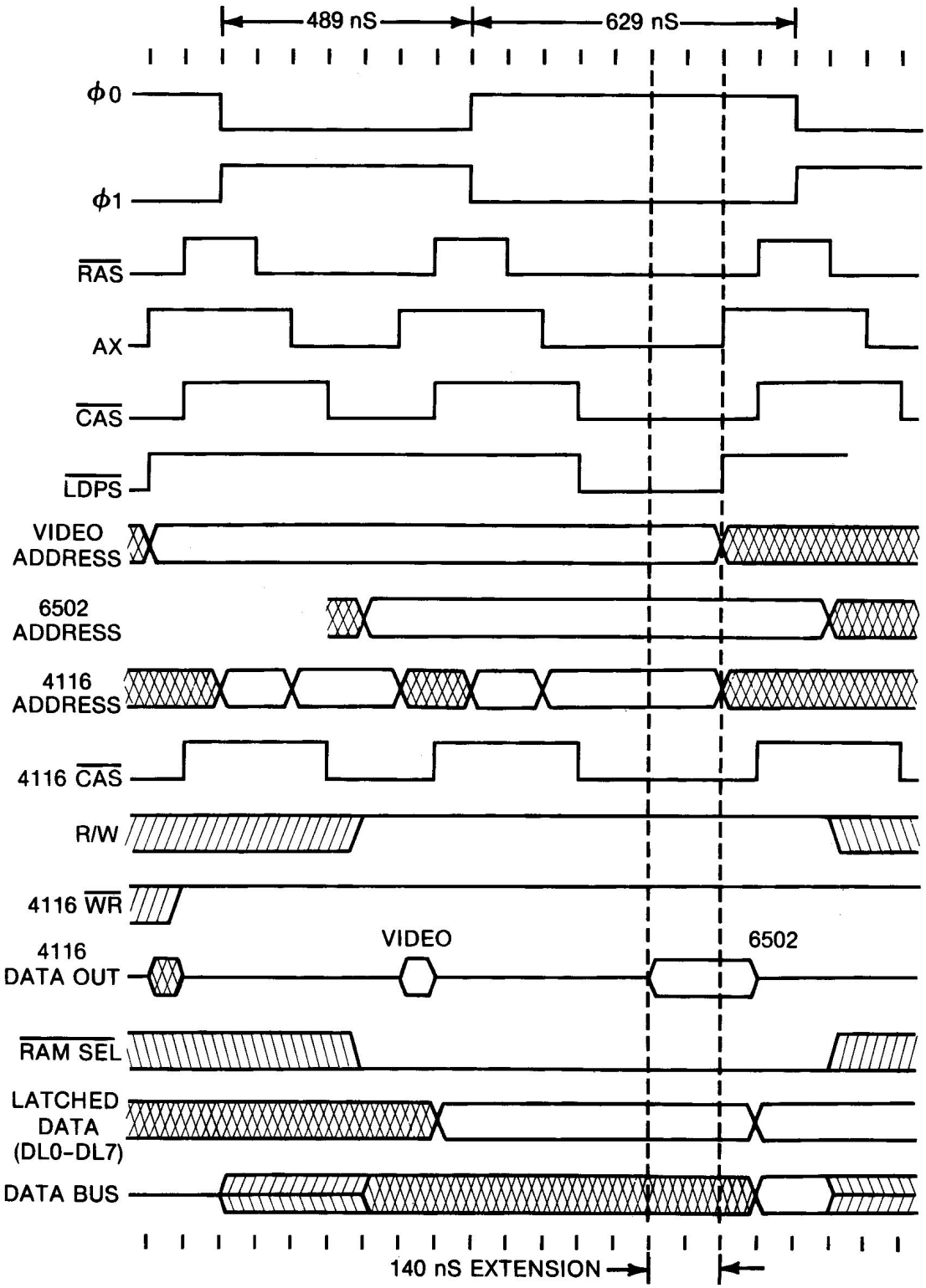


Fig. 5-4. RAM read cycle with extended ϕ0 clock.

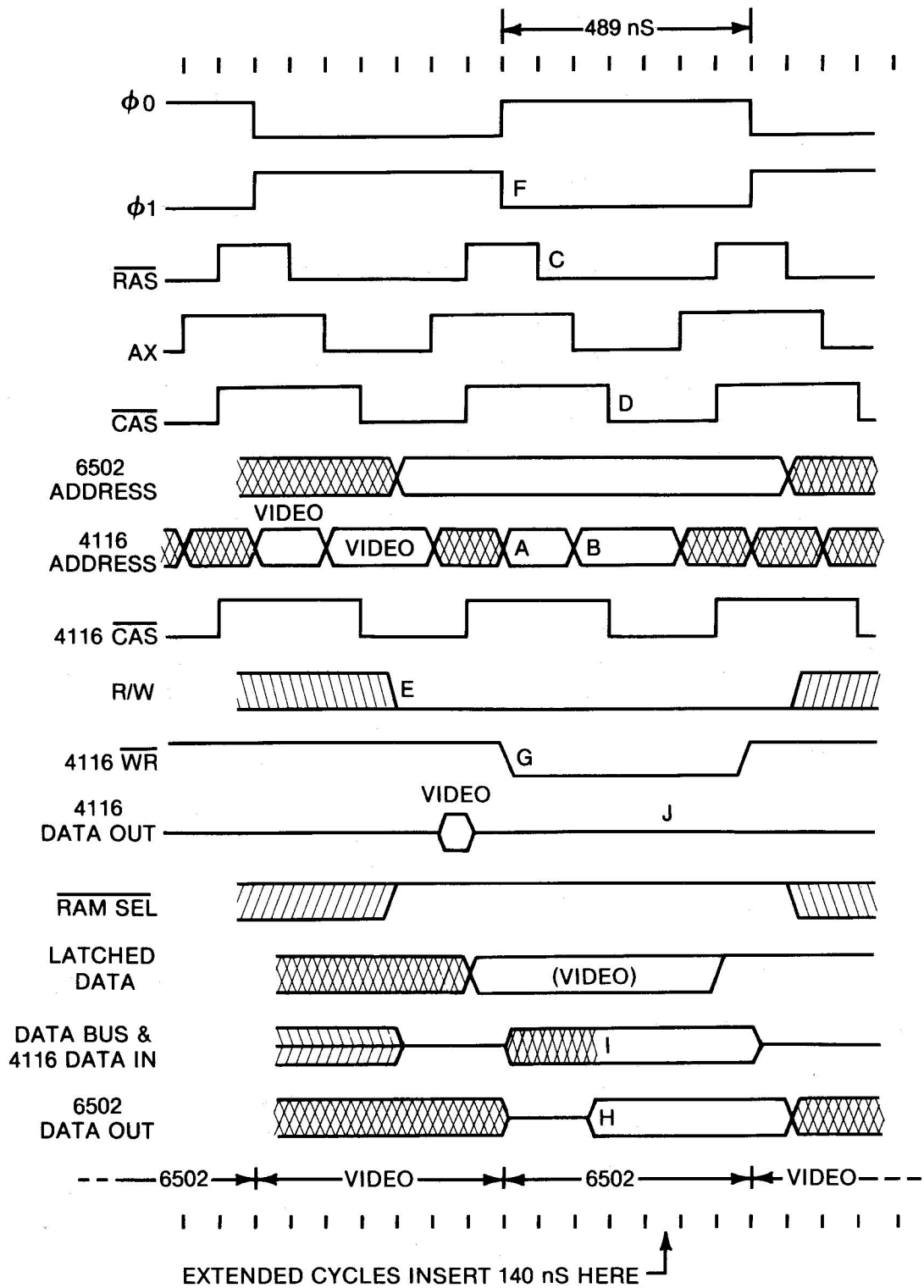


Fig. 5-5. RAM write cycle.

Address Multiplexing and Mapping

Fig. 5-6 is a block diagram of the hardware process that occurs when we write to a video screen location. Say we want to write the pixel that is "2 down and 3 over" (we are being very general here). We have looked in the *Apple II Reference Manual* and found the "address" of that pixel. What we have found is the location in RAM that corresponds to that pixel. The address from the *Apple II Reference Manual* is what the 6502 should put on the address bus to access that location. Now let's examine Fig. 5-6 to see what happens in the hardware. The 6502 puts the address on the address bus, puts the data (on, off, blue, green, etc.) on the data bus, and performs a RAM write cycle. On such a cycle, the address multiplexer selects the address bus.

How do we see on the crt what we have written to RAM? The video address generator outputs the video address of each pixel in sequence. When the pixel "2 down and 3 over" is addressed, this address is mapped to a RAM address by the mapper. It is then selected by the address multiplexer and sent to the RAM where it accesses the same location previously written. The data held in that location is read and sent to the video generator to create the desired pixel characteristics (on, off, blue, etc.). You can see that the mapper is the key to determining which RAM locations correspond to which screen locations.

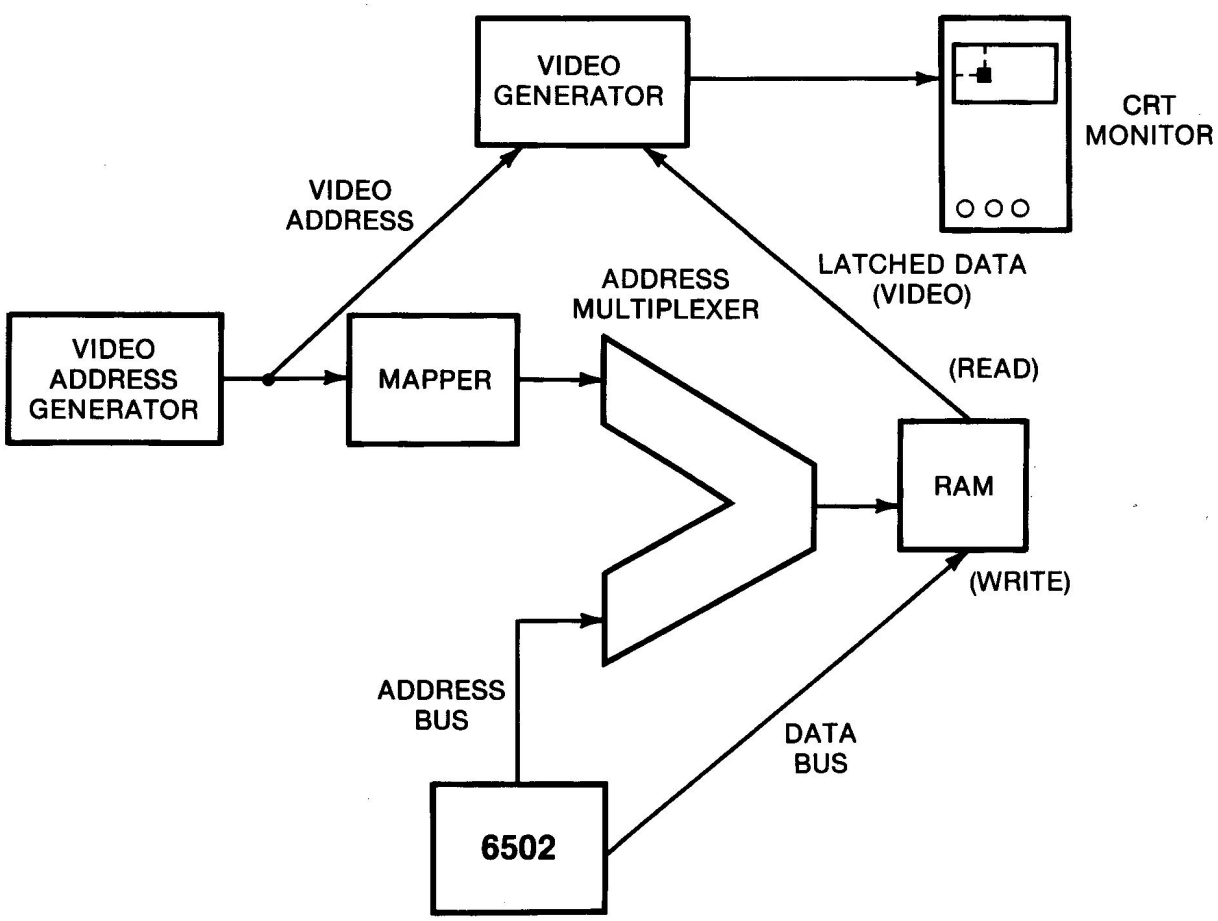


Fig. 5-6. Block diagram displaying a picture element.

Fig. 5-7* is an aid to analyzing the mapping circuitry. The video address bits (H0 to V4) appear on the left. They map through ICs C1, C12, and E14 to the RAM. The RAM address is shown as 14 row and column bits. The 6502 address bits are shown in the center of the figure. The 14 low order bits map to the RAM as shown. The two high order bits are decoded by ICs J1 and F2 to select one of three $\overline{\text{CAS}}$ lines. On video cycles, C1, J1, and F2 decode signals HIRES and PAGE 2 to select the correct $\overline{\text{CAS}}$ line. The four columns on the right of the figure represent four possible configurations of the mapper.

We now follow the diagram in detail, referring to Fig. 5-7* when needed.

6502 Access—E11, E12, E13, and C1 are four-to-one multiplexers (Figs. C-5* and C-6*). They multiplex 28 lines to 7 lines for the RAM addresses. At the beginning of a 6502 memory access, $\phi 0$ and AX are high (Fig. 5-3 point N). This causes the multiplexers to select inputs AD0, AD1, AD2, AD3, AD7, AD8, and AD12 as the 4116 row address—they are mapped as shown in Fig. 5-7*. For the column address, AX goes low causing the multiplexers to select inputs AD4, AD5, AD6, AD9, AD10, AD11, and AD13—they are mapped as shown in Fig. 5-7*. Meanwhile, $\phi 0$ is also high at J1-1 (Fig. C-5*). Thus, J1 selects AD14 to output on pin 9 and AD15 to output on pin 12. Addresses AD14 and AD15 then appear on the select inputs of F2. When $\overline{\text{CAS}}$ (F2-1) goes low, F2 is enabled and outputs a $\overline{\text{CAS}}$ on one of its three output pins (4, 5, or 6). The selected $\overline{\text{CAS}}$ then selects a unique 16K memory range. Consequently F2 output pins 10, 11, and 12 are gated in D2-8 to create $\overline{\text{RAM SEL}}$ as previously described. If the address is above 48K, then both AD14 and AD15 are high, and none of the *used* outputs of F2 are activated. This is correct since we do not want $\overline{\text{RAM SEL}}$ or any of the 4116 $\overline{\text{CAS}}$ lines activated when we are not accessing RAM.

Page 1 Text—When the Apple is displaying Page 1 Text, HIRES and PAGE 2 are both low. Since this is a video access, $\phi 0$ will also be low. $\phi 0$ low at J1-1 causes it to select ground (pin 14) to output on pin 12. Recall that on a 6502 access, AD15 is connected to pin 12. Thus, a ground maps to AD15 as shown in Fig. 5-7*. The IC J1 also selects HIRES • PAGE 2 (gate H1-8) to output on pin 9. Recall that on a 6502 access, AD14 is connected to pin 9. Thus, H1-8 maps to AD14 and is 0 as shown in Fig. 5-7* for Page 1 Text.

HIRES is low at C12-1 (Fig. C-5*) causing it to select HBL (pin 14) to output on pin 12. Also, C12 selects ground (pin 11) to output on pin 9. During the first part of the video cycle, AX is high (Fig. 5-3 point C). Thus, $\phi 0$ low and AX high at C1 pins 2 and 14 (Fig. C-5*) cause it to select HBL (via C12-12) to be 4116 row address 6. As shown in Fig. 5-7*, row address 6 maps to AD12. During the visible portion of a line, HBL is low, so a 0 is shown in Fig. 5-7* for AD12 during Page 1 Text. When AX goes low (Fig. 5-3 point F), C1 (Fig. C-5*) selects ground (via C12-9) to be 4116 column address 6. As shown in Fig. 5-7*, column address 6 maps to AD13.

HIRES low at C12-1 will cause it to select PAGE 2 (low at this time) to output on pin 7. Furthermore $\phi 0$ and AX low at E11 pins 14 and 2 (Fig. C-6*) cause it to select PAGE 2 (low) to output as 4116 column address 3. Fig. 5-7* shows that column address 3 maps to AD11. Also, C12 selects the high at C11-2 ($\overline{\text{PAGE 2}}$ in Fig. C-5*) to output on pin 4. Data selector E11 then selects this high to output as 4116 column address 0. Fig. 5-7* shows that column address 0 maps to AD10. Note that for Page 1 Text (Fig. 5-7*), AD10 is 1 and all *higher* bits are 0. This address is 1024 decimal which we know to be the start of Page 1 Text. The 10 LSBs give a range of 1K which we also know to be correct.

But what about those 10 LSBs; where do they come from? They are derived from

the video address. Signals H0, H1, H2, V0, V1, and V2 connect directly to multiplexers E11, E12, and E13 and map as shown in Fig. 5-7*. The five lines H3, H4, H5, V3, and V4 are reduced to four lines ($\sum 0$, $\sum 1$, $\sum 2$, and $\sum 3$) at the output of 4-bit adder E14 (Fig. C-6*). These four lines then connect to the multiplexers and map as shown in Fig. 5-7*. Their waveforms are shown in Fig. 5-8* along with signals previously derived. The address bit to which each signal maps is shown in parentheses next to the signal name. Ignore the numbers in brackets for now; they apply to HIRES mode only.

Let's examine how the $\sum$ signals are derived. A 74LS283 adder performs a binary addition of two 4-bit inputs and a single carry-in bit:

				C_{in}
	A3	A2	A1	A0
+	B3	B2	B1	B0
C_{out}	$\sum 3$	$\sum 2$	$\sum 1$	$\sum 0$

In the Apple this addition becomes:

				V3
	$\overline{H5}$	V3	H4	H3
+	V4	$\overline{H5}$	V4	1
	$\sum 3$	$\sum 2$	$\sum 1$	$\sum 0$

The carry-out is not used. Note that the symbol "+" used in this section represents *binary addition* and not logical OR.

Let's derive the signals for point A in Fig. 5-8*:

$$\begin{aligned} \sum 0 &= V3 + H3 + 1 \\ &= 0 + 1 + 1 \\ &= 0, \text{ carry } 1 \end{aligned}$$

We plot a low for $\sum 0$ at point A then continue:

$$\begin{aligned} \sum 1 &= H4 + V4 + \text{carry from } \sum 0 \\ &= 1 + 0 + 1 \\ &= 0, \text{ carry } 1 \end{aligned}$$

$$\begin{aligned} \sum 2 &= V3 + \overline{H5} + \text{carry from } \sum 1 \\ &= 0 + 1 + 1 \\ &= 0, \text{ carry } 1 \end{aligned}$$

$$\begin{aligned} \sum 3 &= \overline{H5} + V4 + \text{carry from } \sum 2 \\ &= 1 + 0 + 1 \\ &= 0, \text{ carry } 1 \end{aligned}$$

We end up plotting a low for all four $\sum$ signals at point A. This method can be used to determine the values of the $\sum$ signals at all points in Fig. 5-8*. Note how the waveforms change as a function of V3 and V4 (points B, C, D, and E).

We know from Chapters 3 and 4 that point A occurs at the upper left of the screen. We know this based on the value of the video address at point A. Let's find

the memory location which maps to this video address. Examining point A of Fig. 5-8*, we see that AD3 through AD9 are low. We have not shown H0, H1, and H2 in Fig. 5-8*, but let's assume they are low. They map to AD0, AD1, and AD2 (see Fig. 5-7*). We are still discussing Page 1 Text mode, so AD10 is high. This gives us a decimal address of 1024. If you check Figure 1 (page 16) of the *Apple II Reference Manual,* you will find that the character at the upper left indeed has an address of 1024. This method of adding up the address bits can be used to determine the address of every point in Fig. 5-8*. We have done this for several points.

Point F is at the upper right of the screen. There are 40 bytes or characters between points A and F. (In Fig. 5-8*, the addresses increment by eight since the three LSBs are not shown.) Note that the addresses between F and G (shown in parentheses) are outside the range used for Page 1 Text. This does not matter since they occur during blanking. The second horizontal scan line starts at G and addresses location 1024 again. In fact, the first eight scan lines all start by accessing location 1024.

Recall that a text character is displayed in a 7 by 8 dot cell. The eight vertical dots account for the eight separate accesses of the same location. (Chapter 8 will illustrate the video display in more detail.) Point H starts the ninth scan line and the second text line. Find its address (1152) in Figure 1 of the *Apple II Reference Manual.* Points I and J (1064 and 1104) occur one-third and two-thirds of the way down the screen. Point K is at the lower right of the screen and vertical blanking extends from points K to L. The next field starts at point L, which is at the upper left of the screen.

Page 2 Text—Page 2 Text differs from Page 1 Text only as described here. PAGE 2 is high at C12-5, so C12-7 is high (Fig. C-6). This maps to AD11. C12-2 and C12-4 are low, and this maps to AD10. We have shown AD11 = 1 and AD10 = 0 in the Page 2 Text column of Fig. 5-7*. This makes Page 2 Text range from 2K to 3K decimal.

Page 1 HIRES—Page 1 HIRES is similar to Page 1 Text; it differs as follows. HIRES is high at C12-1 causing it to select C11-2 to output on C12-9 (Fig. C-5*). PAGE 2 is low, so C12-9 is high. Signals $\phi 0$ and AX low at C1 pins 2 and 14 cause it to select C12-9 to output on C1-7. This high maps to AD13 as shown in Fig. 5-7*. HIRES high at C12-1 causes it to select VC to output on C12-12. Signals $\phi 0$ low and AX high at C1 cause it to select VC (via C12-12) to output on C1-7. Thus, VC maps to AD12. HIRES high at C12-1 causes it to select VB and VA to output on C12 pins 7 and 4. Thus, VB and VA map to AD11 and AD10.

Note from Fig. 5-7* that for Page 1 HIRES, AD13 = 1, AD14 = 0, and AD15 = 0. This address is 8192 decimal which we know to be the start of Page 1 HIRES. The 13 LSBs give a range of 8K which we also know to be correct. These 13 LSBs are all shown in Fig. 5-8* where we have used brackets to indicate the address bits to which VA, VB, and VC map.

Recall that point A in Fig. 5-8* is at the upper left of the screen. Let's find the memory location that maps to this point. At point A, AD3 through AD12 are all low. Signals H0, H1, and H2 (not shown) are also low and map to AD0, AD1, and AD2. From Fig. 5-7*, we see that AD13 = 1, AD14 = 0, and AD15 = 0 for Page 1 HIRES. This gives us a decimal address of 8192. If you check Figure 3 (page 21) of the *Apple II Reference Manual,* you will find that the byte at the upper left indeed has an address of 8192. This method of adding up the address bits has been used to find the decimal addresses of several points in Fig. 5-8*. They are listed along the bottom of the figure.

Point F is at the upper right of the screen and there are 40 bytes between points A and F. Each byte generates seven dots horizontally. The addresses increment by eight since the three LSBs are not shown. The addresses between F and G occur during blanking. The second horizontal scan line begins at G and accesses location 9216. Recall that in TEXT mode, each location was accessed eight times. In HIRES mode, each location is accessed once. Point H starts the ninth scan line. Find its address (8320) in Figure 3 of the *Apple II Reference Manual.* Point K is at the lower right of the screen and vertical blanking extends from points K to L. The next field starts at point L which is at the upper left of the screen.

Page 2 HIRES—Page 2 HIRES differs from Page 1 HIRES only as described here. The signals PAGE 2 and HIRES are both high at H1-9 and H1-10 (Fig. C-5*). Thus, J1-11 and J1-9 are high, and this high maps to AD14 on Fig. 5-7*. PAGE 2 is high at C11-1 causing C12-10 and C12-9 to be low. When $\phi 0$ and AX are low at C1-2 and C1-14, C12-9 is selected to output on C1-7. Thus, a low maps to AD13. We have shown AD14 = 1 and AD13 = 0 in the Page 2 HIRES column of Fig. 5-7*. This makes Page 2 HIRES range from 16K to 24K.

Refresh

Remember that memory refreshes can be obtained without additional circuitry only if two requirements are met: The 4116 row addresses switch through all 128 combinations in 2 milliseconds or less.

Let's see if we will get free refreshes. From Fig. 5-7* we find the seven signals that map to the row address as shown in Table 5-2.

Table 5-2. Refresh Address Mapping

RAM Address Bit	Signal
Row 2	H0
Row 5	H1
Row 1	H2
Row 4	$\Sigma 0$
Row 6	HBL (TEXT Mode) or VC (HIRES Mode)
Row 0	V0
Row 3	V1

The signals are listed in order of increasing period. By examining their waveforms (found in Figs. 3-4, 4-2, and 5-8)*, you can see that each signal in the list has a period at least twice that of the preceding signal. Thus, in one period of V1, the seven bits switch through all 128 combinations. From Fig. 4-1*, we can find that V1 has a period of 2038 microseconds. This is close enough to 2 milliseconds to meet the requirement. We have now proven that the video accesses indeed refresh the memory.

SUMMARY

Table 5-3 lists a summary of the signals presented in this chapter.

Table 5-3. A Summary of Signals Presented in Chapter 5

Signal	Description
AD0 – AD15	System Address Bus
AX	Address Multiplex (high for row, low for column)
$\overline{\text{CAS}}$	4116 Column Address Strobe
DI	4116 Data In
DO	4116 Data Out
D0 – D7	System Data Bus
DL0 – DL7	Latched Data
HIRES	Goes high for HIRES display
$\overline{\text{KBD}}$	Keyboard (low for keyboard access)
PAGE 2	Goes high for Page 2 Text or Graphics
$\overline{\text{PAGE 2}}$	Complement of PAGE 2
ϕ0	6502 Phase 0 Clock
RA0 –RA6	RAM Address
$\overline{\text{RAM SEL}}$	RAM Select (low for keyboard or RAM read)
$\overline{\text{RAS}}$	4116 Row Address Strobe
R/W	Read/Write on system bus
$\sum$0–$\sum$3	Mapping of H3, H4, H5, V3, and V4
$\overline{\text{WR}}$	4116 Write

CHAPTER 6

The 6502 and System Bus

The Apple II provides a complete microcomputer system bus with the added bonus of decoded address select lines. This efficient design technique reduces the address circuitry needed on peripheral cards. In this chapter we cover the 50 pin system bus and the 6502 microprocessor. As usual, we start with an overview before digging into the details.

Schematic Reference: Figs. C-5*, C-7*, C-9*, C-10*, C-11*, C-12* and C-13*.

OVERVIEW

The 6502

The 6502 is a single-voltage, 8-bit, NMOS (n channel metal oxide semiconductor) microprocessor with a 64K byte address range. The device used in the Apple II is specified for a clock frequency of 1 MHz. This results in a 1 microsecond clock period or *clock cycle*. Each read or write of a single memory location requires one of these clock cycles. In addition to memory operations, some clock cycles are used for processing that occurs inside the 6502. The clock cycles are strung together to generate *instructions*, a term that should be familiar to assembly language programmers. An instruction is the smallest part of the microprocessor's time that can be specified by a programmer. Normally, an instruction ranges in duration from 2 to 7 clock cycles. In the hardware realm we have an opportunity to adjust the duration of an instruction by inserting *wait-states* or altering the clock's period. We can also interrupt the microprocessor between instructions. More on these features later.

The signals in and out of the 6502 can be divided into three groups: address, data, and control. Fig. 6-1 shows the 6502 pin assignments.

Address—The address bus (A0–A15) is a group of 16 lines that presents a binary address to the system memory and peripherals. The 16 lines allow the 6502 to access (read or write) 65,536 unique locations. The address bus is *undirectional* with addresses coming *out* of the microprocessor.

Data—The data bus (D0–D7) is a group of eight lines over which data is transferred between the 6502 and the addressed memory or peripheral location. On a write cycle, data is transferred from the 6502 to the addressed location. On a read

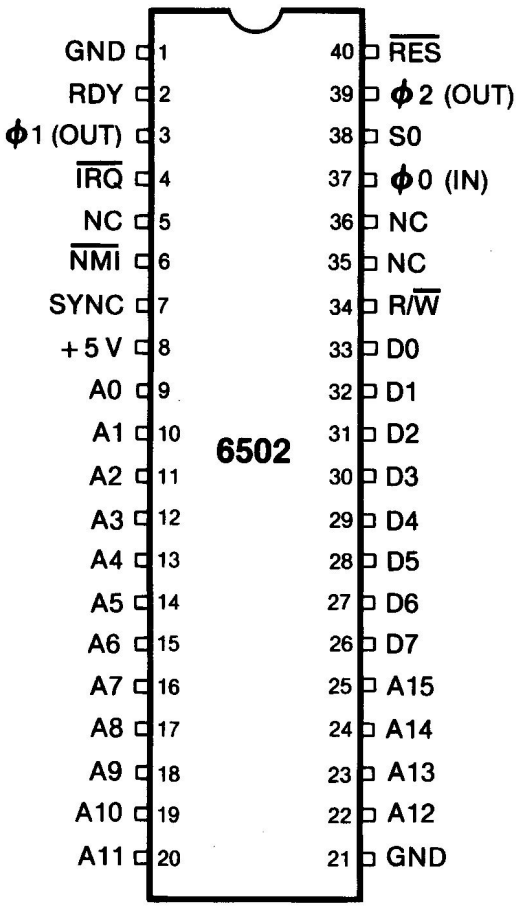


Fig. 6-1. 6502 Microprocessor.

cycle, data is transferred from the addressed location to the 6502. Thus, the data bus is *bidirectional*.

Control—The control group consists of the following signals:

Inputs

φ0—Clock In
RDY—Ready
NMI—Non-maskable Interrupt
IRQ—Interrupt Request
RES—Reset
SO—Set Overflow

Outputs

φ1—Clock Out
φ2—Clock Out
R/W—Read/Write
SYNC

Clocks—The timing of all 6502 data transfers is controlled by the clocks of the microprocessor. φ0 is a clock input on which you normally place a square wave. For a 6502 specified at 1 MHz, this would be a 1 MHz square wave (Fig. 6-2). From this input, the 6502 generates two clock outputs: φ1 and φ2. These clock outputs are

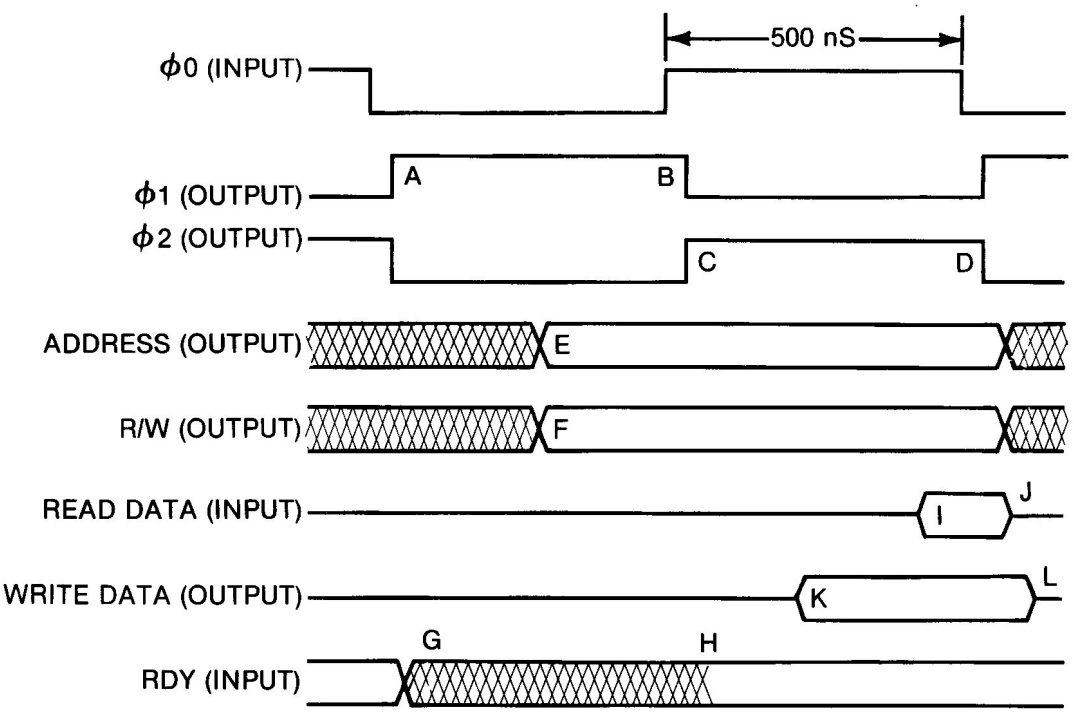


Fig. 6-2. Standard 6502 read/write timing.

approximate square waves of opposite phase. When we speak of an event occurring "during $\phi 1$," we mean during the time that $\phi 1$ is high (points A to B in Fig. 6-2). Likewise, "during $\phi 2$" refers to the time between points C and D. Note the slight delay between the input ($\phi 0$) and the outputs ($\phi 1$ and $\phi 2$). The 6502 $\phi 2$ output is not used in the Apple II—$\phi 0$ is used in its place. We will still refer to events occurring "during $\phi 2$."

Read/Write—Signal R/W controls the direction of the data bus. R/W is high for a read cycle and low for a write cycle. The R/W and the address bus stabilize during $\phi 1$ (points E and F in Fig. 6-2). The data transfer takes place during $\phi 2$. We will examine the exact sequences later.

Ready—Signal RDY is used to delay a 6502 read cycle momentarily so that slow memory or peripherals can have time to place their data on the data bus. Signal RDY is *high* to indicate that the memory is ready. Signal RDY is *low* to delay the 6502. The RDY signal must change only during $\phi 1$ or during the first 100 nS of $\phi 2$ (points G and H). See References 6.2, 6.9, 6.12, and 6.15 for timing details. The RDY signal is sampled during $\phi 2$, and if high, the cycle will complete. If RDY is low, the cycle will be extended to the next $\phi 2$ when RDY is again sampled. In this manner, the RDY line can insert an integer number of clock cycles or *wait-states*. The RDY line is ignored during write cycles. No circuitry on the mother board uses the RDY line, but it is available at the peripheral connectors. It is usually driven by open collector devices.

Interrupts ($\overline{IRQ}$ and $\overline{NMI}$)—Signals $\overline{IRQ}$ and $\overline{NMI}$ are inputs to the 6502. When activated (low), they request the 6502 to interrupt the current program and jump to another routine. After executing the new routine, the processor usually returns to the interrupted program. An application of interrupts would be a real time clock. For example, the clock could interrupt the processor every second to run a routine. This routine could do something like update a clock display on the screen.

The $\overline{\text{IRQ}}$ line is *maskable*. This means that the 6502 can be programmed to ignore the $\overline{\text{IRQ}}$ line by using the SEI instruction (Set Interrupt Disable Status). The SEI instruction sets the interrupt disable flag which is one bit in the internal status register of the 6502. The bit may be cleared by using the CLI instruction (Clear Interrupt Disable Bit). A situation that might call for disabling interrupts would be a time-critical process. For example, if you used a "software UART" routine to drive your printer, you would not want an interrupt to divert the 6502 in mid-character.

The $\overline{\text{IRQ}}$ input is *level-sensitive*. This means that whenever $\overline{\text{IRQ}}$ is low (and interrupts are enabled), the 6502 will be interrupted. Because of this, the 6502 automatically sets the disable interrupt flag when it recognizes an interrupt. It is up to the user to restore interrupts using the CLI or RTI (Return from Interrupt) instruction. However, this should not be done until the interrupting device has released the $\overline{\text{IRQ}}$ line. Otherwise the interrupt from that device would be recognized again.

As the name says, the $\overline{\text{NMI}}$ line is *non-maskable*; it cannot be disabled by the processor. An application for a non-maskable interrupt would be to run a *trap* routine. For example, the hardware may detect an error condition and activate the $\overline{\text{NMI}}$ line. The 6502 would always recognize $\overline{\text{NMI}}$ and run the trap routine. This routine could then read various registers and send the results to the printer for diagnostic purposes.

Note that the presence of both maskable and non-maskable interrupts in the 6502 provides a method of prioritizing interrupts. When $\overline{\text{NMI}}$ is recognized, the interrupt disable flag is set and the 6502 ignores $\overline{\text{IRQ}}$. When $\overline{\text{IRQ}}$ is recognized, the interrupt disable flag is also set. However, this has no effect on $\overline{\text{NMI}}$. If $\overline{\text{NMI}}$ is activated while the processor is servicing an interrupt from $\overline{\text{IRQ}}$, the first interrupt routine will be interrupted and the 6502 will service the request from $\overline{\text{NMI}}$. Afterwards, it will return to the first interrupt routine and then finally to the main program. Thus a non-maskable interrupt request can interrupt a maskable interrupt routine, but not vice versa. This gives non-maskable interrupts priority over the maskable type.

The $\overline{\text{NMI}}$ input is *edge-sensitive*. This means that it is recognized once for each negative transition. It must return high then low to be recognized again.

Interrupts $\overline{\text{IRQ}}$ and $\overline{\text{NMI}}$ are sampled during $\phi 2$. If either signal is low, the 6502 will finish the current instruction, then start the appropriate interrupt routine. An interrupt request will not be recognized if RDY is low. Interrupts $\overline{\text{IRQ}}$ and $\overline{\text{NMI}}$ are normally driven with open collector devices. Interrupts $\overline{\text{IRQ}}$ and $\overline{\text{NMI}}$ are not used on the mother board, but they are available at the peripheral I/O connectors.

Note that applications for non-maskable interrupts in the Apple II are limited. This is due to the software timing loops used in the floppy disk operating system. You can probably imagine the chaos that could occur if a non-maskable interrupt altered a critical timing loop while the disk was spinning.

Reset—The $\overline{\text{RES}}$ input is used to start the 6502 from a powered-down condition, or to set the program counter to a known location at any time. Simply stated, whenever $\overline{\text{RES}}$ is taken low, then returned high, the 6502 will start executing code from a known location, usually in ROM. In the Apple II, this location is in the monitor ROM. There are three sources of reset in the Apple as follows:

1. A power-up reset circuit (555 timer A13)
2. The keyboard reset button
3. The peripheral connectors

Sync—The 6502 SYNC output goes high during clock cycles that are op code fetch cycles—SYNC is not used in the Apple II.

Set Overflow—The 6502 SO (set overflow) input is used to set the overflow flag—SO is not used in the Apple II and is tied low.

6502 Read and Write Cycles

Before leaving Fig. 6-2, let's discuss the sequence of read and write cycles. We will assume that RDY is high.

On a read cycle, the address and R/W become stable no later than points E and F. The R/W signal will be high for a read cycle. The devices on the bus must now decode the address and the device that is selected must put its data on the bus no later then point I. The data must be held until point J. The data is strobed into the 6502 on $\phi 2$ falling at point D. This concludes the read cycle.

On a write cycle, the address and R/W have the same timing that they do on a read cycle. The R/W signal will be low, however, for a write cycle. The 6502 puts the write data on the bus no later than point K and holds it until point L. The accessed device must strobe in the data sometime between these two points. This concludes the write cycle. Additional 6502 cycle types will be described later in this chapter.

In this overview we have made only a brief exploration of the 6502. We must move ahead, however, to topics related more specifically to the Apple II. The references for this chapter contain more information on the 6502.

Address Decoding

We learned in Chapter 5 that addresses in the range 0 to 48K are decoded to access the RAM. Now we will discuss the remaining 16K of address space. The top 12K of this space is assigned to six 24-pin sockets. These sockets are arranged to accept type 9316B 2K-byte by 8-bit ROMs that are similar to (but not completely pin compatible with) 2716 EPROMs. The sockets are usually referenced by their address range rather than by their X-Y location on the mother board. In this book, we too will refer to the ROMs by their address. See Fig. C-10* for the physical locations of the ROM sockets. Fig. 6-3 shows the pin assignments for a 9316B ROM.

The F8 socket usually contains one of the monitor ROMs (original or autostart). In the original Apple II, F0, E8, and E0 contain the Integer BASIC ROMs. Sockets D8 and D0 are available for optional firmware such as the Programmer's Aid No. 1.

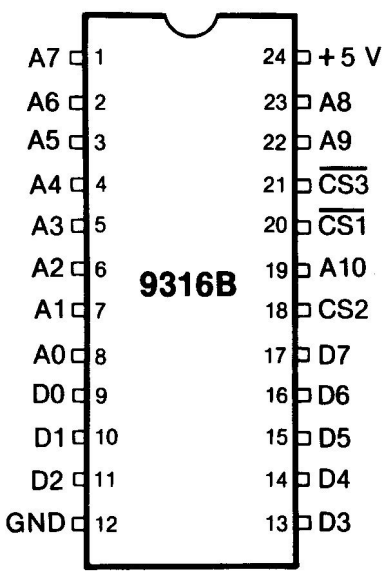


Fig. 6-3. 9316B ROM.

In the Apple II Plus, sockets F0, E8, E0, D8, and D0 contain the Applesoft floating point BASIC ROMs. These ROM variations are just optional ways of arranging the firmware. All the hardware really needs is a ROM in the F8 socket so that the 6502 will have a program to execute on power-up.

Let's look at the circuitry involved with the address decoding. One-of-eight decoder F12 divides the top 16K of address space into eight 2K blocks. The top six blocks ($D000–$FFFF) are assigned to the six ROM sockets. The next block down ($C800–CFFF) is assigned to the peripheral connectors (pin 20, $\overline{\text{I/O STB}}$). The final block ($C000–$C7FF) is further decoded by H12. Decoder H12 divides this 2K block into eight blocks of 256 bytes each. The top seven of these blocks are assigned to peripheral connectors 1 through 7 (pin 1, $\overline{\text{I/OSEL}}$). The remaining 256 byte block is further decoded by H2 and F13. Decoder H2 divides the upper half of the block ($C080–$C0FF) into eight blocks of 16 bytes each. These eight blocks are assigned to the peripheral connectors (pin 41, $\overline{\text{DEV SEL}}$). There now remain 128 bytes of address space ($C000–$C07F) unaccounted for. Decoder F13 and associated ICs decode this block for the on-board I/O. Address decoding will be covered in detail later in this chapter. Decoding of the 128 on-board I-O locations will be covered in Chapter 7.

You will note from the previous discussion that a peripheral connector has both 16-byte and 256-byte address blocks decoded for it. This decoding scheme uses only two ICs on the mother board and eliminates several potential ICs on each peripheral card. The ICs eliminated are ones that would be needed if the peripheral card had to decode its address range directly from the 16-bit address bus. This reduction of ICs is the efficient design which we mentioned in the introduction.

The bus signal $\overline{\text{INH}}$ (inhibit) is used by a peripheral to inhibit the on-board ROM. A typical application of this signal is the Apple ROM card. It uses $\overline{\text{INH}}$ to switch out the on-board ROM so that it can substitute its own ROM into the same memory space.

Direct Memory Access

Direct memory access (DMA) is a process whereby a peripheral gains direct access to the system memory, bypassing the microprocessor. A common reason for using DMA is speed; for example, when transferring data between memory and a hard disk. Without DMA, the processor must transfer data a byte at a time using a software routine. This routine would take many clock cycles per byte as it fetches op codes, computes addresses, etc. A peripheral with DMA capability can transfer one byte per clock cycle.

While a peripheral is using DMA, the processor cannot access memory. A method used to avoid conflict is to place the processor in a stopped or *hold* state. When an Apple II peripheral wishes to perform DMA, it takes the $\overline{\text{DMA}}$ line low. This stops the 6502 and forces the address bus and R/W to a high impedance. When the 6502 stops, its $\phi 1$ output is high and this forces the data bus to a high impedance. With R/W, address, and data buses at high impedance, the peripheral can put its own address and data on the buses. It can also control R/W. The peripheral now has access not only to memory, but also to the peripheral connectors and the on-board I/O. In other words, a peripheral with DMA capability can access any location that the 6502 can.

Daisy Chains

The $\overline{\text{DMA}}$ line is connected in common to all peripheral I/O connectors. What if more than one peripheral pulls low on $\overline{\text{DMA}}$ at the same time? An obvious conflict would occur. A DMA daisy chain is provided that can prevent a bus conflict if used with the proper circuitry. The daisy chain forms a series circuit running through the peripheral connectors. Circuitry is provided on each peripheral (with DMA capability) to break the daisy chain when DMA is requested. The nature of the chain is such that higher priority is given to peripherals in lower slot numbers. Should two or more peripherals simultaneously request DMA, only the highest priority peripheral will seize the bus. The exact operation will be described later in this chapter.

There is also a daisy chain for interrupts that can be used for the same purpose as the DMA daisy chain. Use of the interrupt daisy chain is optional, however, since a software polling routine could be used to detect and prioritize simultaneous interrupt requests.

Keyboard

The keyboard plugs into mother board location A7. When a key is pressed, a 7-bit code representing that key appears at A7 along with a strobe signal. The strobe is used to set a latch. When the 6502 addresses the keyboard, the seven data bits plus the latched strobe are driven onto the data bus. The 6502 then reads the data. If the program finds the strobe bit set, it uses the other seven bits to determine the key pressed. The program also resets the latch to wait for the next character. Fig. 6-4 shows the keyboard connector pin assignments at A7.

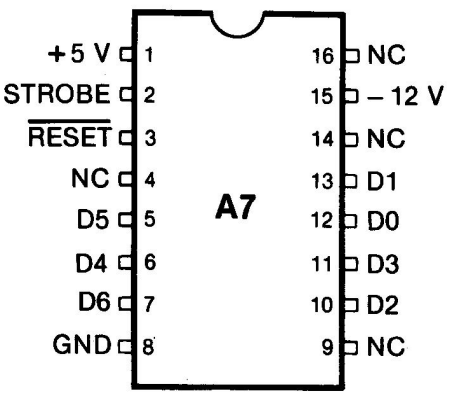


Fig. 6-4. Keyboard connector.

DETAILED CIRCUIT ANALYSIS

6502 Read from RAM

Fig. 6-5 depicts a cycle in which the 6502 reads from RAM. Note that we have shown the length of a half cycle as it actually is in the Apple—489 nS. This means the Apple's 6502 runs about 2% faster than the recommended 1 MHz maximum rate of a standard-speed part.

To start the cycle, the 6502 outputs the address during $\phi 1$. The address becomes stable by point A. The $\overline{\text{DMA}}$ is high due to pull-up resistor RA01-3 (Fig. C-9*). Thus, C11-12 is low to enable address bus drivers H3, H4, and H5. The address arrives on the system bus at point B (Fig. 6-5). Since this is a read cycle, 6502 R/W stabilizes high at point C. H5-5 is enabled, so BUS R/W goes high at point D. Since we are addressing RAM in the space 0–48K, one of the inputs of D2 (pins 9, 10, or 12 of Fig.

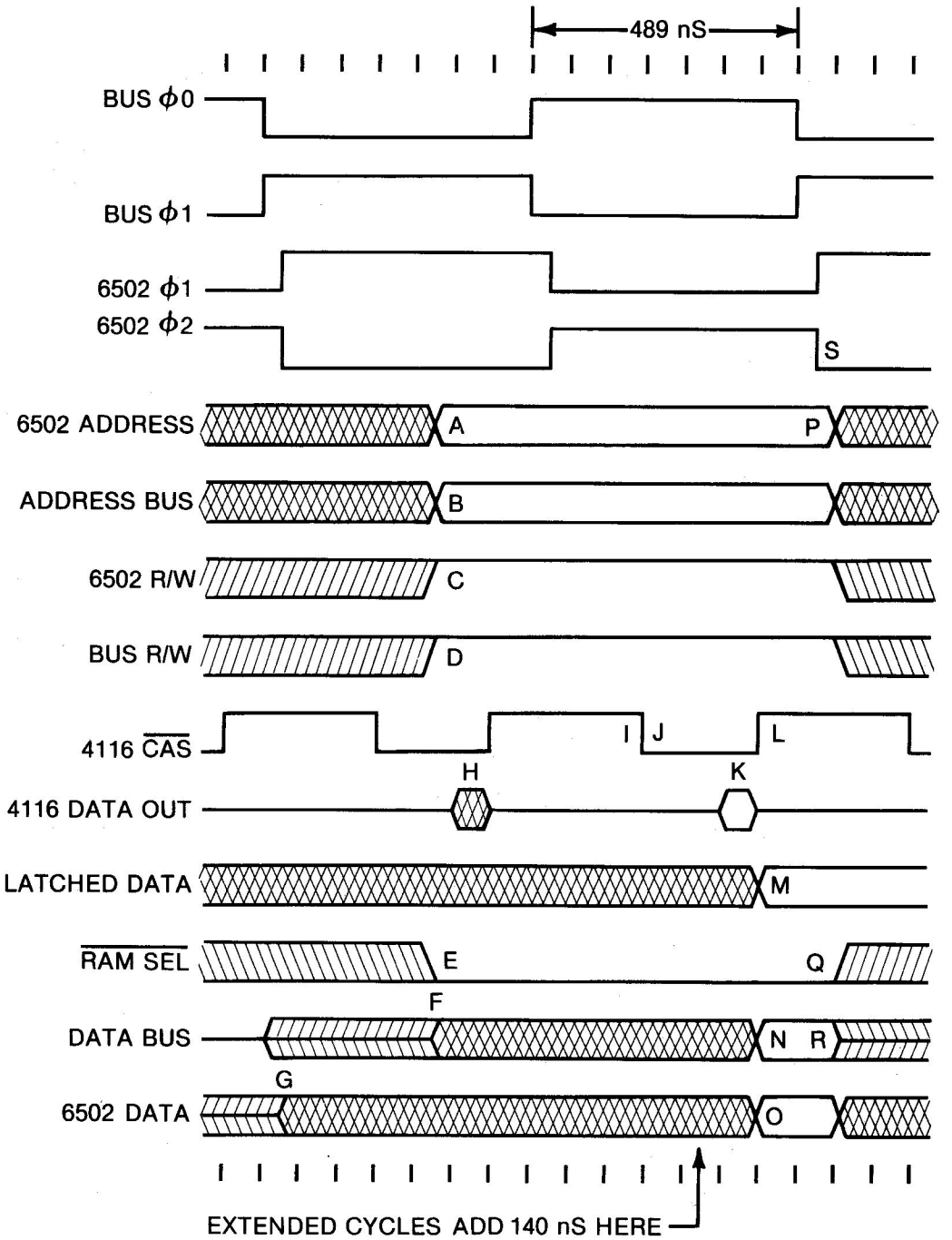


Fig. 6-5. 6502 read from RAM.

C-5*) will go low when the address stabilizes. Thus $\overline{\text{RAM SEL}}$ goes low by point E (Fig. 6-5). The low on $\overline{\text{RAM SEL}}$ will enable multiplexers B6 and B7 (Fig. C-13)*, and they drive garbage onto the data bus at point F (Fig. 6-5). Signal 6502 $\phi 1$ is high, so starting at point G, data transceiver H10 (Fig. C-9*) drives the data bus into the 6502. Meanwhile, a video RAM read cycle is occurring during $\phi 1$ as described in Chapter 5. The 4116s output the video data at point H (Fig. 6-5).

Next the address bus is multiplexed into row and column bits (also described in Chapter 5). The row and column addresses are strobed into the RAMs at points I and

J. The 4116 outputs go low impedance with valid data at point K and the data is held until $\overline{CAS}$ goes high at point L. Just before it disappears from the 4116 outputs, the data is latched in B5 and B8 (point M). The data is immediately driven onto the data bus by B6 and B7 (point N). The data then passes through H10 into the 6502 (point 0). The R/W and the address are held until point P. After that time, they may change. This change propagates to $\overline{RAM\ SEL}$ (point Q). When $\overline{RAM\ SEL}$ goes high at point Q, it disables multiplexers B6 and B7, and the data bus goes high impedance (point R). Just before the data is lost from the bus, it is strobed into the 6502 on 6502 $\phi 2$ falling (point S). This completes the cycle.

If this cycle occurs on one of the clock cycles that is extended, there will be an extra 140 nS inserted as shown in Fig. 6-5.

6502 Write to RAM

A cycle in which the 6502 writes to RAM is shown in Fig. 6-6. The R/W and the addresses stabilize during $\phi 1$ at the same time they do during a read cycle. This is a write cycle, however, so R/W will stabilize low (point A). The R/W is low at A2-4 (Fig. C-5*), so $\overline{RAM\ SEL}$ will be high (point B of Fig. 6-6). When $\overline{RAM\ SEL}$ goes high at point B, multiplexers B6 and B7 (Fig, C-13*) are disabled. This leaves no one driving the data bus, so it goes high impedance at point C (Fig. 6-6). When BUS $\phi 1$ falls at point D, 4116 $\overline{WR}$ goes low (point E) due to gate C14-11 (Fig. C-7*). This prepares the 4116s for an early-write cycle as described in Chapter 5. When 6502 $\phi 1$ falls at point F, it reverses the drive direction of data transceiver H10 via gate C14-8 (Fig. C-9*). This reversal lets the 6502 data bus go high impedance (point G of Fig. 6-6), and drives garbage onto the system data bus (point H). The 6502 makes its data bus low impedance and equal to the write data no later than point I. Data transceiver H10 drives the data onto the system data bus, point J. The data is strobed into the 4116s on $\overline{CAS}$ falling, point K. The R/W, the address, and the data are held until about point L. This concludes the write cycle.

As before, there is a 1 in 65 chance that this cycle could be extended 140 nS. Since this is the case on all 6502 clock cycles, we will not mention it again. We will note on each figure where the extension would be inserted.

6502 Read from ROM

The timing for a 6502 read from ROM cycle (Fig. 6-7) is similar to the timing for a RAM read cycle. The R/W and the addresses become stable at point A, and there is a video read during $\phi 1$ with output at point B. The $\overline{RAM\ SEL}$ goes high (point C) on BUS $\phi 0$ rising. This is due to the multiplexing action of $\phi 0$ at J1-1 (Fig. C-5*). While $\phi 0$ is low (prior to point C), J1 selects the video address which is always below 32K. Either F2-12 or F2-11 will be low. Thus, $\overline{RAM\ SEL}$ will be low. When $\phi 0$ goes high at point C, J1 selects the address bus which is above 48K since we are addressing ROM. None of F2-10, 11, or 12 are low and $\overline{RAM\ SEL}$ is high. The high on $\overline{RAM\ SEL}$ disables B6 and B7 (Fig. C-13*) so they release the data bus (point D of Fig. 6-7).

Right at this same time ($\phi 0$ rising), $\phi 1$ falls to partially enable decoder F12 via pins 4 and 5 (Fig. C-10*). Since we are addressing above 48K, both AD14 and AD15 are high. Thus, H1-6 is high to fully enable F12 via pin 6. Decoder F12 now decodes address lines AD11, AD12, and AD13 to select one of eight 2K blocks above 48K. Since we are addressing ROM, one of the six F12 outputs assigned to a ROM chip select will go low (point E of Fig. 6-7). The F12 outputs are assigned to chip select input pins 20 and 21. A third chip select pin on each ROM is tied in common to

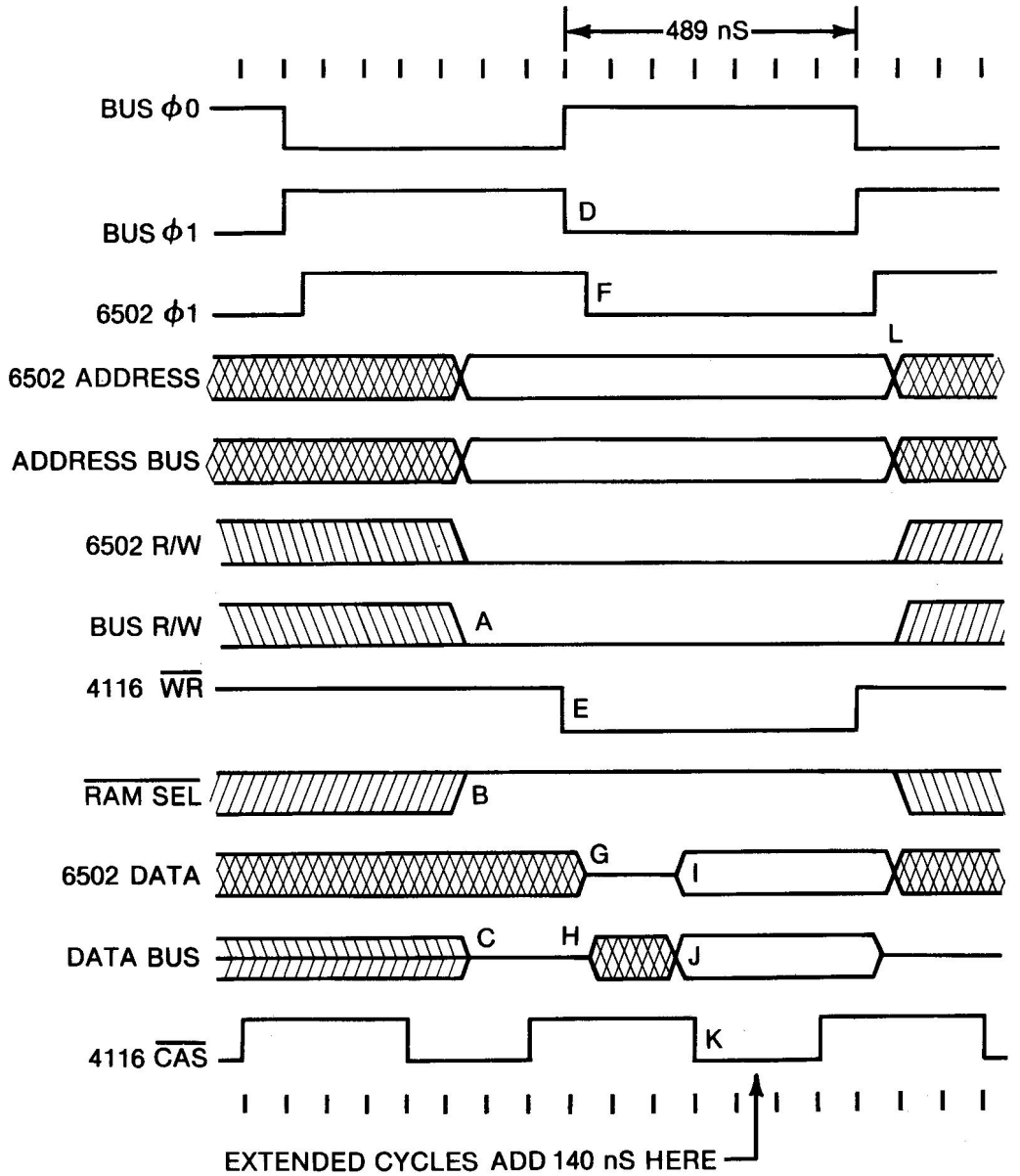


Fig. 6-6. 6502 write to RAM.

signal $\overline{\text{INH}}$. It is held high (ROMs enabled) by resistor RA01-7. Address lines AD0 through AD10 connect to the address inputs of the ROM. These 11 lines select one of the 2K locations in each ROM. After the IC's access delays from address and chip select, the ROM outputs will equal the data contained in the addressed location (point F of Fig. 6-7). This data is driven into the 6502 by H10 (point G). The ROM chip select goes high again on ϕ1 rising, point H. This disables the ROM which removes its data from the bus after a short delay, point I. Just before the data disappears from the 6502 data inputs (point J), it is strobed into the 6502 on 6502 ϕ2 falling (point K). This concludes the cycle.

Since the address for this cycle is above 48K, there is no active $\overline{\text{CAS}}$ at point L. It follows that the 4116 outputs remain high impedance at point M, and that the latched data at point N is garbage.

6502 Read from Peripheral

The cycle for reading from a peripheral is almost identical to the one for reading from ROM. We will reuse Fig. 6-7 for our explanation. Peripheral access cycles can be divided into four categories based on the address decoding:

1. I/O STROBE (Pin 20)—If the address is in the 2k range, $C800–$CFFF, F12-14 will go low during $\phi 2$ ($\phi 1$ low). Signal F12-14 is connected in common to $\overline{\text{I/O STB}}$ (pin 20) of all eight peripheral connectors. The typical application for this signal is to access a 2K ROM on a peripheral card. More than one card can have such a ROM, but there must be a method of enabling only one card at a time. This method is described on Page 84 of the *Apple II Reference Manual* under "Expansion ROM." The ROM decodes the 11 address LSBs and when accessed by $\overline{\text{I/O STB}}$ (point E of Fig. 6-7), puts the selected data on the data bus. It removes the data on $\overline{\text{I/O STB}}$ rising, point H.

2. I/O SELECT (Pin 1)—If the address is in the 2K range, $C000–$C7FF, F12-15 will go low during $\phi 2$. This partially enables decoder H12 (Fig. C-11*) via pins 4 and 5. Decoder H12 is fully enabled by the high on pin 6 from resistor RA01-8. Next, H12 decodes address bits AD8, AD9, and AD10 to select one of its eight outputs. Each output corresponds to a 256 byte range. The seven high order outputs connect to $\overline{\text{I/O SEL}}$ (pin 1) of peripheral connectors 1 through 7. Connector 0 has no connection to pin 1. The address range for each connector is $CN00–$CNFF where N is the slot number (1–7). The typical application for $\overline{\text{I/O SEL}}$ is to access a 256 byte ROM on a peripheral card. The ROM decodes the eight address LSBs and when accessed by $\overline{\text{I/O SEL}}$ (point E of Fig. 6-7), puts the selected data on the data bus. It removes the data on $\overline{\text{I/O SEL}}$ rising, point H.

3. DEVICE SELECT (Pin 41)—The low order output from H12 (pin 15) will go low during $\phi 2$ if the address is in the range $C000–$C0FF. This low will partially enable H2 via pin 4. Decoder H2 is also partially enabled during $\phi 2$ by the low ($\phi 1$) on pin 5. Finally, H2 will be fully enabled when AD7 (H2-6) is high. This corresponds to the range $C080–$C0FF. Next, IC H2 decodes address bits AD4, AD5, and AD6 to select one of its eight outputs. Each output corresponds to a 16 byte range and connects to $\overline{\text{DEV SEL}}$ (pin 41) of the peripheral connectors. The address range for each connector is $C0N0–$C0NF where N is the slot number plus 8 (that is N = $8 to $F). The typical application for $\overline{\text{DEV SEL}}$ is to access assorted latches, registers, UARTs, etc., on a peripheral card. The card decodes the four address LSBs and when accessed by $\overline{\text{DEV SEL}}$ (point E of Fig. 6-7), puts the selected data on the data bus. It removes the data on $\overline{\text{DEV SEL}}$ rising, point H.

4. Full Address Decode—A peripheral card does not have to use the decoded address on pins 1, 20, and 41. The card can perform its own decoding of the address bus (AD0–AD15). An application would be a card that is addressed in a range not available on pins 1, 20, or 41. For example, the Apple ROM card responds to the range $D000–$FFFF; not available as a predecoded select line. Note that the address bus becomes stable (point O) before the decoded select lines (point E), an advantage of this decoding method. However, the peripheral should still wait until $\phi 1$ is low to output to the data bus. This is to avoid a bus conflict since B6 and B7 drive the bus until point D.

A Caution for Designers—A few words about the read data hold time are in order. They apply to all four addressing methods discussed previously. The 6502 data sheet specifies that at the end of a read cycle, the data must remain stable on the 6502 data inputs for a 10 nS *hold time* after 6502 $\phi 2$ falling. Referring to Fig.

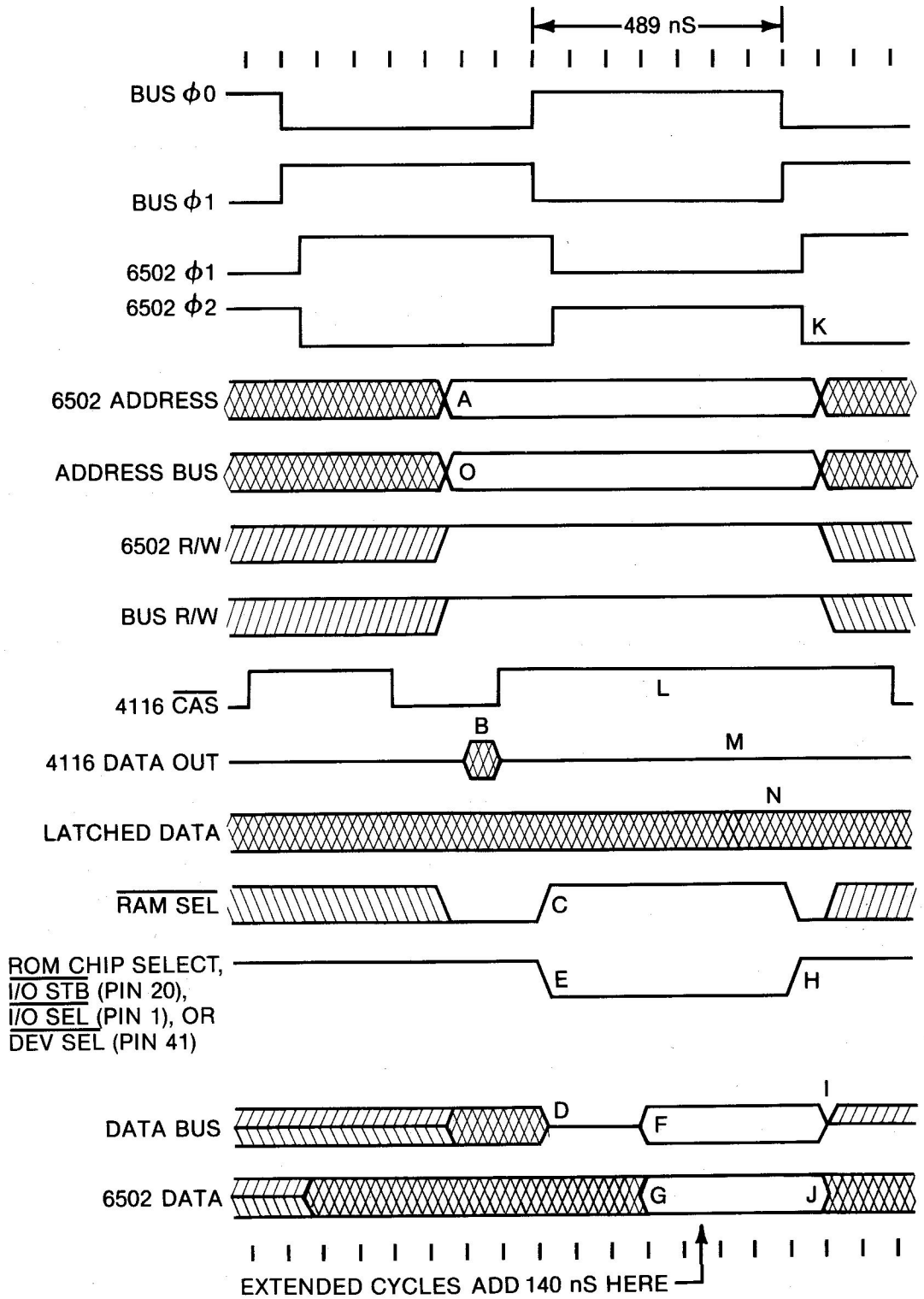


Fig. 6-7. 6502 read from ROM or peripheral.

6-7, we see that point J (6502 data no longer valid) must occur at least 10 nS after point K (6502 $\phi 2$ falling). Recall that 6502 $\phi 2$ is not used in the Apple II. The signal that is used in its place to enable data onto the bus is BUS $\phi 0$.

BUS $\phi 0$ leads 6502 $\phi 2$ by the propagation delays through B11-3 and the 6502. Thus, when BUS $\phi 0$ goes low to remove the data from the bus, it may remove the data *before* the 6502 hold time has been satisfied. This problem can be helped by using $\overline{\text{I/O STB}}$, $\overline{\text{I/O SEL}}$, $\overline{\text{DEV SEL}}$, or inverted $\phi 1$ instead of $\phi 0$ to enable the data onto the bus. Also, the use of a data bus buffer that is slow to disable will help. In summary, the designer of an Apple II peripheral card should make a careful timing analysis.

6502 Write to Peripheral

A cycle in which the 6502 writes to a peripheral (Fig. 6-8) is similar to a cycle in which it reads from a peripheral. On a write cycle, R/W will stabilize low (point A) and $\overline{\text{RAM SEL}}$ will stabilize high (point B). When $\overline{\text{RAM SEL}}$ goes high, it will disable B6 and B7 (Fig. C-13*), releasing the data bus at point C. On $\phi 1$ falling, $\overline{\text{I/O STB}}$, $\overline{\text{I/O SEL}}$, or $\overline{\text{DEV SEL}}$ will go low (point D). Also on $\phi 1$ falling, 4116 $\overline{\text{WR}}$ goes low (point E). This does nothing, however, since there is no 4116 $\overline{\text{CAS}}$ while this 4116 $\overline{\text{WR}}$ is active.

On 6502 $\phi 1$ falling, data transceiver H10 (Fig. C-9*) reverses its drive direction. This releases the data bus toward the 6502 (point F of Fig. 6-8) and drives garbage onto the system data bus (point G). The peripheral card is selected by $\overline{\text{I/O STB}}$, $\overline{\text{I/O SEL}}$, $\overline{\text{DEV SEL}}$, or a full decoding of the address bus. It then waits for the write data of the 6502 to become valid and driven onto the data bus, point H. The peripheral card should strobe the data into its memory or registers on the rising edge of $\overline{\text{I/O STB}}$, $\overline{\text{I/O SEL}}$, $\overline{\text{DEV SEL}}$ (point I), or $\phi 1$ (point J). This completes the write cycle.

6502 Read from Keyboard

In our discussion of the peripheral address decoding, we determined that H12-15 (Fig. C-11*) goes low during $\phi 2$ when the address is in the range \$C000–\$C0FF. This low partially enables decoder F13 via pin 5 (Fig. C-12*). Decoder F13 is further enabled via pin 6 when $\phi 0$ is high. Decoder F13 is finally fully enabled via pin 4 when AD7 is low. This corresponds to the 128 byte address range \$C000–\$C07F. Next, F13 decodes address bits AD4, AD5, and AD6 to select one of its eight outputs. Each output corresponds to a 16 byte range. The low order output (pin 15) is of interest to us here; it is the signal $\overline{\text{KBD}}$—$\overline{\text{KBD}}$ will go low during $\phi 2$ if the address is in the range \$C000–\$C00F. When programming, we simply use \$C000.

Fig. 6-9 shows a cycle during which the 6502 reads the keyboard. (It is similar to Fig. 6-7, a cycle during which the 6502 reads from ROM.) The 6502 puts out the keyboard address at point A and takes R/W high at point B. When R/W goes high, $\overline{\text{RAM SEL}}$ goes low (point C). When $\phi 0$ goes high at point D, multiplexer J1 switches (Fig. C-5*). Since we are addressing above 48K, D2 pins 9, 10, and 12 all go high. This would normally take $\overline{\text{RAM SEL}}$ high. We are addressing the keyboard, however, so on $\phi 0$ rising, $\overline{\text{KBD}}$ goes low (point E of Fig. 6-9). A low $\overline{\text{KBD}}$ signal at D2-13 makes D2-8 high and keeps $\overline{\text{RAM SEL}}$ low (point F). Thus, $\overline{\text{KBD}}$ low at multiplexers B6 and B7 (pin 1) causes them to select the keyboard connector A7 (Fig. C-13*). (Data bits 0–6 go to the connector; bit 7 is the latched keyboard strobe, B10-9.) With $\overline{\text{RAM SEL}}$ low at B6 and B7 pin 15, they drive the keyboard data and strobe onto the

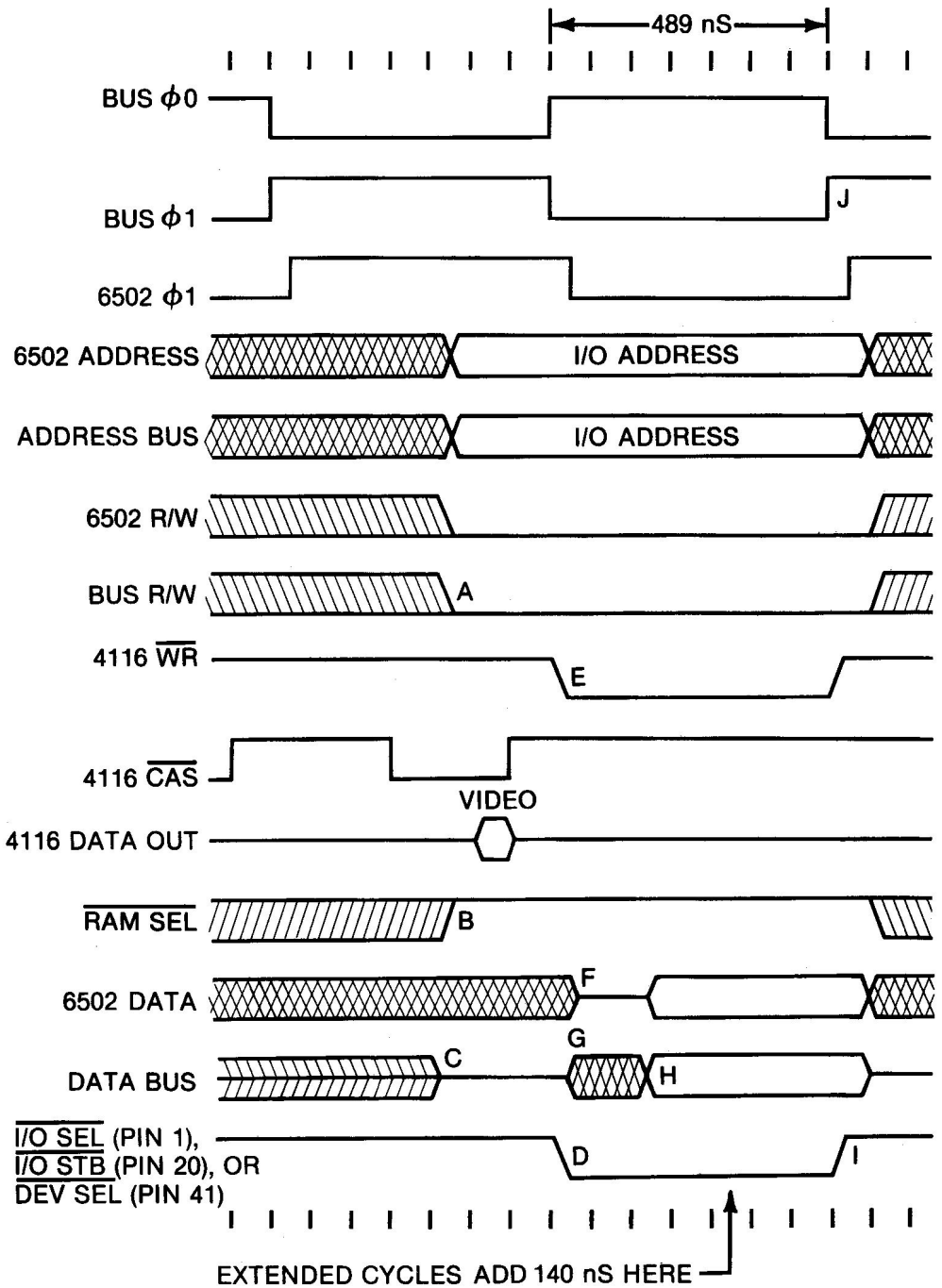


Fig. 6-8. 6502 write to peripheral.

data bus, point G. Data transceiver H10 drives the data into the 6502 which strobes it on 6502 ϕ2 falling, point H. This concludes the keyboard cycle.

When any character key is pressed on the keyboard, A7-2 goes high (Fig. C-13*). This sets flip-flop B10 and its pin 9 goes high. When a program scans or reads the keyboard, it checks data bit 7 (B10-9). A 1 (high) indicates that a key has been pressed and that data bits 0–6 contain the character. The program must then address location $C010. This takes $\overline{\text{CLR STB}}$ low and clears B10-9 via C11-4 and A12-1. If this is not done, B10-9 will still be high on the next scan, and the same character will be read again.

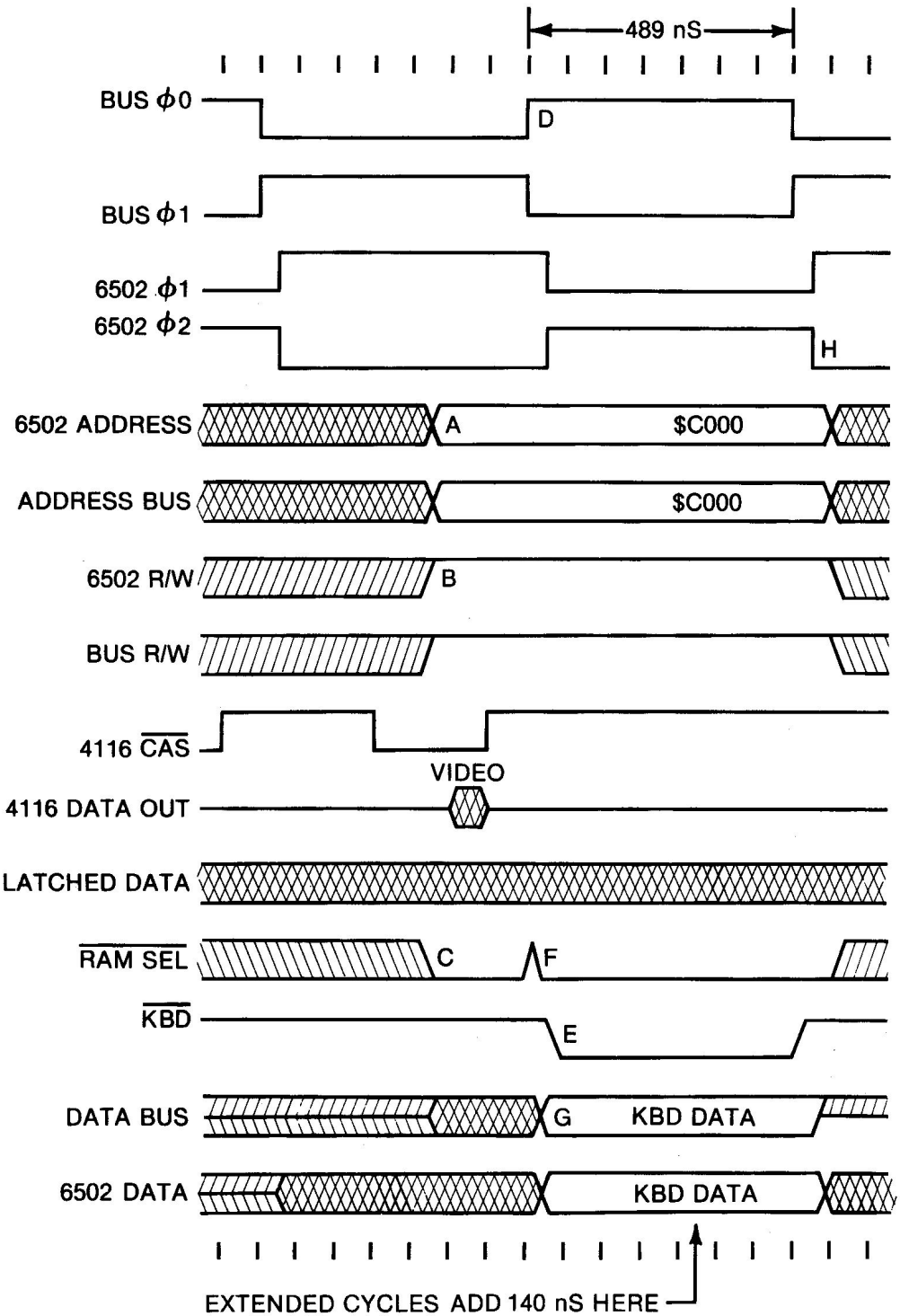


Fig. 6-9. 6502 read from keyboard.

USER 1

A signal named USER 1 is connected in common to pin 39 of all eight peripheral connectors (Fig. C-11*). USER 1 then connects to a jumper on the mother board. When this jumper is absent, USER 1 can be used as a signal line between two or more peripheral cards. Its application would be up to the designer of the peripherals.

When the jumper is installed, USER 1 connects to H12-6. This gives the peripherals control over the 2K address space $C000–$C7FF. This range consists of the on-board I/O and the memory blocks selected by $\overline{\text{I/O SEL}}$ and $\overline{\text{DEV SEL}}$. When USER 1 is high, this range is not affected. When USER 1 is low, the range $C000–$C7FF is inhibited.

The Ready Line

The RDY (Ready) line gives the peripheral cards the ability to momentarily delay the 6502 an integer number of clock cycles. An application would be the use of peripheral devices with long access times.

Fig. 6-10 depicts a read cycle that has been extended by one *wait state*. The basic cycle is similar to the read from peripheral shown in Fig. 6-7. The main difference occurs when the peripheral takes RDY low to indicate that it will not have data ready within the usual one cycle. This occurs during $\phi 1$ after the peripheral recognizes its address, between points A and B in Fig. 6-10. During $\phi 2$, the 6502 will sample RDY and see that it is low. As a result, the 6502 will not strobe in the data at the end of $\phi 2$. Instead, it maintains R/W and the address for another clock cycle. Our peripheral is ready now, so during $\phi 1$ of the second cycle (points C to D) it releases RDY. Resistor RA01-5 pulls RDY high (Fig. C-9*). The 6502 again samples RDY during $\phi 2$ and, seeing that it is high, completes the read cycle.

The peripheral card can start decoding the address as early as point E (Fig. 6-10). However, it should not drive the data bus unless $\overline{\text{I/O STB}}$, $\overline{\text{I/O SEL}}$, $\overline{\text{DEV SEL}}$, or $\phi 1$ is low. This is to avoid a bus conflict with multiplexers B6 and B7. Note that any number of wait states may be inserted to extend the cycle. With the appropriate circuitry, RDY can be used to single step the processor one clock cycle at a time.

DMA Daisy Chain

We introduced the concept of DMA in the overview. The DMA allows a peripheral card to gain direct access to the system bus and take over in place of the 6502. Only one card at a time can be allowed to do this. Fig. 6-11* shows the DMA daisy chain that allows access to the bus, one card at a time, on a priority basis. We have shown cards with DMA capability plugged into slots 0, 2, and 7. The card plugged into slot 1 does not have DMA capability. A line on this card simply connects DMA IN (pin 27) to DMA OUT (pin 24).

The daisy chain starts at slot 0 pin 27 and continues through the cards and the mother board to slot 7 pin 24. The DMA OUT from slot 7 (pin 24) goes nowhere and is a don't care. (The optional connection at solder pad "7," Fig. C-11*, is not associated with DMA.) The DMA IN at slot 0 also has no connection on the mother board. Although not shown in this example, slots 3 through 6 must contain cards so that the daisy chain will be continuous. You do not need eight cards to use DMA. If you have fewer cards, they must be arranged so that the chain is continuous. The chain need not start in slot 0 or end in slot 7. The cards with higher priority are placed toward slot 0.

Normally, the signal CHAIN on each card is high. A pull-up resistor is provided on each card so that DMA IN of the highest priority card will be high. As a result, the daisy chain is normally high all along its path.

For DMA operation, suppose the card in slot 2 needs to perform a DMA. The logic on the card checks DMA IN and finds it to be high. This "tells" the card it may proceed since no higher priority card needs DMA. The slot 2 card then takes CHAIN

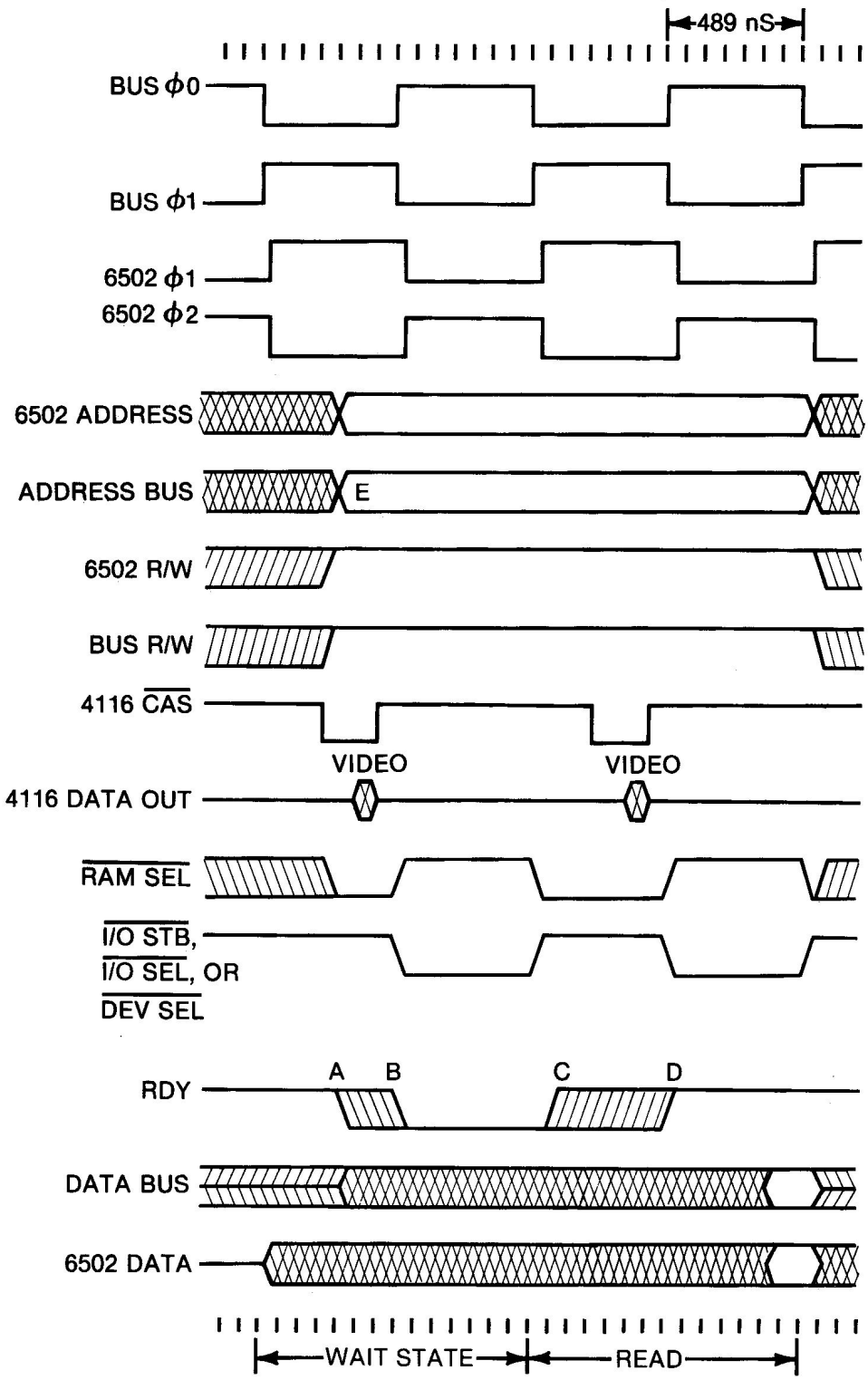


Fig. 6-10. 6502 read from peripheral extended by ready line.

low with the result that DMA OUT goes low. This low will propagate down the chain to lower priority cards, preventing them from starting DMA cycles. The slot 2 card also takes $\overline{\text{DMA}}$ low via an open collector gate. When $\overline{\text{DMA}}$ is low, it removes the 6502 from the bus—more on this later.

Now suppose a higher priority card (slot 0) needs to perform a DMA. Slot 0

DMA IN is high, so the card in slot 0 proceeds. It takes CHAIN and DMA OUT low. This low propagates to slot 2 where it prevents that card from performing a DMA.

The DMA operation must be *synchronous*. This means that peripheral cards must get their timing from the system clocks and start DMA cycles only during $\phi 1$. Otherwise, a peripheral could cut short a 6502 cycle and higher priority peripherals could cut short cycles of lower priority peripherals.

DMA Read from RAM

Fig. 6-12 shows a DMA read from RAM followed by a 6502 read from RAM. On $\phi 1$ rising (point A), the peripheral card checks DMA IN (point B). The DMA IN is high,

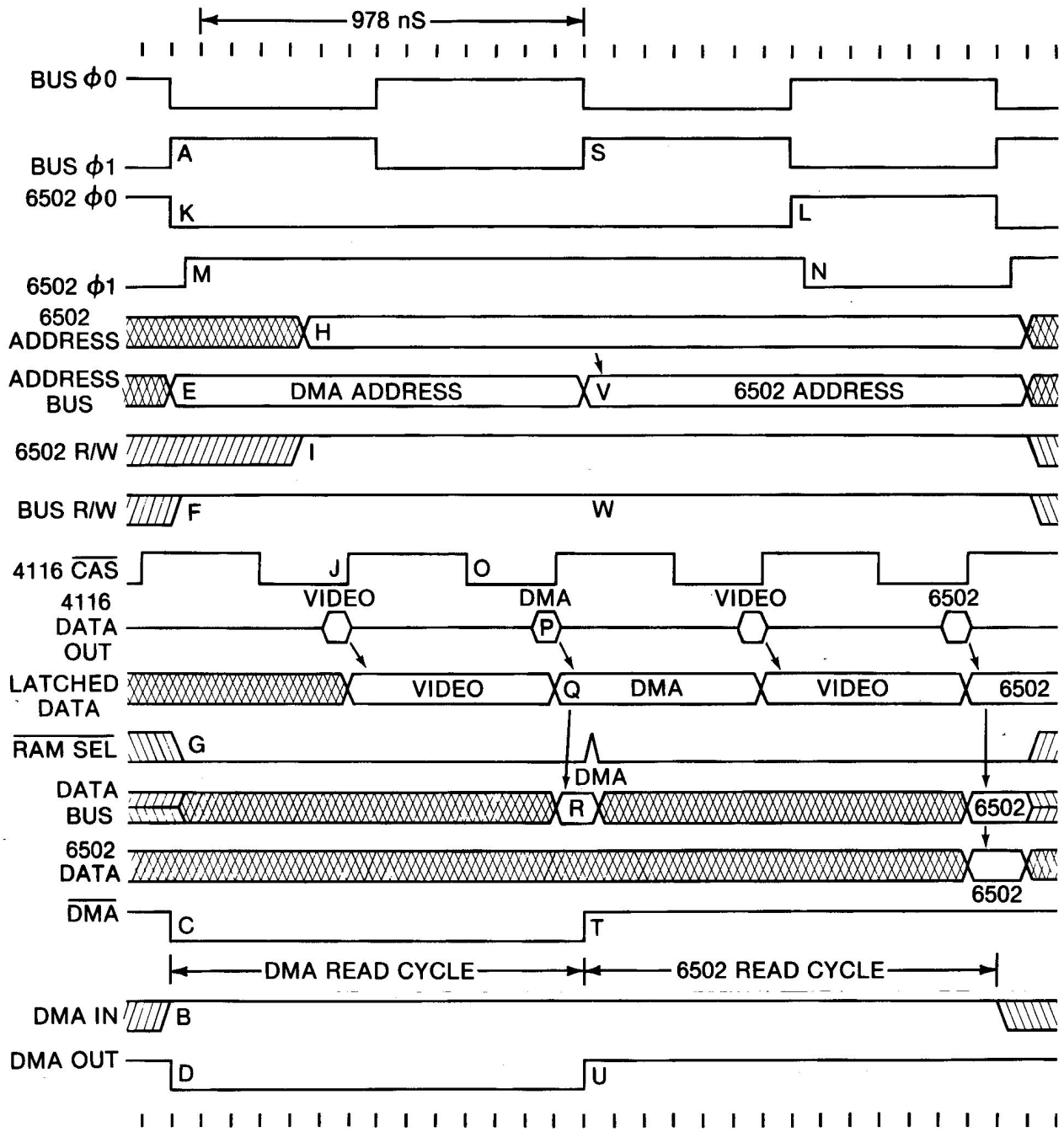


Fig. 6-12. DMA read from RAM.

so the card takes $\overline{\text{DMA}}$ low (point C) and DMA OUT low (point D). It also drives an address onto the bus (point E) and takes R/W high (point F). The $\overline{\text{DMA}}$ low at C11-13 takes C11-12 high to disable bus drivers H3, H4, and H5 (Fig. C-9). Thus, the 6502 address and R/W are removed from the bus so they will not conflict with the DMA address and R/W.

This is a RAM read cycle, so $\overline{\text{RAM SEL}}$ goes low as soon as the DMA address is decoded on the mother board (point G of Fig. 6-12). The 6502 address and R/W become valid at points H and I, but have nowhere to go yet. Note that the DMA cycle does not interfere with the video read from RAM at point J. The $\overline{\text{DMA}}$ low at B11-1 will keep 6502 $\phi 0$ low all the way from point K to point L. It will appear to the 6502 as if its clock input had stopped. The 6502 will hold its $\phi 1$ output high from point M to point N and will not enter $\phi 2$ until point N.

At point O, the 4116 $\overline{\text{CAS}}$ goes low for the memory bank selected by the DMA address. The data becomes valid at point P and is latched at point Q. It is driven onto the data bus by B6 and B7 (Fig. C-13*) at point R. The peripheral card should strobe in the data on $\phi 1$ rising at point S. Also on $\phi 1$ rising, the card returns $\overline{\text{DMA}}$ and DMA OUT high (points T and U).

With $\overline{\text{DMA}}$ now high, H3, H4, and H5 (Fig. C-9*) are enabled and drive the 6502 address and R/W onto the bus (points V and W of Fig. 6-12). Gate B11-3 now lets the 6502 clock continue and the 6502 enters $\phi 2$ at point N. The 6502 read from RAM continues as previously described.

DMA Write to RAM

Fig. 6-13 shows a DMA write to RAM cycle followed by a 6502 read from RAM. The DMA write operation is very similar to the DMA read cycle just described. Since this will be a write cycle, the first difference occurs when the peripheral card takes bus R/W low (point A). Bus R/W will cause $\overline{\text{RAM SEL}}$ to be high, point B. The card also drives the data to be written onto the data bus, point C. Since bus R/W is low, 4116 $\overline{\text{WR}}$ will go low at point D. The 4116 $\overline{\text{CAS}}$ goes low at point E to strobe the DMA data into RAM. Note that 6502 $\phi 1$ is high from point F to point G. As a result, C14-8 (Fig. C-9*) is high to keep data transceiver H10 from driving the data bus and conflicting with the DMA write data.

On $\phi 1$ rising (point H), the card releases $\overline{\text{DMA}}$, bus R/W, the address bus, and the data bus. The 6502 read from RAM cycle that follows has been previously described.

Note that Figs. 6-12 and 6-13 show simple examples of DMA cycles. Variations and refinements are possible. Also note that peripheral cards can have direct access to much more than memory. We showed DMA cycles that read and write the RAM. A card can also access ROM, on-board I/O, and other peripherals. It can perform reads or writes as appropriate. We have shown 6502 read from RAM cycles following the DMA cycles in Figs. 6-12 and 6-13. Of course, any cycle type could have followed, including another DMA from the same or a different card.

Recall that part of the DMA process involves stopping the $\phi 0$ clock input of the 6502. The 6502 is a dynamic device. If its clock input is stopped for too long, it will lose the data in its internal registers. Thus, a peripheral DMA device cannot hold the $\overline{\text{DMA}}$ line low continuously. The $\overline{\text{DMA}}$ line must be released at 10 μS minimum intervals so that the 6502 can refresh itself. Note that 10 μS is the *most restrictive* value obtained from the 6502 data sheets (References 6.9, 6.12, and 6.15 in Appendix D).

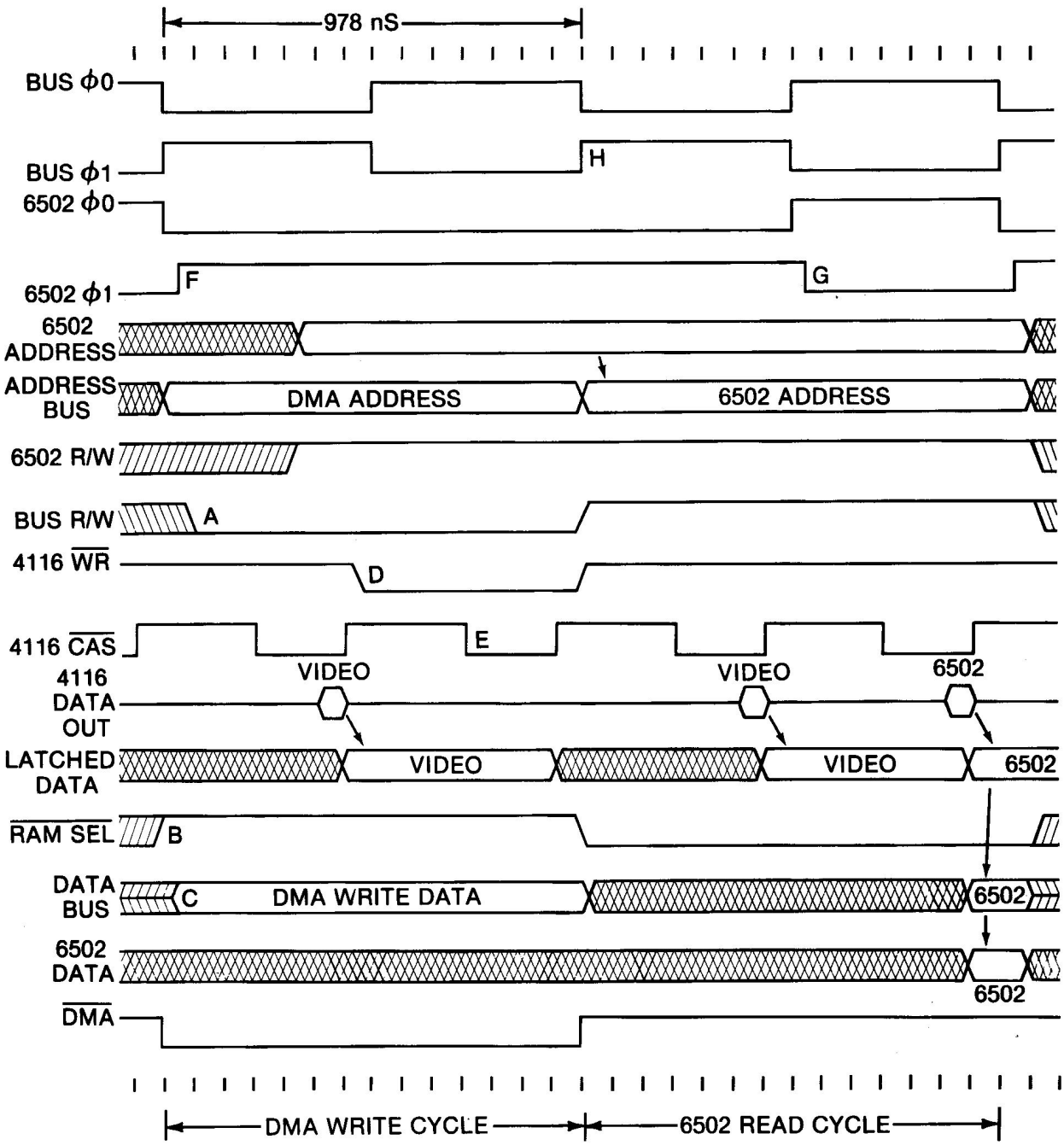


Fig. 6-13. DMA write to RAM.

Interrupts

Interrupt Daisy Chain—Interrupts of the 6502 microprocessor were discussed in the overview of this chapter. There we mentioned that an interrupt daisy chain was available that could be used to prioritize interrupt requests. This interrupt daisy chain operates just like the DMA daisy chain shown in Fig. 6-11*. Just substitute INT IN (pin 28) and INT OUT (pin 23) for DMA IN and DMA OUT. Also substitute $\overline{IRQ}$ or $\overline{NMI}$ for $\overline{DMA}$. (Note: The optional connection at solder pad "8," Fig. C-11*, is not associated with interrupts.)

Interrupt Sequence—Fig. 6-14* shows a typical interrupt sequence. Let's say that at about point A a peripheral card needs to interrupt the processor. The card checks INT IN (point B) and finding it high, proceeds to take $\overline{\text{IRQ}}$ or $\overline{\text{NMI}}$ low (point A). It also takes INT OUT low (point C).

Note that the interrupt request ($\overline{\text{IRQ}}$ or $\overline{\text{NMI}}$) may be *asynchronous*. This means it may occur at any time relative to the system clocks. It is sampled during $\phi 2$, and if it is found to be low, the 6502 will start its interrupt routine on $\phi 1$ following the current instruction. Note that the processor finishes the current instruction and not just the current clock cycle.

In Fig. 6-14, we have shown $\overline{\text{IRQ}}$ or $\overline{\text{NMI}}$ going low during $\phi 1$ of the last clock cycle of an instruction. This means the next clock cycle *could* be the start of the interrupt routine. It *is*, if the interrupt request is on $\overline{\text{NMI}}$. If the interrupt request is on $\overline{\text{IRQ}}$, the 6502 first checks the status of the interrupt mask bit. In our example the mask bit is off (low, point D), so the next clock cycle will be the start of the interrupt routine. One of the first things the interrupt routine does is set the interrupt mask bit (high, point E). The 6502 then proceeds with the interrupt routine. At the end of the routine, commands are sent to the interrupting card to let it return $\overline{\text{IRQ}}$ or $\overline{\text{NMI}}$ high (point F). The card also returns INT OUT high, point G. Then the 6502 executes the RTI (Return from Interrupt) or CLI (Clear Interrupt Disable Bit) instruction, resetting the interrupt mask bit (point H). This concludes the sequence and the 6502 is now ready to receive another interrupt request.

Note that the sequence in Fig. 6-14 is merely an example and that variations are possible. For example, it may be desirable for the peripheral to release $\overline{\text{IRQ}}$/$\overline{\text{NMI}}$, and for the 6502 to execute a CLI instruction *early* in the routine. This would then allow other peripherals to interrupt the first interrupt routine. See References 6.3, 6.4, 6.11, and 6.17 for more information. Also note that use of the interrupt daisy chain is optional.

Reset

Fig. 6-15 shows the Apple II power-up reset sequence. At point A the power is off and V_{CC} (the +5 V supply) is at 0. At point B the power is on and V_{CC} is near +5 volts. This starts the logic and clocks operating, but they are not yet stable (point C). As power is applied, the $\overline{\text{RESET}}$ line starts to pull high via resistor RA01-6 (point D). As soon as the power is on, 555 timer A13 goes into operation (Fig. C-13*). Initially capacitor C4 is discharged. This low at A13 pins 2 and 6 cause it to trigger and take its output (pin 3) high. This provides base drive to Q5 through R14. Q5 turns on and pulls the $\overline{\text{RESET}}$ line low (point E in Fig. 6-15). Now C4 begins to charge through R26. After a time determined by the formula T = 1.1 RC, A13 will turn its output off (low). This releases the $\overline{\text{RESET}}$ line (point F). $\overline{\text{RESET}}$ is low for about 240 mS.

The 6502 reset sequence actually starts on $\overline{\text{RESET}}$ rising at point F. The 6502 waits six clock cycles, then loads the program counter with the vector stored at $FFFC and $FFFD (point G). The processor then fetches the first op code (point H) and is off and running.

Note that the 240 mS power-up reset is available on the bus to reset circuitry on the peripherals. Also, a reset signal could originate on a peripheral. When the reset button on the keyboard is pressed, A7-3 (Fig. C-13*) goes low sending a reset to the 6502 and the peripherals. (There is a switch option on newer keyboards that can require both CONTROL and RESET to be pressed to take A7-3 low.) The power-up reset from A13 also clears (initializes) the keyboard strobe via A12-1.

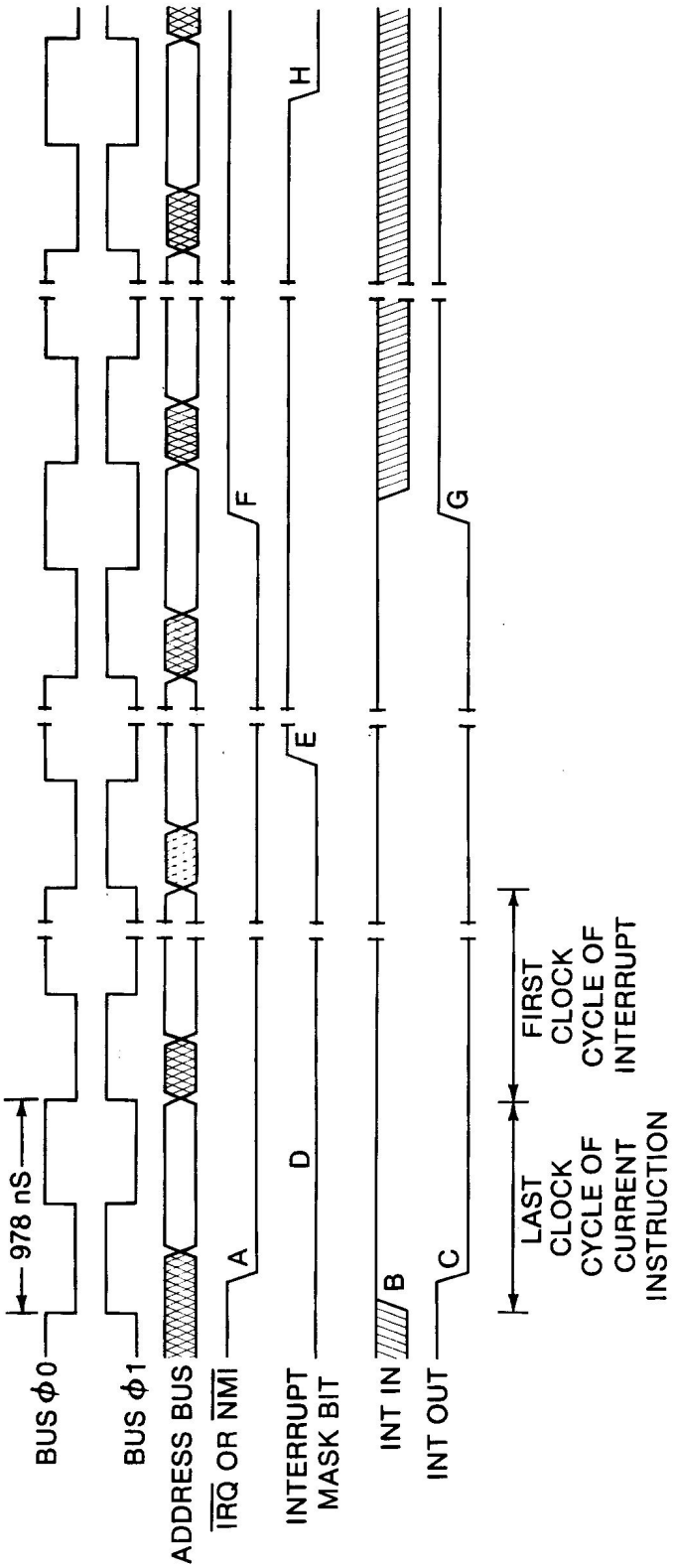


Fig. 6-14. 6502 interrupts.

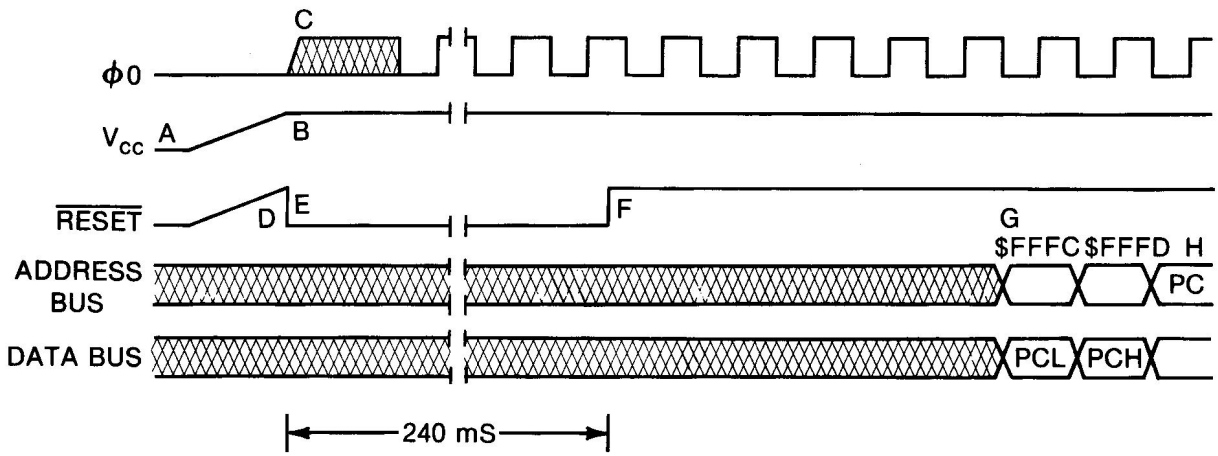


Fig. 6-15. 6502 reset sequence.

6502 'Scope Loop

In this chapter we have presented an assortment of processor cycles: RAM read, RAM write, ROM read, keyboard read, etc. For each one we have examined a timing diagram that shows just that one cycle. The waveforms look nice when printed on paper, but what do they look like on an oscilloscope?

Viewing microprocessor cycles on an oscilloscope is a bit tricky. It is not as simple as looking at a clock or a video waveform. Clocks, for example, are repetitive and follow a sequence that is defined by the way the hardware is wired. The signals from a microprocessor, on the other hand, are determined by the program that is running.

If we want to examine the timing of a specific 6502 cycle, we can use a short program called a *'scope loop*. In the program we put instructions that contain the cycle types we want. For example, if we wanted to examine the timing of a RAM read followed by a peripheral write, we could use an STA-absolute instruction. This instruction is four clocks long and consists of three reads followed by a write. Since we specified a read from RAM, our 'scope loop would have to execute in RAM. Since we specified a write to a peripheral, the address in the instruction would have to be in the peripheral's address space. The 'scope loop is made short so the waveforms will have a high repetition rate and be easily seen on an oscilloscope.

The program below is a short loop that can be run on the Apple II. It allows us to view several types of 6502 cycles as listed in the comments.

```
                        ;SCOPE LOOP.                    CLOCK
                ORG $800;                               CYCLES:
0800 AD30C0 LOOP LDA $C030 ;TOGGLE SPEAKER              4
0803 AD00C0      LDA $C000 ;READ KEYBOARD               4
0806 AD00F8      LDA $F800 ;READ PROM                   4
0809 8D0009      STA $900  ;WRITE TO RAM                4
0812 4C0008      JMP LOOP  ;NOTE: ANY OP CODE FETCH     3
                           ;IS A READ FROM RAM          ---
                           ;TOTAL =                     19
```

The results of the program are shown in Fig. 6-16*. The first instruction (LDA $C030) tells the processor to load its accumulator with the data found at location

$C030. Location $C030 is the address assigned to the on-board speaker. Whenever we access that location, the speaker toggles. We do not care that the speaker does not return any data; we are using the LDA instruction merely as a means to generate the address.

This instruction consists of four clock cycles. The first three cycles read the op code and its operand from consecutive RAM locations. These locations ($800, $801, and $802) are shown in Fig. 6-16* on the address bus. At location $800, the op code ($AD) is read from RAM as shown on the 4116 DATA OUT trace. This data is then latched, put on the data bus, and finally put on the 6502 data lines. At location $801, the operand low order byte ($30) is read. The high order byte ($C0) is read from location $802.

On the fourth cycle, the processor performs the operation specified by the op code. It first outputs $C030 on the address bus (point A). This is decoded on the mother board and found to be above 48K. Thus, there is no 4116 $\overline{\text{CAS}}$, and $\overline{\text{RAM SEL}}$ goes high during $\phi 2$ (point B). The address is further decoded by the string of decoders F12, H12, and F13 (see Figs. C-10*, 11*, and 12*). The specific address $C030 causes F13-12 to go low during $\phi 2$ (point C of Fig. 6-16*). The rising edge of F13-12 causes flip-flop J13-5 to toggle and create a sound in the speaker (point D). More on this in Chapter 7. Notice that the speaker does not respond with any data, so the data bus goes high impedance during $\phi 2$ (point E). The resulting garbage on the data inputs of the 6502 is strobed on $\phi 2$ falling at the end of the cycle. Remember that we do not care about this data anyway.

You will notice the video cycles (indicated by a "V") interleaved with the 6502 cycles throughout Fig. 6-16*.

If you were to actually enter the 'scope loop program (or any other program) into the Apple and view the processor signals on an oscilloscope, you would see the effect of the extended clock cycles. This is a messy display that could cause confusion. To eliminate the extended cycles, lift IC D2 pin 6 (Fig. C-2*). Everything except color displays should continue to operate properly, and you will get cleaner oscilloscope displays.

If you wish to view one complete period of the loop, you need to trigger the oscilloscope on a signal that occurs once each period. You could use F13-12 (Fig. C-12*) in our example.

Let's return to Fig. 6-16* and discuss the remaining instructions. The second instruction (LDA $C000) is similar to the first. Notice that the RAM locations for the op code and its operand start at $803 which follows the last address used by the previous op code. In the fourth cycle of this instruction we address the keyboard ($C000) which *does* respond with data (point F).

The third instruction (LDA $F800) is similar to the first two. Here we load the first byte from the monitor ROM ($4A) into the accumulator.

The fourth instruction (STA $900) tells the processor to store the contents of the accumulator at location $900. The first three clock cycles of this instruction fetch the op code ($8D) and its operand ($900) from RAM locations $809, $810, and $811. On the fourth cycle, the 6502 takes R/W low (point G). When R/W goes low, $\overline{\text{RAM SEL}}$ goes high. Since R/W is low, 4116 $\overline{\text{WR}}$ goes low during $\phi 2$ (point H). Later during $\phi 2$, the 6502 outputs the contents of its accumulator ($4A obtained in the previous instruction). The data is strobed into the RAM on 4116 $\overline{\text{CAS}}$ falling during $\phi 2$ (point I).

The fifth and last instruction is JMP $800. This tells the processor to jump to location $800 and execute the op code found there. Since $800 is the beginning of the program, the loop repeats.

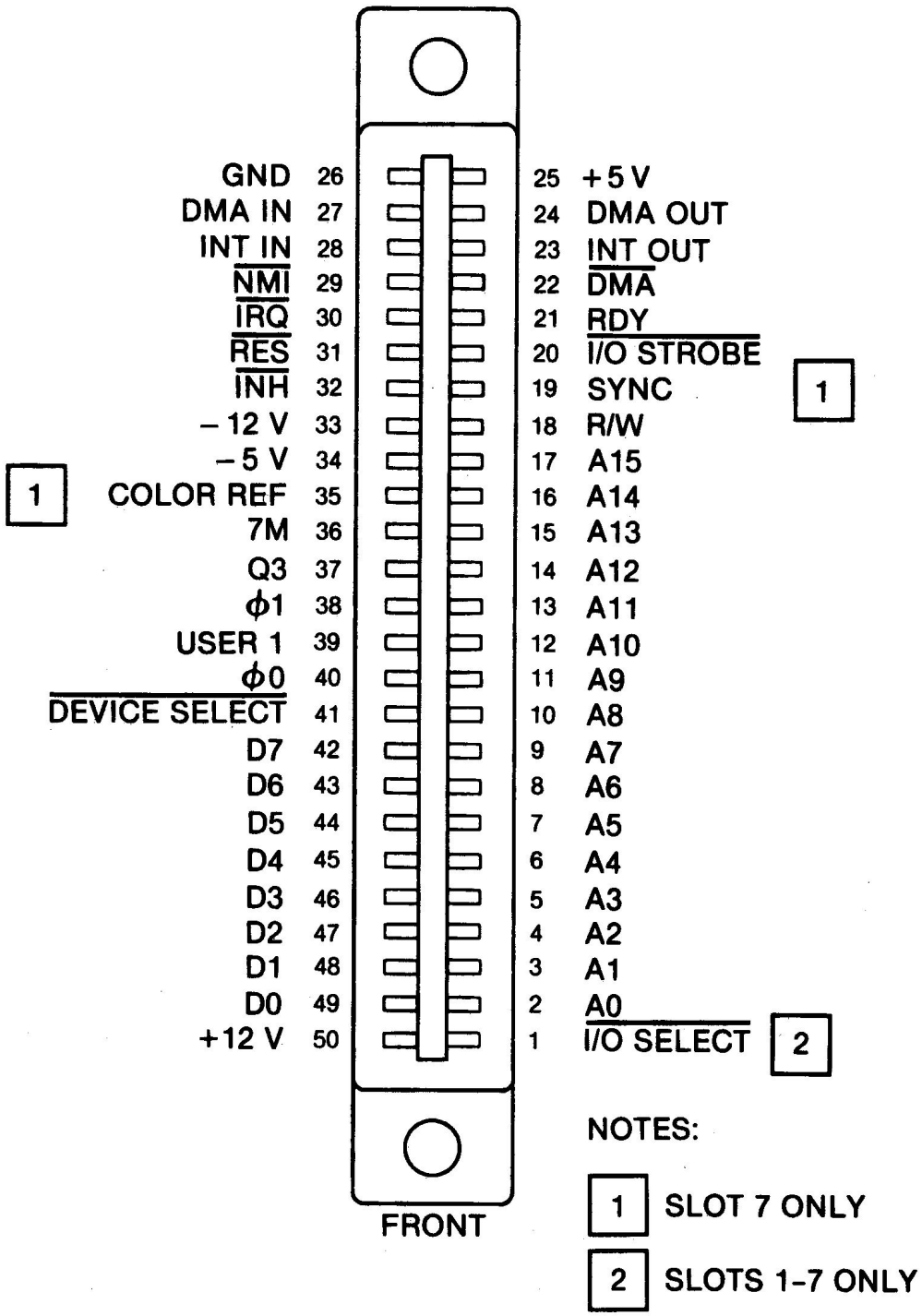


Fig. 6-17. Peripheral connector pinout. (Courtesy Apple Computer, Inc.)

Table 6-1 The Apple II Bus

Pin Number	Signal Name [1]	Mode [2]	Description
1	$\overline{\text{I/O SEL}}$	I	INPUT/OUTPUT SELECT. Seven individually decoded lines for slots 1–7. $\overline{\text{I/O SEL}}$ at any one connector goes low during $\phi 2$ when $CN00–$CNFF is accessed; N = slot number.
2–17	AD0–AD15	B	16-bit address bus. Address becomes valid during $\phi 1$ and remains valid throughout $\phi 2$.
18	R/W	B	READ/WRITE. Becomes valid during $\phi 1$ and remains valid throughout $\phi 2$. High for read; low for write.
19	SYNC	I	Video synchronization signal (C13-8). Connects to slot 7 only.
20	$\overline{\text{I/O STB}}$	I	INPUT/OUTPUT STROBE. Common select line that goes low during $\phi 2$ when $C800–$CFFF is accessed.
21	RDY	B	READY. Taken low during $\phi 1$ to insert wait states. The 6502 recognizes low on RDY only during read cycles.
22	$\overline{\text{DMA}}$	O	DIRECT MEMORY ACCESS. Taken low at the beginning of $\phi 1$ to halt the processor and make the address, data, and R/W lines high impedance.
23	INT OUT	O	INTERRUPT OUT. Daisy chain to lower priority slots. Normally driven high if used; connects to pin 28 if not used.
24	DMA OUT	O	DIRECT MEMORY ACCESS OUT. Daisy chain to lower priority slots. Normally driven high if used; connects to pin 27 if not used.
25	+5 V		+ 5 volts
26	GND		Ground
27	DMA IN	I	DIRECT MEMORY ACCESS IN. Daisy chain input from higher priority slots. Normally high.
28	INT IN	I	INTERRUPT IN. Daisy chain input from higher priority slots. Normally high.
29	$\overline{\text{NMI}}$	O	NON-MASKABLE INTERRUPT. Taken low to start a 6502 non-maskable interrupt.
30	$\overline{\text{IRQ}}$	O	INTERRUPT REQUEST. Taken low to start a 6502 maskable interrupt. Recognized only if interrupt disable flag is not set.
31	$\overline{\text{RESET}}$	B	As an output, taken low to reset the 6502 and other peripherals. As an input, goes low during resets from the keyboard, on power-up, and from other peripherals.

Table 6-1—cont. The Apple II Bus

Pin Number	Signal Name [1]	Mode [2]	Description
32	$\overline{\text{INH}}$	O	INHIBIT. Taken low to inhibit the ROM address space ($D000–$FFFF).
33	−12 V		− 12 volts
34	−5 V		−5 volts
35	COLOR REF	I	COLOR REFERENCE. Connects to slot 7 only.
36	7M	I	7MHz. 7.15909 MHz clock.
37	Q3	I	2.040968 MHz (average) clock.
38	ϕ1	I	PHASE 1. 1.020484 MHz (average) system clock. Used on bus in place of 6502 ϕ1.
39	USER 1	O	Taken low to inhibit the I/O address space ($C000–$C7FF).
40	ϕ0	I	PHASE 0. 1.020484 MHz (average) system clock and compliment of ϕ1. Used on bus in place of 6502 ϕ2.
41	$\overline{\text{DEV SEL}}$	I	DEVICE SELECT. Eight individually decoded select lines, one for each slot. $\overline{\text{DEV SEL}}$ at any one connector goes low during ϕ2 when $C0X0–$C0XF is accessed; X = N + 8, N = slot number.
42–49	D7–D0	B	Eight bit bidirectional data bus. Data becomes valid during ϕ2 and remains valid to the end of ϕ2.
50	+12 V		+ 12 volts

Notes: [1] Unless otherwise specified, all signals appear on all eight connectors.

[2] I: Input
O: Output
B: Bidirectional
With respect to peripheral card.

There are 19 clock cycles in the loop, giving a period of 19 × 978 nS = 18.58 μS. However, if an extended clock cycle falls during a period of the loop, then that period will be 18 × 978 nS + 1117 nS = 18.72 μS. Note the slight difference.

The extended clock cycles produce some subtle effects. We have just demonstrated the first effect. That is, if you are using a sequence of processor instructions to produce a short and precise delay, an extended cycle will increase your delay by 140 nS. A second effect was discussed in Chapter 3; extended cycles decrease the processor's speed from 1.023 MHz to 1.020 MHz. This slight difference would have an effect on the computation time of *very long* calculations.

A third effect arises since the processor clock is not a pure frequency—it has a 15.7 kHz component. To demonstrate this effect, consider our 'scope loop. It has an average period of 19 ÷1.020 MHz = 18.63 μS. Each time through the loop the speaker is toggled. A complete period of speaker cone movement requires two passes through the loop. This period is 37.3 μS. The frequency is 26.8 kHz—too high to

hear. Yet when you run the 'scope loop, you *do* hear a high pitched sound from the speaker. What you hear is the beat between 26.8 kHz and the 15.7 kHz clock component—this is 11.1 kHz, clearly audible.

SUMMARY

This chapter concludes with a summary of the Apple II bus. A brief review of each signal is given in Table 6-1. Fig. 6-17 shows the physical arrangement of the bus on the peripheral connectors.

In the next chapter we will examine the on-board I/O and the soft switches.

CHAPTER 7

On-Board I/O

A computer is of little use unless it communicates with the outside world via its I/O facilities. Minimum I/O usually consists of a keyboard for input and a crt display for output. Some computer architectures require plug-in peripheral boards to provide *any* I/O, including this minimum set. In the Apple II, many of the frequently used I/O devices are located on the mother board. This built-in I/O contributes to your ability to just plug in the Apple II and start computing.

In this chapter we cover the built-in or *on-board* I/O. This includes the cassette I/O, the game port, the speaker, and the keyboard.

Schematic Reference: Figs. C-12*, C-13*, and C-22*.

OVERVIEW

In Chapter 6 we mentioned that the address range $C000–$C07F was used for the on-board I/O. This range is decoded by ICs F13 and F14 (Fig. C-12*). There are nine inputs and 13 outputs listed as follows:

Inputs

Keyboard
Cassette Tape
Game Switches 0–2
Game Paddles 0–3

Outputs

Clear Keyboard Strobe
Cassette Tape
Speaker
Game Paddle Trigger
Game Utility Strobe
Game Annunciators 0–3
TEXT Soft Switch
Mixed Mode Soft Switch
Page 2 Soft Switch
HIRES Soft Switch

Keyboard

The built-in keyboard consists of two printed-circuit boards. On one board are mounted the 52 keys. On the second board are mounted five ICs and assorted dis-

crete components. Keyboard encoder IC B6 scans the 47 character keys looking for one that is depressed. These 47 keys are arranged in a 9 by 10 matrix as shown in Fig. C-22*. Not every point in the matrix is occupied by a key. This matrix arrangement allows B6 to detect a key closure using only 19 instead of 47 pins. When B6 detects a depressed key, it generates the corresponding ASCII character at its output. If the SHIFT key, CTRL key, or both are also depressed, B6 generates the ASCII character for a shift, control, or shift-control function. The output of B6 is buffered by B5 and B4-3 then connects by a short flat cable to A7 on the mother board.

Timer IC B2 is connected to oscillate at about 15 Hz. It provides a repeat strobe to B3-3 when the REPT key is depressed. The RESET key connects directly via the flat cable to A7 on the mother board. Note that the SHIFT, CTRL, and REPT keys do not generate characters unless used with another key. The RESET key also does not generate characters since it connects directly to the 6502.

Cassette Tape

The Apple II can store and retrieve digital data using a standard audio cassette tape recorder. The hardware involved is fairly simple; all the intelligence is in the firmware. When you store a program, routines in the monitor ROM cause flip-flop J13-9 (Fig. C-12*) to generate square or rectangular waveforms. Thus, J13-8 then drives the CASSETTE DATA OUT jack. This jack is meant to be connected to the microphone input of the recorder.

The earphone output of the recorder connects to the CASSETTE DATA IN jack of the Apple. When you load a program, the signals from the tape pass through operational amplifier K13, then appear at an input of data selector H14. The 6502 can read the output of H14 under program control. The program (again in the monitor ROM) then processes the cassette signal to reproduce the data.

Speaker

Tones are produced in the speaker of the Apple by toggling flip-flop J13-5 under program control. Transistor Q4 amplifies the output of J13-5 to drive the speaker.

Game Connector

Several of the on-board inputs and outputs appear at connector J14 (Fig. 7-1). Although J14 is called the "game connector," it can have other applications.

Up to three switches can connect to the J14 pins named SW0, SW1, and SW2.

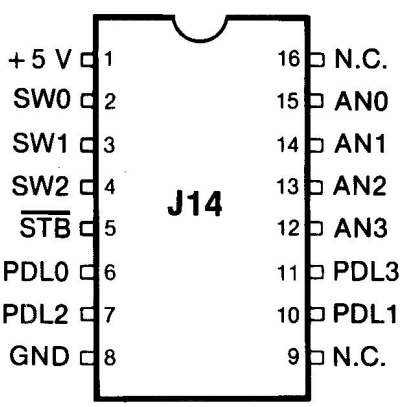


Fig. 7-1. Game connector.

The 6502 can select and read the switches via H14 (Fig. C-12*). In the game application, the switches are usually momentary-operate push buttons.

Up to four variable resistors can connect to J14 at the pins named PDL0–PDL3. These external variable resistors control the timing of quad timer H13. The four outputs of H13 connect to H14 which selects one of them to be read by the 6502. By using a software timing routine, the 6502 can measure the pulse duration of the timer. This duration is in turn a function of the external resistance connected to PDL0–PDL3. In the game application, the external resistance is usually a paddle or joystick.

Signal $\overline{\text{STB}}$ on the game connector is a utility strobe. It is activated under program control and its application is determined by the user. Signal $\overline{\text{STB}}$ goes low for 489 nS whenever the address range $C040–$C04F is accessed.

The application of game connector signals AN0 (Annunciator 0) through AN3 is also up to the user. These four signals are *soft switches*. This means they act like switches that are under software control. The software can turn them on or off, and they will stay in that condition until accessed again. An application for one of the annunciator outputs is use as a serial output port driven by a software UART.

More Soft Switches

Four of F14's outputs are soft switches used by the video circuitry to configure the video memory mapping. We made use of one of these (PAGE 2) in Chapter 5. We will find uses for the others in Chapter 8. All four are listed in Table 7-1.

Table 7-1. F14 Soft Switches

IC Pin	Signal Name
F14-4	TEXT MODE
F14-5	MIX MODE
F14-6	PAGE 2
F14-7	HIRES MODE

DETAILED CIRCUIT ANALYSIS

Two-Piece Keyboard

Keyboard Encoder IC—At the heart of the keyboard is B6 (Fig. C-22*), a type AY-5-3600 scanning keyboard encoder IC. The 9 X outputs and 10 Y inputs of B6 are arranged in a matrix with the character keys at the cross points. (*Character keys* exclude SHIFT, CTRL, REPT, and RESET.) Only 47 of the possible 90 cross points are occupied by keys. When a key is depressed, it connects the X line to the Y line.

Integrated circuit B6 contains counters that count through an X sequence and a Y sequence as it scans the keyboard looking for a depressed key. These counters are driven from an internal clock. The frequency of the clock is about 90 kHz and is set by R1 and C5. The frequency is not critical, but it must be between 10 kHz and 100 kHz for B6 to operate properly.

Fig. 7-2 shows the clock and the sequence of pulses generated on the X outputs of B6. When a key is depressed, one of the X outputs connects to a Y input. When this Y input is scanned, the key closure will be detected. Keyboard encoder B6 then "looks" at the SHIFT and CTRL key inputs and encodes the key into the proper ASCII character. The character is latched and appears inverted at B6's output (B1

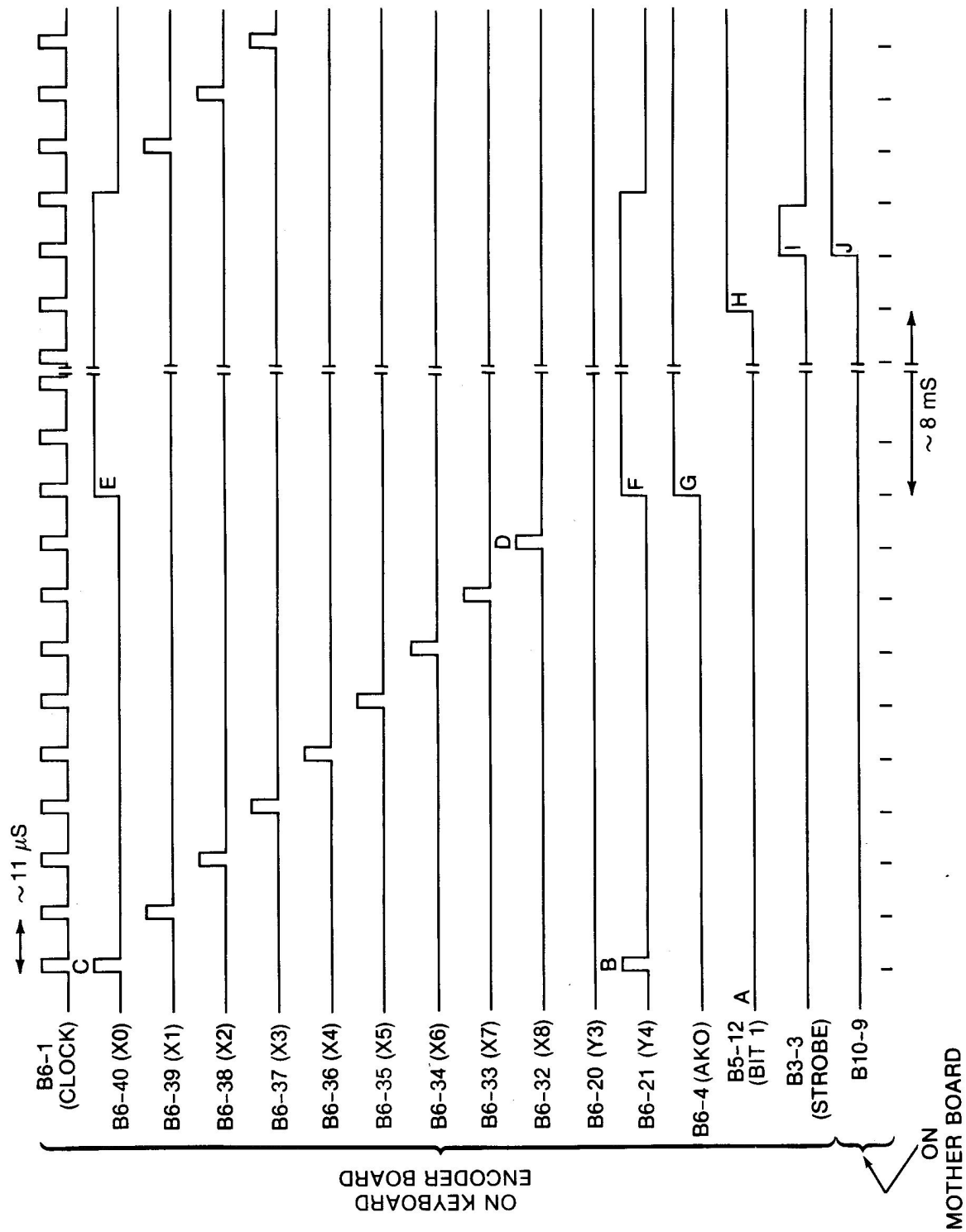


Fig. 7-2. X scan (two-piece keyboard).

through B7). (B6 is a mask programmed device. The mask determines the exact encoding of matrix cross points into output codes. The device used in the two-piece keyboard has been custom programmed for Apple Computer. This custom IC allows the two-piece keyboard to be a functional replacement for the older single-piece keyboard.)

Inverter B5 and NAND gate B4-3 buffer the character and invert it to positive-true. The character then finds its way through the flat cable to location A7 on the mother board. Since the character was latched inside B6, it will remain at B6's output until another key is depressed. Each time a key is closed, B6 outputs a pulse on its Data Strobe Output (pin 16).

Sequence of Operation—(Follow along on Fig. 7-2.) We will assume that the previous character typed was an "8." The ASCII LSB for "8" is 0, so B5-12 will be latched low from the previous character (point A). Now suppose you depress the "7" key. From the keyboard matrix we see that the "7" key connects X0 to Y4. Thus, as long as the "7" key is down, Y4 (point B) will be the same as X0 (point C). We will further assume that the Y scan is currently looking at Y3. This means the high on Y4 at point B will not be detected.

The X outputs continue to scan, completing with X8 at point D. Keyboard encoder B6 now increments its Y counter to look at Y4, and it restarts the X scan. This time when X0 and Y4 go high at points E and F, the high on Y4 *is* detected. Keyboard encoder B6 now stops scanning and takes its Any Key Down Output (AKO) high (point G).

Next, B6 waits a *Strobe Delay* time of about 8 mS (set by C2 in Fig. C-22*). The purpose of the delay is to alleviate the problem of extra characters caused by contact bounce. (Bounce refers to the rapid on and off fluctuations that occur when a mechanical switch is operated or released.)

After the strobe delay, B6 encodes the depressed key. Recall that bit 1 of the previous character was low, point A. Bit 1 of our new character (7) is high. Thus, bit 1 goes high (point H) when the key is encoded. One clock pulse later, B6-16 (Data Strobe) goes high. This high propagates through B3-6 and B3-3 to take the keyboard's strobe output high at point I. The strobe is one clock period long. The strobe's rising edge sets flip-flop B10-9 on the mother board, point J. (Chapter 6 covered the reading of the keyboard as far as circuitry on the mother board is concerned.)

Fig. 7-3 is another view of the signals that were shown expanded in Fig. 7-2. Again let's assume that "8" was the previous character so that B1 is low at point A. Sometime around point B we depress the "7" key. This connects Y4 to X0, and these two signals will be the same as long as the key is down. The key is not recognized until the Y scan reaches Y4 (point C). At this point the scanning stops, and the strobe delay starts. At the end of the strobe delay, bit 1 goes high and there is a strobe pulse (point D) as shown expanded in Fig. 7-2.

Fig. 7-4 is a more compressed view showing a complete key stroke. The key is depressed at point A and released at point B. The strobe is sent to the mother board at the end of the strobe delay, point C.

Non-TTL Levels—If you observe the matrix with an oscilloscope, you will find that B6's X outputs and Y inputs are not at TTL levels. Their typical voltage levels are shown in Fig. 7-5 along with the clock waveform from B6-1.

Repeat Function—B2 is a 555 timer connected to oscillate at a frequency set by R8, R7, and C7 (Fig. C-22*). The frequency is

$$\frac{1.44}{(R8 + 2R7)C7} = 15 \text{ Hz}$$

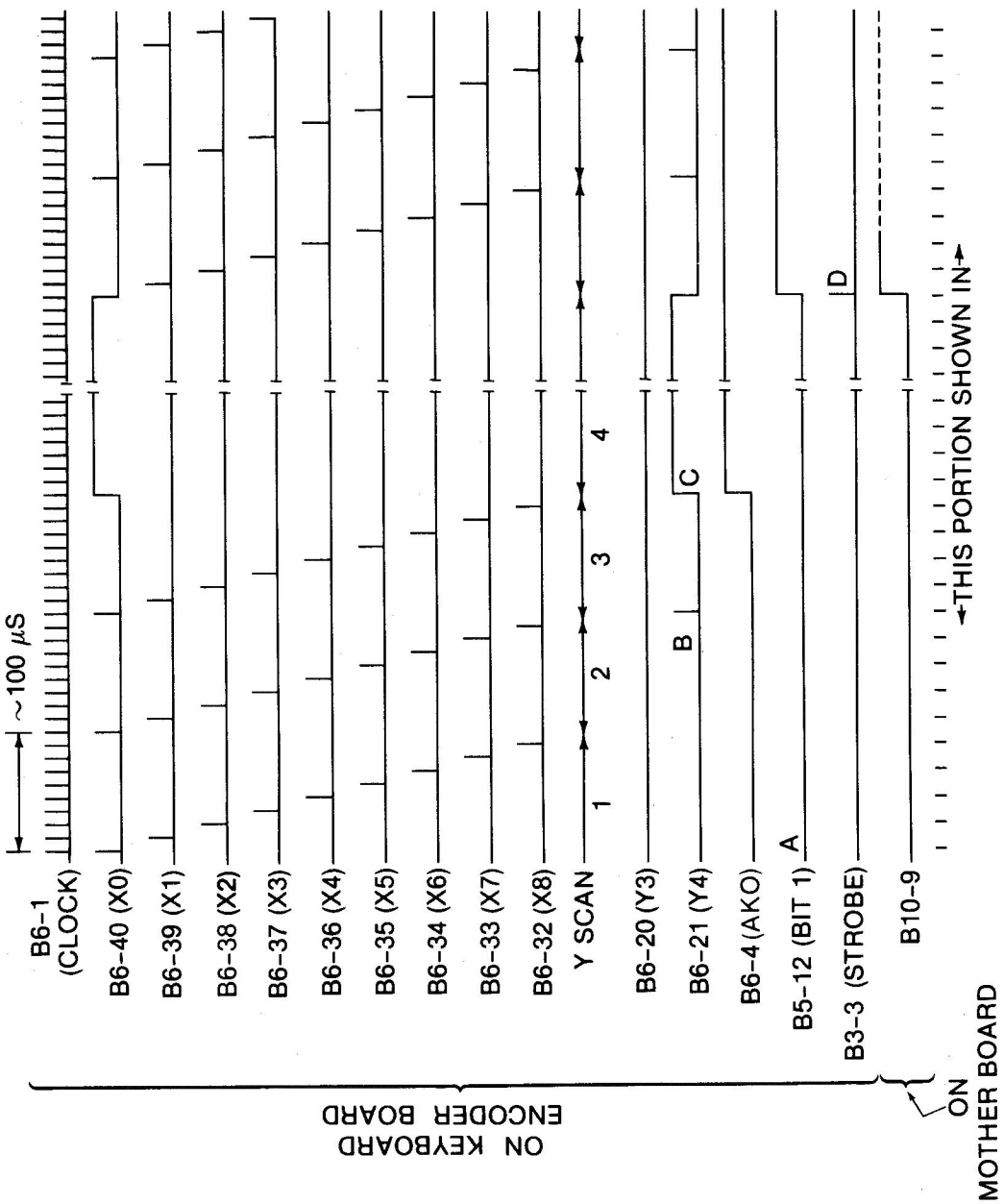


Fig. 7-3. Y scan (two-piece keyboard).

B2 is normally held reset by R10. When the REPT key is depressed, and another key is also pressed (AKO high), then the reset is removed. Timer B2 now oscillates and its output is coupled through C6 to gates B4-11, B4-8, and B3-3. The 15-Hz pulses at B3-1 create additional strobe pulses to the mother board.

Roll Over and Phantoms—The keyboard encoder IC is capable of *N-key roll over.* N-key roll over means that you can have any number of keys down simultaneously, and the encoder will correctly send the characters in the order keyed. This capability, however, can be used only if there is a diode at every cross point in the key matrix. The Apple II keyboard does not have these diodes. Without them, de-

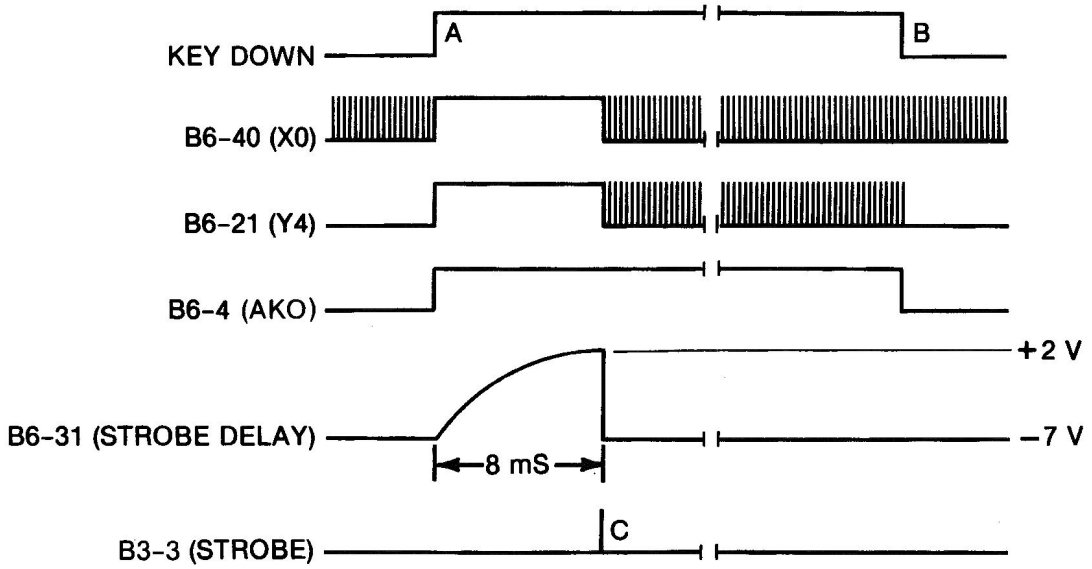


Fig. 7-4. Strobe delay (two-piece keyboard).

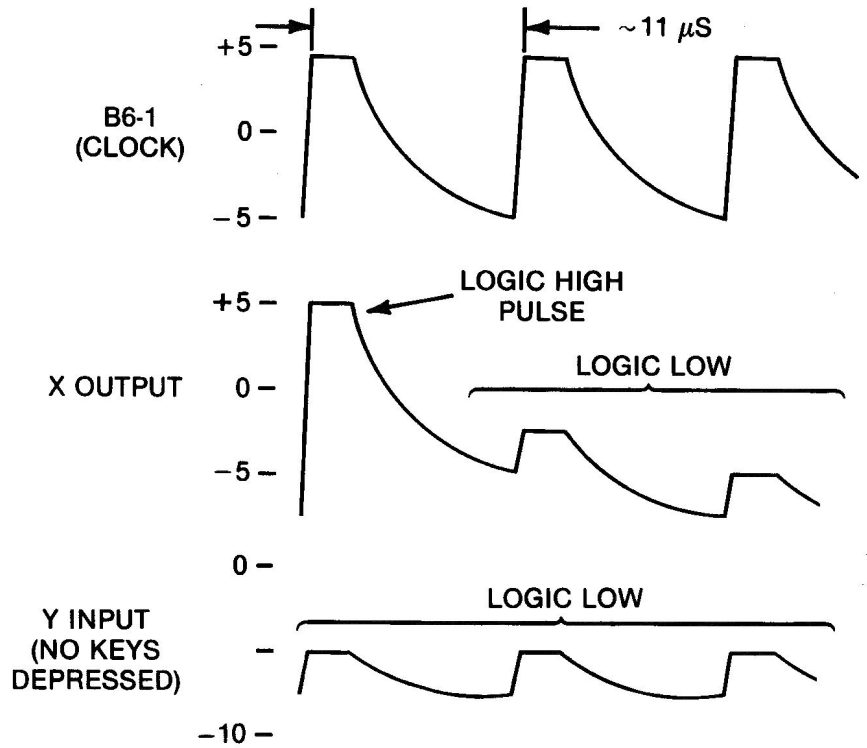


Fig. 7-5. Typical AY-5-3600 keyboard encoder waveforms (two-piece keyboard).

pressing three or more keys simultaneously may generate extraneous or *phantom* characters. As a result, the keyboard is described as having *2-key roll over*. No more than two keys may be depressed simultaneously if phantom characters are to be avoided.

It is the sneak paths that are possible in a matrix without diodes that generate the phantom characters. For example, simultaneously pressing the space, N, and B keys will connect X4 to Y4. This creates a phantom "A." Simultaneously pressing space, N, and M will connect X4 to Y6. No key is assigned to this matrix cross point. However, the AY-5-3600 will still output a character. In the original AY-5-3600

mask, all unused matrix cross points were programmed to be an ASCII *null.* A null typed inadvertently in a Pascal program can be undesirable. A later mask for the AY-5-3600 programs all unused cross points to be an ASCII *space.* These two versions of encoder IC are identified as 331-0931-A and 331-0931-B, respectively.

Quad-Mode Operation—B6 is a *quad-mode* encoder. This means each key can generate up to four different characters as a function of the SHIFT and CTRL keys (Table 7-2).

Table 7-2. Characters as a Function of Control and Shift

B6-28 (CTRL)	B6-29 (SHIFT)	Character Group
0	0	Unshift
0	1	Shift
1	0	Control
1	1	Shift-Control

The four characters for each key are shown in Fig. C-22*. Only the "P," "M," and "N" keys generate four different characters. The rest generate fewer.

Lowercase—There are two bow ties on the board with the ICs. They may be cut as a user option. The circuit is then restored by adding switch S2. In the normal position of S2, the circuit functions as previously described. When S2 is operated, output bits 9 and 8 are substituted for bits 5 and 6. Encoder B6 has been programmed with ASCII lowercase letters. This substitution of bits will make the lowercase letters available. Lowercase operation is as follows: with the shift key not depressed, pressing any letter key will output lowercase. With the shift key pressed, pressing any letter key will output uppercase. The three characters @,], and ^ are not available directly from the keyboard when S2 is in its lowercase position.

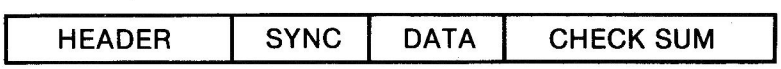
Numeric Pad—As a user option, a nine-pin connector may be installed on the keyboard at J2. Then J2 can be extended to a numeric pad as shown in Fig. C-22*.

Accidental Reset—To prevent accidental resets, S1 may be moved to its "CTRL" position (pins 1 and 2 connected). In this position of S1, it is necessary to press both CTRL and RESET to reset the Apple.

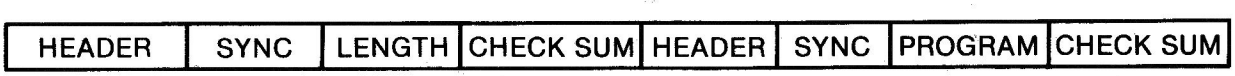
Cassette Tape

In this section we will describe the hardware involved in writing and reading cassette tapes. We will also touch on some aspects of the cassette system that are actually determined by the firmware.

Data to be recorded on tape is formatted into *records.* Each record consists of several seconds of *header tone* followed by a *sync bit,* the actual data, and a *check sum.* Binary files and machine code are recorded as one record (Fig. 7-6A). BASIC



(A) Binary files or machine code.



(B) BASIC programs.

Fig. 7-6. Cassette tape formats.

programs are recorded as two records (Fig. 7-6B). The first record contains the BASIC program's length in the record's data field. The second record contains the BASIC program itself.

Fig. 7-7 shows individual bits or cycles of the tones that make up the format. The times shown are approximate, but they are within 2% of measured values. The header tone consists of half cycles of 650 microseconds each. This is a frequency of 770 Hz. The sync bit contains one-half cycle of 200 microseconds followed by one-half cycle of 250 microseconds. A 0 bit consists of two 250 microsecond half cycles. Data of all 0's would be a tone of 2000 Hz. A 1 bit consists of two half cycles of 500 microseconds each. Data of all 1's would be a 1000-Hz (1-kHz) tone. It takes 500 microseconds to transmit a 0 and 1000 microseconds to transmit a 1. The average transmission rate for random data is therefore 1333 bits per second.

Data is recorded low byte first. Within each byte the MSB is recorded first.

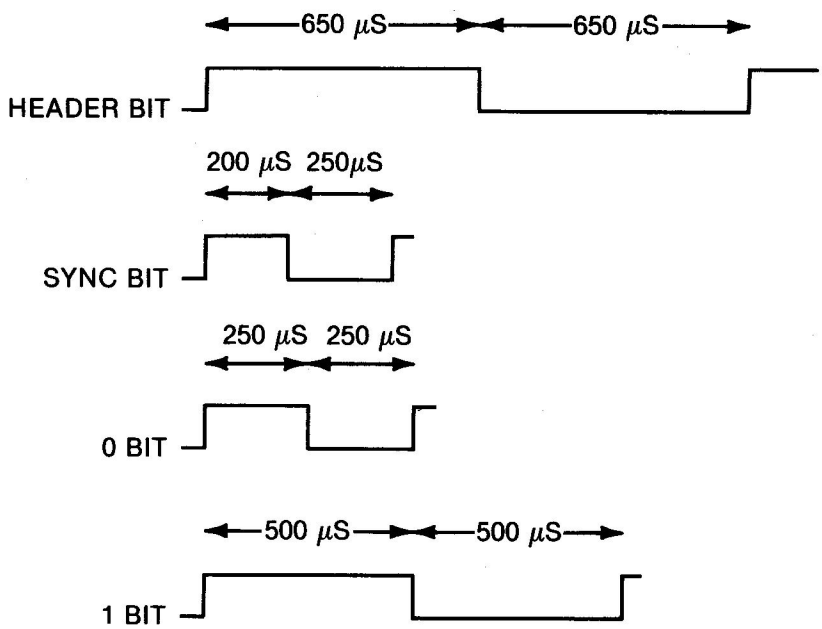


Fig. 7-7. Cassette bit timing.

Fig. 7-8 shows typical tape cassette waveforms near the time of the sync bit. When the cassette output ($C020–$C02F) is accessed, J13-11 goes low for 489 nS (point A). On J13-11's rising edge, J13-8 is clocked (point B). Since J13-8 connects to J13-12 (Fig. C-12*), this flip-flop will toggle when clocked. In Fig. 7-8 we have shown the last two and one-half cycles of the header tone. The sync bit comes next, followed by the first data byte. In this example, the first byte is $B0 or 10110000 (binary).

Note that on power-up, J13-8 can be either low or high. The processor can neither initialize nor read the state of J13-8. As a result, the J13-8 waveform could be inverted from what is shown in Fig. 7-8. No matter in which state J13-8 starts, it toggles every time J13-11 goes high. That is all that matters.

Resistors R18 and R19 attenuate J13-8 to about 0.8%, or 32 mV peak-peak open circuit. The attenuated signal appears at the CASSETTE DATA OUT jack of the Apple. It is connected to the microphone input of the recorder when data is stored. This concludes the discussion of the hardware used in the write direction. We now turn our attention to the read process.

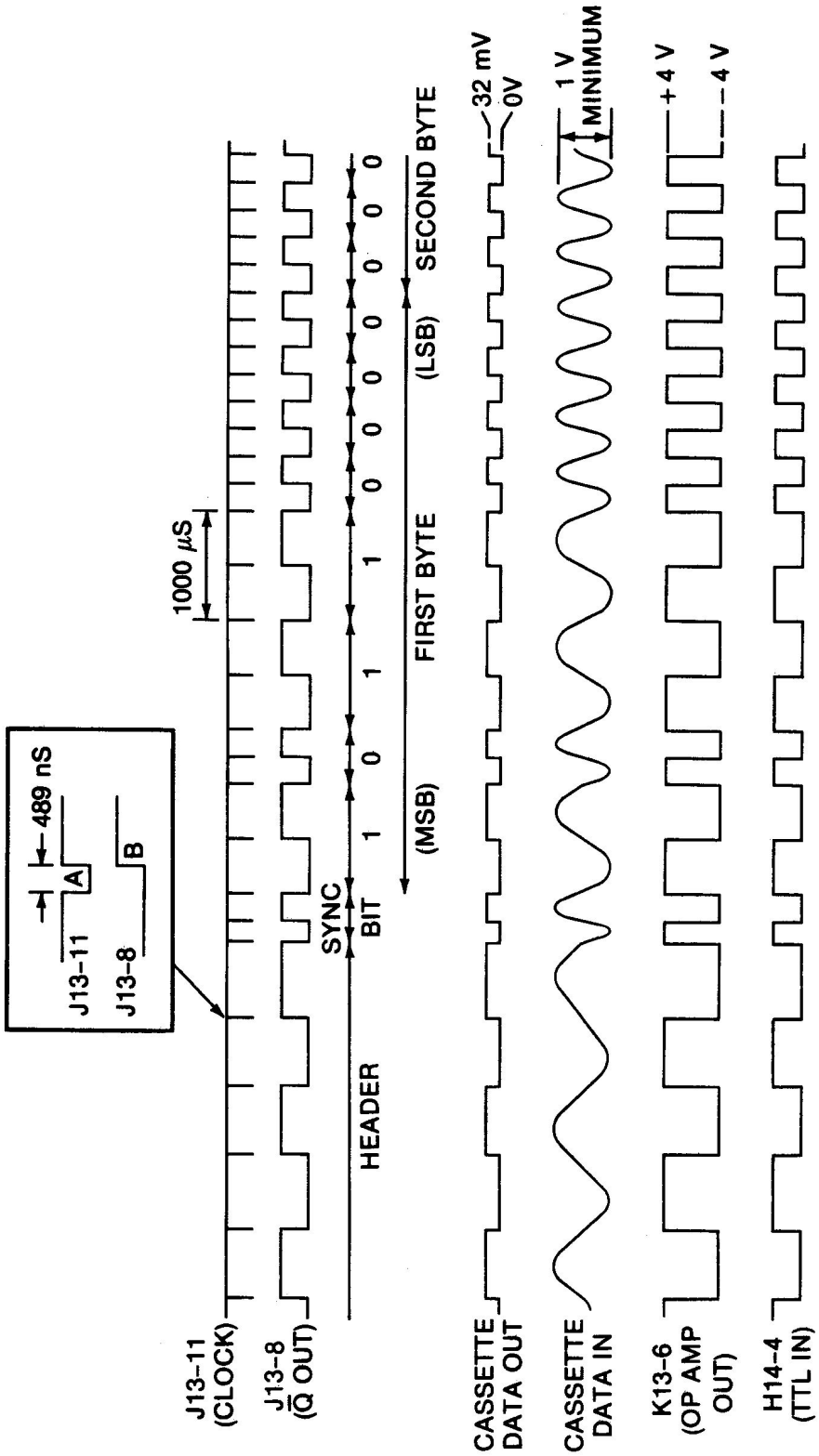


Fig. 7-8. Tape cassette waveforms.

On data retrieval, the earphone jack of the recorder connects to the CASSETTE DATA IN jack of the Apple. There may or may not be a polarity inversion in going through the recorder; it does not matter. The cassette input is ac coupled by C10 and attenuated to 50% by R17 and R30. The signal then connects to the inverting input of operational amplifier K13.

Operational amplifier K13 is wired to act as a comparator. Ground through R16 provides an average comparator threshold of 0 volt and R15 provides about 100 mV of hysteresis at the input of the comparator. This circuit arrangement is called a zero crossing detector. The minimum cassette input that produces good results is about 1 volt peak-peak. While reading tapes, the output of K13 swings from plus-saturated to minus-saturated, or about ± 4 volts.

Resistor R29 limits current out of the input of H14 when K13-6 is low. Resistor R29 and the input clamping diode of H14 effectively convert the signal to a TTL level at H14-4. Note that the signal is inverted by K13. Except for an inversion, the TTL signal at J13-8 has been recreated at H14-4. To see this, compare the waveforms for J13-8 and H14-4 as shown in Fig. 7-8.

At this point we must discuss how the cassette signal gets through data selector H14 (Fig. C-12*). If the address is in the range $C060–$C06F, then F13-9 will go low during $\phi 2$. This enables H14 via pin 7. Data selector H14 selects one of its eight inputs as a function of address lines AD0, AD1, and AD2. When the address is $C060 or $C068 (AD0, AD1, and AD2 all equal 0), H14 is enabled and selects input pin 4 to output on pin 5. Pin H14-5 connects to data bus bit 7. Thus, at either of these addresses, the cassette input is put on data bus bit 7. Bit 7 is used since it is easily tested by various 6502 instructions. Since AD3 is not decoded by this part of the circuit, there are two addresses eight locations apart that can access the cassette input. In such cases, we usually use the lower address ($C060).

Routines in the monitor scan bit 7 at location $C060 every 12.8 microseconds looking for a transition. By measuring the time between transitions, the routines can distinguish between header, sync, 0, and 1 bits.

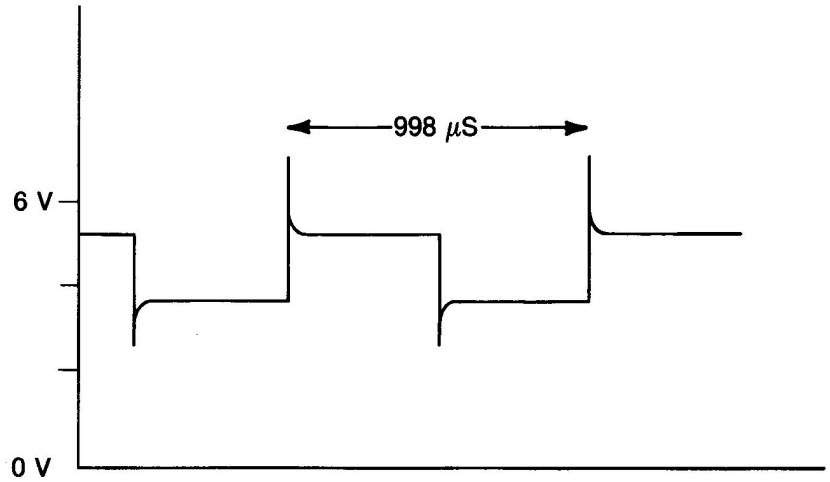
Speaker

Decoder F13-12 goes low during $\phi 2$ whenever the range $C030–$C03F is addressed. On the rising edge of F13-12, J13-5 is clocked. This flip-flop will toggle since pin 6 is tied to pin 2. It takes two accesses to $C030 to generate one period of speaker oscillation. The square or rectangular wave at J13-5 is coupled through C11 and R24 to the base of Q4. Diode CR1 provides a discharge path for C11. Transistor Q4 is a Darlington pair that amplifies the signal to drive the speaker through R25. Capacitor C12 provides some integration and also absorbs some of the energy stored in the speaker winding when Q4 turns off.

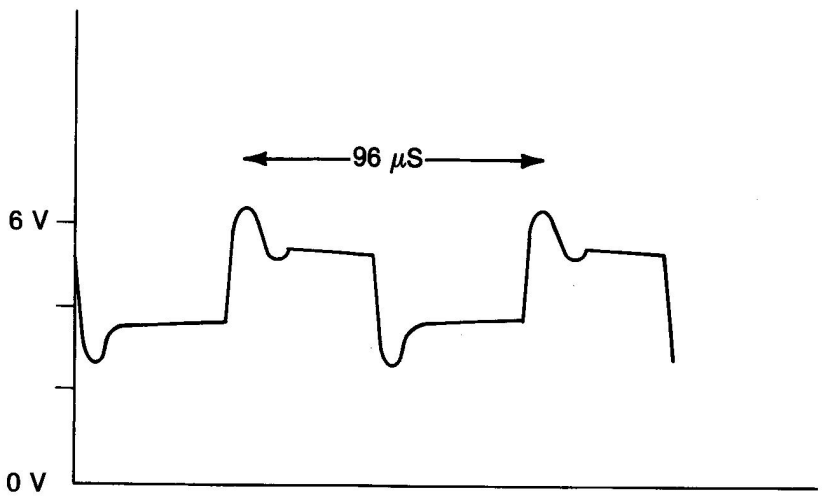
To generate tones from the speaker, you must calculate the number of clock cycles needed between each toggle. The clock cycles are then obtained using a software loop. Using this method, you can generate tones from below to above the audible range. If you want absolute pitch, remember that the average clock cycle is 980 nS long.

Fig. 7-9 shows the voltage (to ground) measured at the internal speaker for 1002-Hz and 10.4-kHz tones. Note the overshoot at each transition.

You can get a much fuller sound by disconnecting the small internal speaker and connecting a larger external speaker. Be careful with the speaker leads; one side is connected directly to +5 volts.



(A) 1002-Hz tone.



(B) 10.4-kHz tone.

Fig. 7-9. Speaker output.

Game Switches

The three game switches (SW0–SW2) connect to inputs of data selector H14 (Fig. C-12*). The operation of H14 was described in the section on the cassette input. Here we list the addresses that will select the game switches (Table 7-3).

Table 7-3. Addresses and Game Switches

Location	Game Switch
$C061 and $C069	SW0
$C062 and $C06A	SW1
$C063 and $C06B	SW2

A game switch is typically wired as shown in Fig. 7-10. With the switch normal, the 560-ohm resistor pulls SW0 low. When $C061 is accessed, data bit 7 will go low

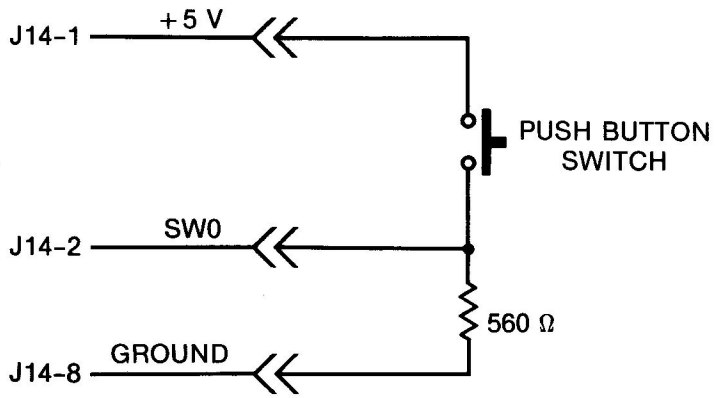


Fig. 7-10. Game switch—SW0 is shown as typical.

(0). When the switch is operated, SW0 will connect to +5 volts (high). Now when $C061 is accessed, data bit 7 will go high (1). The operation of SW1 and SW2 is identical to that of SW0.

Game Paddles

Quad timer H13 is a type 558. Each of the four sections has an isolated trigger input, timing input, and output (Fig. C-12*). Two other inputs are common to all sections: $\overline{\text{Reset}}$ (pin 13) and Control Voltage (pin 4). In the Apple, $\overline{\text{Reset}}$ is not used and is tied to +5 volts. Control Voltage is not used either, but since this pin can influence an internal threshold voltage, it is bypassed by C9. In the Apple, the trigger inputs for each section are tied in common.

Let us examine the operation of a typical 558 section (see Fig. 7-11A). The output stage is open collector, so a pull-up resistor is needed to see any voltage changes at the output. Normally the output is low, and in this condition an internal transistor shunts the timing input to ground. When the trigger input makes a high-to-low transition, the output goes high and the shunt is removed from the timing input. Capacitor C begins to charge through resistor R. When the timing input reaches an internally set threshold (the control voltage), the output goes low and the timing input is again shunted to ground. The output high time is equal to the product RC.

Once the timer has triggered, the trigger input has no effect until the output is low again. This characteristic of the trigger input will be used later in this section to explain a well-known paddle phenomena. First, however, we will see how the 558 is used in the Apple.

Apple's paddle scheme functions by replacing R in Fig. 7-11A with a variable resistor and measuring the timer's pulse width using the 6502. Here are the circuit details. The trigger inputs of all four sections connect in common to F13-7 (Fig. C-12*). This line goes low during $\phi 2$ in the address range $C070–$C07F. The 6502 uses address $C070 to trigger all four timers. The output of each timer connects to an input of H14. This IC can select one of these inputs in the same manner that it selects an input from the game switches or the cassette (described in previous sections). The 6502 reads the timer outputs by addressing the locations shown in Table 7-4.

The timing capacitors for each section (C5, C6, C7, and C8) have values of 0.022 microfarad. The timing resistors for each section consist of 100 ohms (Ω) on the mother board (R20, R21, R22, and R23) plus the resistance of the paddles. Fig. 7-12

Table 7-4. Addresses and Game Paddles

Location	Paddle
$C064 and $C06C	0
$C065 and $C06D	1
$C066 and $C06E	2
$C067 and $C06F	3

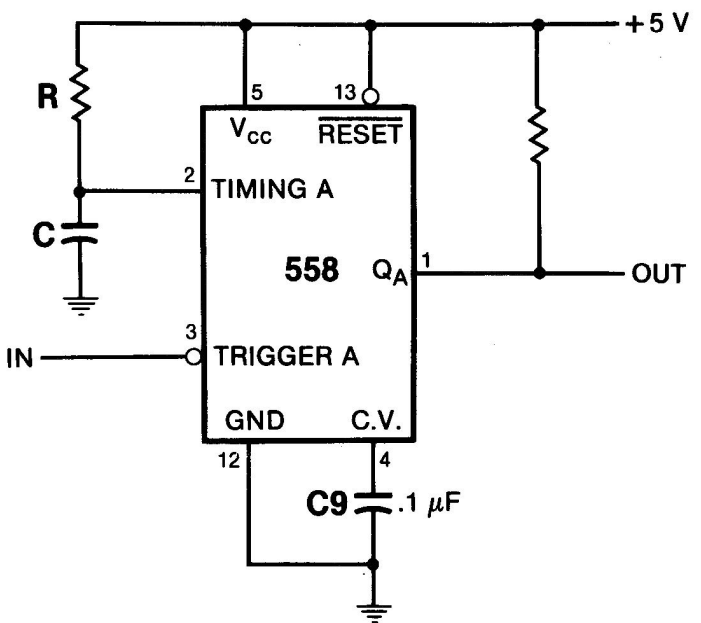
shows the usual wiring of a game paddle. Apple recommends a 150 kΩ value. The longest pulse width is

$$(150\ \text{k}\Omega + 100\ \Omega) \times 0.022\ \mu\text{F} = 3.30\ \text{mS}$$

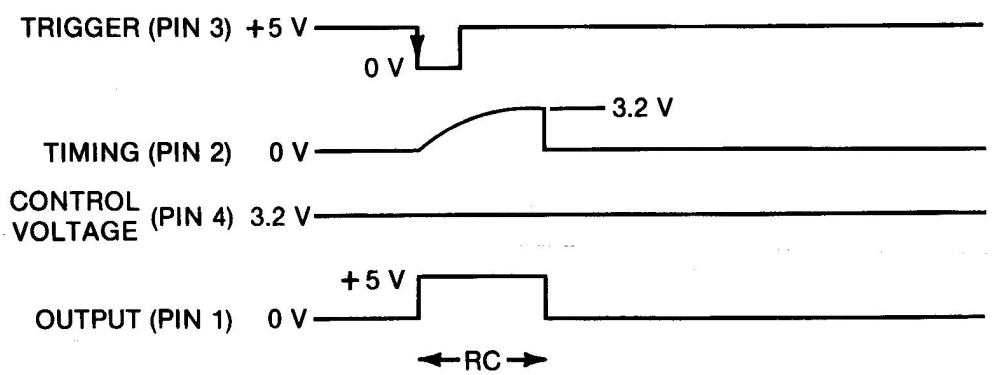
The shortest pulse is

$$100\ \Omega \times 0.022\ \mu\text{F} = 2.2\ \mu\text{S}$$

The purpose of the 100 Ω resistor is to provide some protection for H13 from external shorts and from the condition when the external resistance goes to 0.



(A) Typical wiring diagram.



(B) Signal waveforms.

Fig. 7-11. 558 timer.

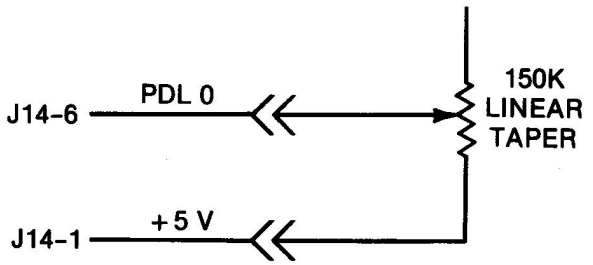


Fig. 7-12. Game paddle—PDL0 is shown as typical.

That's it for the hardware, now how does the firmware determine the position of the paddle? There is a monitor routine at $FB1E that triggers H13, then enters a loop to count until the output of H13 returns low. Fig. 7-13 shows the flowchart. You enter the routine with the paddle number (0–3) in the X register. The routine first triggers H13 (F13-7 goes low). Next the Y register, used for the counter, is initialized to 0. The routine now delays about 4 μS, then reads the timer output for the specified paddle (F13-9 goes low). If the paddle output is low on the first read, this indicates a very short pulse (low resistance). You exit the routine with Y = 0.

If the first read is not low, the routine increments the counter (Y register) and then checks for counter overflow. By overflow we mean that the Y register overflows (counts) from 255 to 0. If there is no overflow, the routine reads the paddle again. The time around the loop is 11 clock cycles, or 10.8 μS. When the timer output goes low, you exit the routine with the paddle position value in the Y register. At the high resistance end of the paddle, the counter may overflow. In this case, the routine decrements the counter and you exit with Y = 255. Note that as the paddle moves from lowest to highest resistance, the paddle value changes from 0 to 255.

Recall that there is a minimun of 100 Ω designed into the circuit. The game paddle may also have some "end resistance." These two items together produce a minimum timer pulse of several microseconds. The 4 μS initial delay in the routine bridges this minimum pulse so that the minimum paddle value can be 0.

At the other end of paddle rotation, we found that 150 K gave us a 3.30 mS pulse. This is equivalent to

$$\frac{3.30\ \text{mS}}{10.8\ \mu\text{S}} = 305\ \text{counts}$$

Thus, we are assured of reaching the maximum paddle value of 255.

You may already know that if you read the value of one paddle too soon after reading another paddle, the value of the second paddle may be in error. Fig. 7-14 shows why this is so. We will take the case where PDL 0 is set for a value of 45 and PDL 1 is set for a value of 240. At point A, the program triggers both timers. After about 500 μS, timer 0 times out (point B), and the program displays a PDL 0 value of 45.

As fast as Integer BASIC can, it triggers the timers again (point C). Timer 0 outputs another pulse (point D), but we don't care since we are now reading timer 1. Timer 1's output is still high from the first trigger; it will not time out until point E. Timer 1 has ignored the trigger at point C. The monitor routine times from point C to point E (about 450 μS) and returns with a PDL 1 value of 40, clearly in error. To correct the situation, you must insert some delay between consecutive reads of different paddles. The total delay between consecutive triggers should be greater than 4 mS.

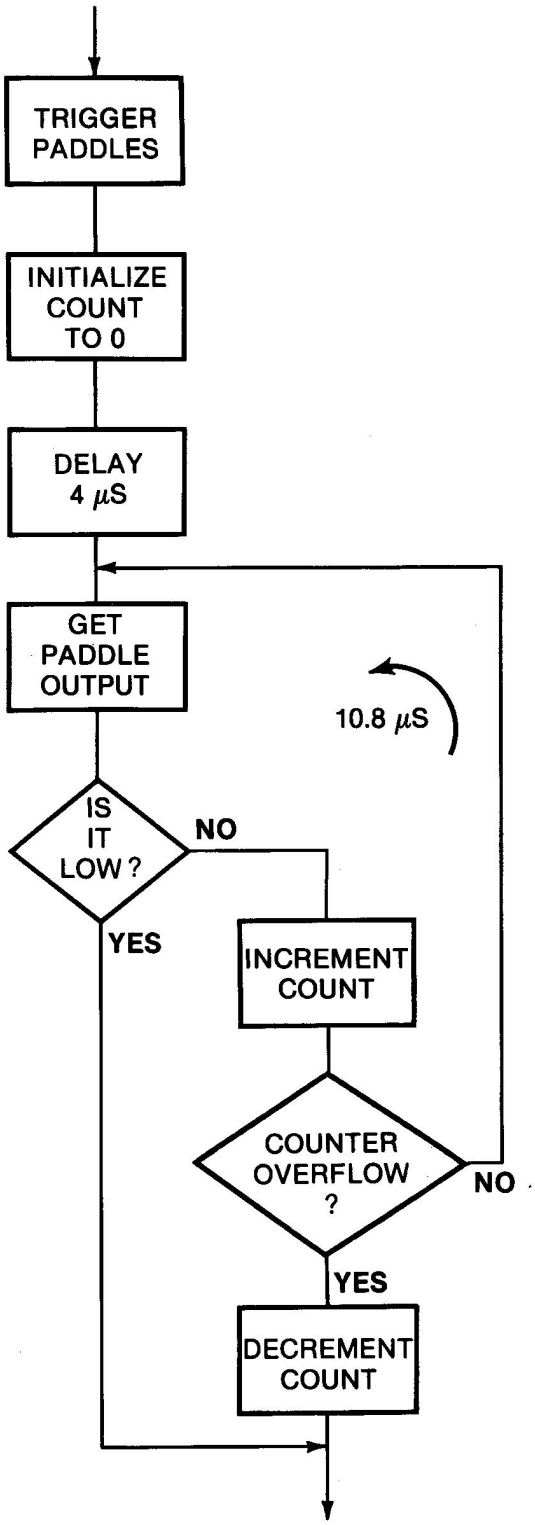


Fig. 7-13. Read game paddle flowchart.

If you examine the outputs of H13 with an oscilloscope, you may see some strange voltage levels. This is due to the lack of pull-up resistors on H13's open collector outputs. The only pull-up on these lines comes from the inputs of H14. The result is a high level of about 1.5 volts instead of the more typical TTL high level of

Table 7-5. Soft Switches

Pin	Signal	State*	Location	Function
F14-4	TEXT MODE	Off	$C050	Graphics or mixed
		On	$C051	All-Text**
F14-5	MIX MODE	Off	$C052	All-Text or graphics
		On	$C053	Mixed
F14-6	PAGE 2	Off	$C054	Select page 1
		On	$C055	Select page 2
F14-7	HIRES MODE	Off	$C056	Select LORES
		On	$C057	Select HIRES
F14-9	AN0	Off	$C058	
		On	$C059	
F14-10	AN1	Off	$C05A	
		On	$C05B	
F14-11	AN2	Off	$C05C	
		On	$C05D	
F14-12	AN3	Off	$C05E	
		On	$C05F	

*Off = 0 = Low, On = 1 = High
**When All-Text is selected, F14-5 and F14-7 are don't cares.

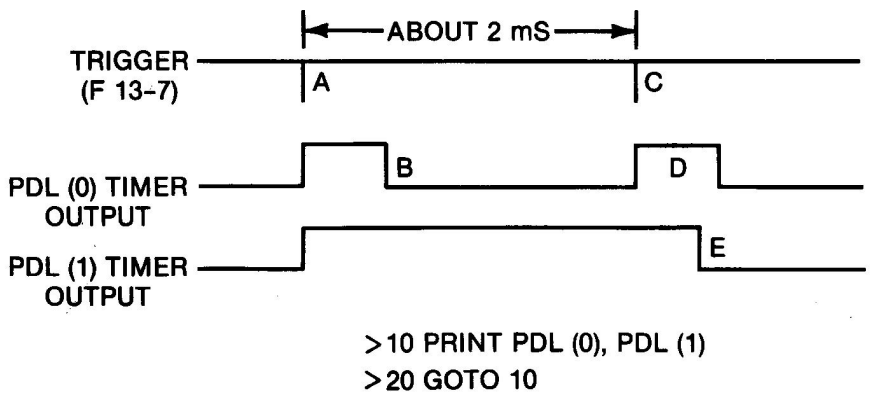


Fig. 7-14. When two paddles are read consecutively without inserting delay, the second paddle value may be in error.

3.5 volts. You may also find that the 1.5 volt high level varies as a function of the other inputs of H14. A *low* level out of H13 looks normal however (0 to 0.4 volt).

Soft Switches

The eight soft switches appear at the outputs of addressable latch F14 (Fig. C-12*). Whenever the range $C050–$C05F is accessed, F13-10 goes low during $\phi 2$. This enables F14. The three address bits AD1, AD2, and AD3 select one of the eight outputs of F14. When enabled, F14 latches the data on pin 13 into the selected output. Input F14-13 connects to address bit AD0, so even addresses will latch a 0 into the output. Likewise, odd addresses will latch a 1 into the selected output.

Table 7-6. On-Board I/O Locations

Location	Function	Remarks
$C000	Keyboard Data Input	*
$C010	Clear Keyboard Strobe	*
$C020	Toggle Cassette Output	*
$C030	Toggle Speaker	*
$C040	Game Utility Strobe	*
$C050-$C05F	Soft Switches	See Table 7-5
$C060 & $C068	Cassette Input	
$C061 & $C069	Game SW0	
$C062 & $C06A	Game SW1	
$C063 & $C06B	Game SW2	
$C064 & $C06C	Game PDL0	
$C065 & $C06D	Game PDL1	
$C066 & $C06E	Game PDL2	
$C067 & $C06F	Game PDL3	
$C070	Game Paddle Trigger	

*Any of 16 locations starting with the listed location will perform the same function.

Two output states times eight different outputs is equal to 16. Thus, all 16 locations in the range $C050–$C05F are separately decoded by F14. Table 7-5 shows each location and its function. For example, to select a mixed TEXT and LORES display from page 1, you would access locations $C050, $C053, $C054, and $C056. Use of the soft switches to control the display will be covered in detail in Chapter 8.

SUMMARY

Table 7-6 is a summary of the I/O address locations covered in this chapter.

In the next chapter, we will explore the video display modes of TEXT, LORES, HIRES, and mixed.

CHAPTER 8

The Video Display

The Apple II's video display is one of its greatest assets. There are three basic modes: black and white text, 16-color low resolution graphics (LORES), and 6-color high resolution graphics (HIRES). Two additional modes allow text to be mixed with either LORES or HIRES graphics. All three basic modes can display either a primary page of memory (Page 1) or a secondary page of memory (Page 2). This chapter describes the video display circuitry and explains how colors are generated by the digital electronics.

OVERVIEW

We start our overview of this chapter with a review of Apple's video display characteristics. In text mode, the display is 40 characters wide by 24 characters high. Each of these 960 characters is stored as one byte in a 1024 byte block of memory. The range of this block is shown as follows:

Page 1	$400–$7FF
Page 2	$800–$BFF

The page to be displayed is selected by the PAGE 2 soft switch. (Table 7-5 lists the soft switch address locations.)

In LORES mode, the display contains 1920 *pixels* arranged 40 wide by 48 high. A pixel is a *picture element,* and is used here to describe one of the LORES color blocks. LORES uses the same Page 1 and Page 2 memory ranges as text mode. In LORES, each byte stores two vertically adjacent pixels. Of a byte, the four low order bits store the upper pixel, and the four high order bits store the lower pixel. The information stored in the four bits is the pixel's color. There are 16 colors possible.

In HIRES mode, the display is 192 pixels high. A HIRES pixel is a mere dot, much smaller than a LORES pixel. It is a little difficult to specify the number of HIRES pixels in the horizontal direction. The width of a pixel is such that 280 pixels can fit on one horizontal line. Thus, the HIRES horizontal resolution is usually defined as 280 pixels. However, there are actually 560 *positions* where pixels can be placed. Here is how we explain this concept. Divide the line into 280 equal parts

called *cells*. The design of the Apple is such that you can position a pixel to be coincident with a cell. You can also position a pixel so that it overlaps two cells, one-half in one and one-half in another. Taken over the full width of the line, there are then 560 positions where you can locate a pixel.

We have not yet said anything about the color of the pixel. On a black and white monitor, there is no color and each pixel is either on (white) or off (black). In black and white HIRES we have 560 horizontal plotting positions.

Color HIRES is a different matter. Here the pixel's exact position will determine its color. If we step a single pixel across the screen, stopping at each of the 560 locations, the pixel's color will sequence through violet, blue, green, orange, violet, etc. Any one color can appear at only one out of four positions. Thus, the color HIRES horizontal resolution could be defined as 140 pixels.

There are two more limitations on color HIRES resolution. First, two adjacent pixels will not display in color, but will be a white dot. Of course you may want a white dot. Second, the pixels stored by any one byte must be selected from one and only one of the following two lists:

List 1	List 2
Violet	Blue
Green	Orange
White	White
Black	Black

Thus, for example, you cannot mix green and orange pixels in the same byte.

One memory byte stores the pixels for seven adjacent cells. The 280 cells per line result in 40 bytes per line, and the 192 lines per screen result in 7680 bytes per screen. In practice, an even 8K-byte memory block stores one HIRES screen:

Page 1	$2000–$3FFF
Page 2	$4000–$5FFF

There are two varieties of mixed mode: text mixed with LORES graphics and text mixed with HIRES graphics. In both cases, the lower one-sixth of the screen contains four lines of text characters.

Fig. 8-1 is a high level block diagram of the video display process. When you display text or graphics, you first configure the circuitry for the desired mode by setting the soft switches under 6502 software control. Next you again use the 6502 to write to the desired locations in the screen memory. Meanwhile the video address generator is always running, using the system clock to create the video address, blanking, sync, and color burst signals. The video address maps to a memory address, and then accesses the screen memory (RAM). The RAM's data is latched and made available to the video generator. The video generator is enabled and disabled by the blanking signal. The video generator also uses inputs from the soft switches to configure itself for the selected mode. Its function is to convert the latched data into a video output. This video output is combined with the sync and color burst signals in Q3 (Fig. 8-1). The output of Q3 is the Apple's composite video output which is used to drive a monitor or an rf modulator. The video generator is the subject of this chapter; the other blocks in Fig. 8-1 have been covered in preceding chapters as noted in the figure.

Fig. 8-2 is a block diagram of the video generator. The ICs that make up each

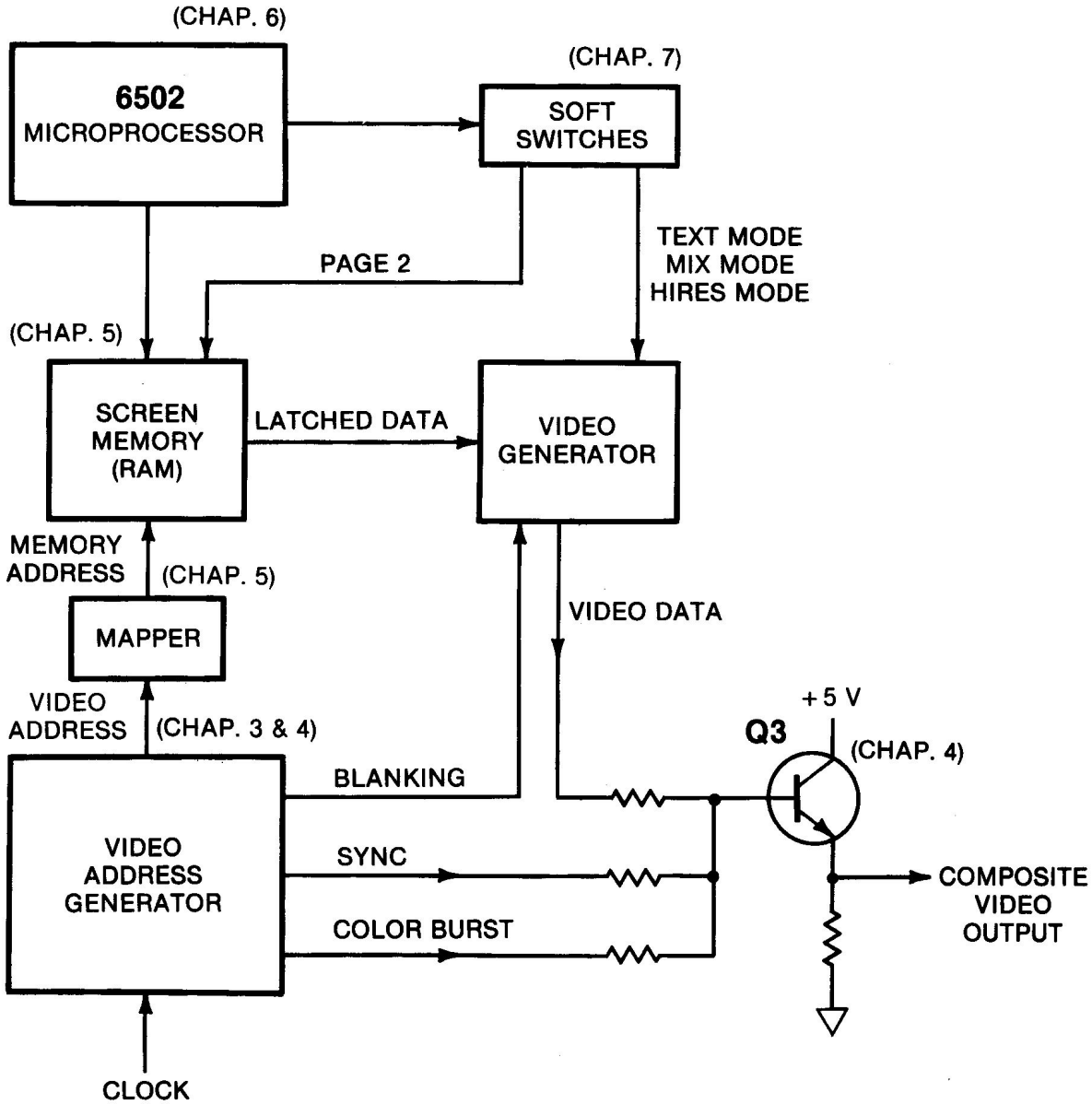


Fig. 8-1. Block diagram of video display process.

block are noted in the figure. The text generator consists of a character generator, a shift register, a timer (for flashing characters), and gates to produce flashing and inverse text. The text generator is always running. It processes the latched data at its input into text at its output. The text video becomes an input to data selector A9.

Shift registers B4 and B9 are configured by data selector A8. When HIRES MODE is low, each shift register takes four bits of latched data at its input and produces a LORES pixel at its output. The LORES video becomes another input to data selector A9.

When HIRES MODE is high, A8 configures B4 and B9 into one long shift register. This long shift register takes the latched data at its input and produces seven HIRES pixels at its output. The HIRES video becomes a third input to A9. Note that shift registers B4 and B9 are always running, producing either LORES or HIRES video.

The TEXT MODE, MIX MODE, and HIRES MODE soft switches cause data

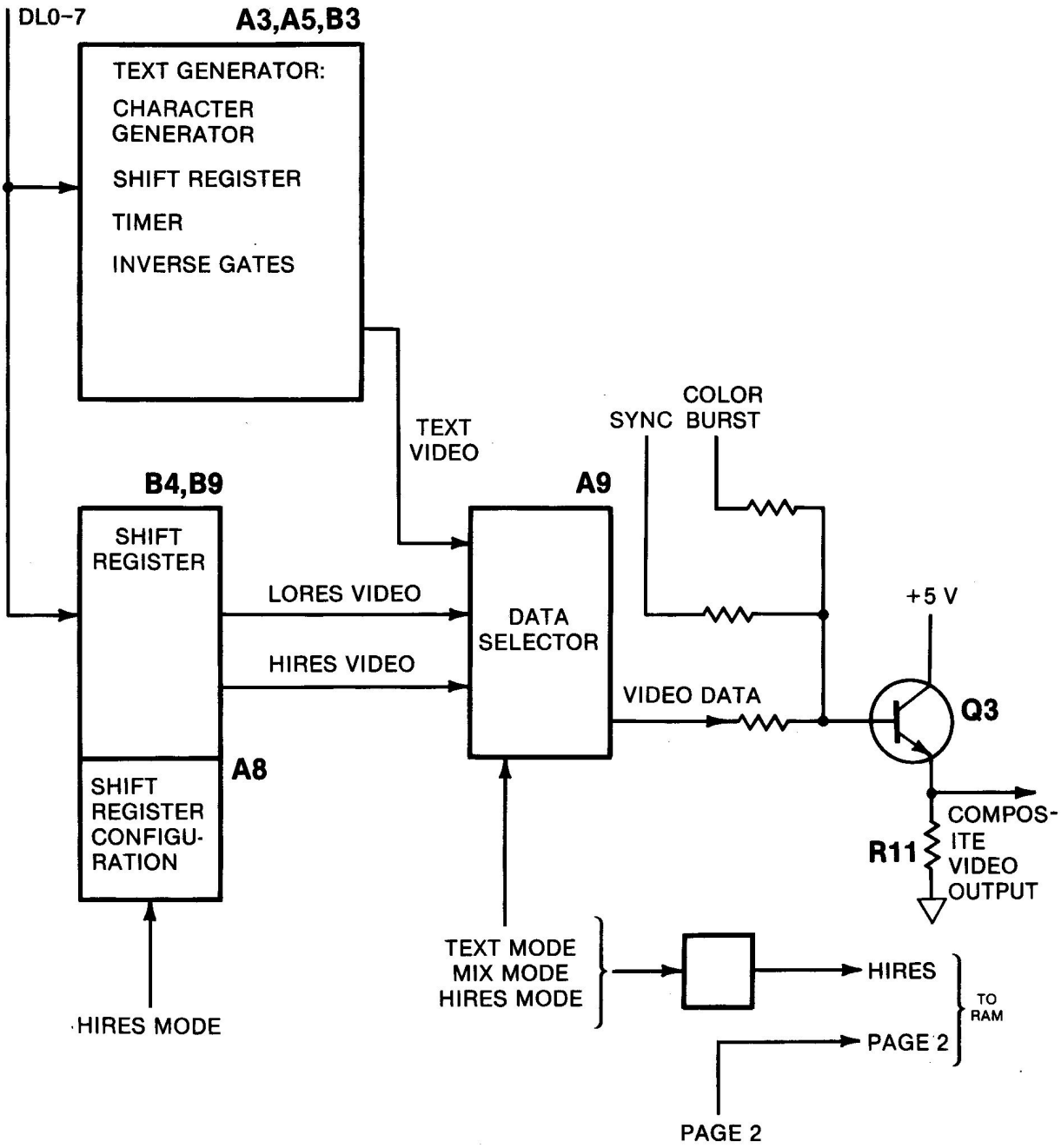


Fig. 8-2. Block diagram of video generator.

selector A9 to select either text video, LORES video, or HIRES video to output toward Q3. There, the video is mixed with SYNC and COLOR BURST to create the composite video output.

TEXT MODE, MIX MODE, and HIRES MODE are combined to generate the signal HIRES. HIRES and PAGE 2 map to memory as described in Chapter 5 and as summarized in Table 8-1.

The way in which data selectors A8 and A9 reconfigure the circuit under control of the soft switches is a key concept in understanding the video generator. Each configuration will be covered separately in the sections that follow.

Table 8-1. Summary of HIRES and Page 2 Map to Memory

PAGE 2	HIRES	Memory Space
0	0	Page 1 Text/LORES
0	1	Page 1 HIRES
1	0	Page 2 Text/LORES
1	1	Page 2 HIRES

Text

The following soft switch settings configure the video generator for All-Text:

TEXT MODE = 1
MIX MODE = X
PAGE 2 = 0 or 1
HIRES MODE = X
(X = Don't care)

Fig. 8-3 shows the block diagram for this configuration. The first block, A5, is a 2K by 8-bit ROM that has been programmed to contain the same ASCII character set as contained in a type 2513 character generator. Thus A5 takes the six low order latched data bits and interprets them as a set of 64 characters. This character set includes numerals, uppercase letters, and most ASCII symbols. It does not include lowercase letters and some remaining ASCII symbols. The output of A5 is a 5 by 7 dot matrix character (Fig. 8-4).

The ROM, A5, generates the character one row at a time. Each row in the character is displayed by adjacent horizontal scan lines on the crt. It takes eight scan lines to display a complete character (including a vertical space above each character). Of course these 8 scan lines also display the other 39 characters in a line of text. As the crt beam scans down the screen, the signals VA, VB, and VC increment at each line. This is shown in Fig. 8-4. Signals VA, VB, and VC address the appropriate contents of A5 on each new row.

At any particular row, the dots on that row are output in parallel by A5. Shift register A3 (Fig. 8-3) converts the dots from parallel to serial. This means that it shifts out the dots one at a time as the crt beam scans across the screen. For each character, shift register A3 shifts seven times. It first shifts out a 0 (low), then the five dots that make the character, then finally another 0. The two 0s are stored in A5 and are used to create the horizontal space between characters. This is seen in Fig. 8-4. The figure also shows the video waveform that is generated for the third row of the character "A."

Referring again to Fig. 8-3, flashing timer B3 provides the rate for flashing characters and the cursor. Gates B11-11 and B13-4 decode the high order two bits (DL6 and DL7) of each character to determine its display mode (see Table 8-2).

Table 8-2. DL6 and DL7 Determine Display Mode

DL7	DL6	Mode
0	0	Inverse
0	1	Flashing
1	0	Normal
1	1	Normal

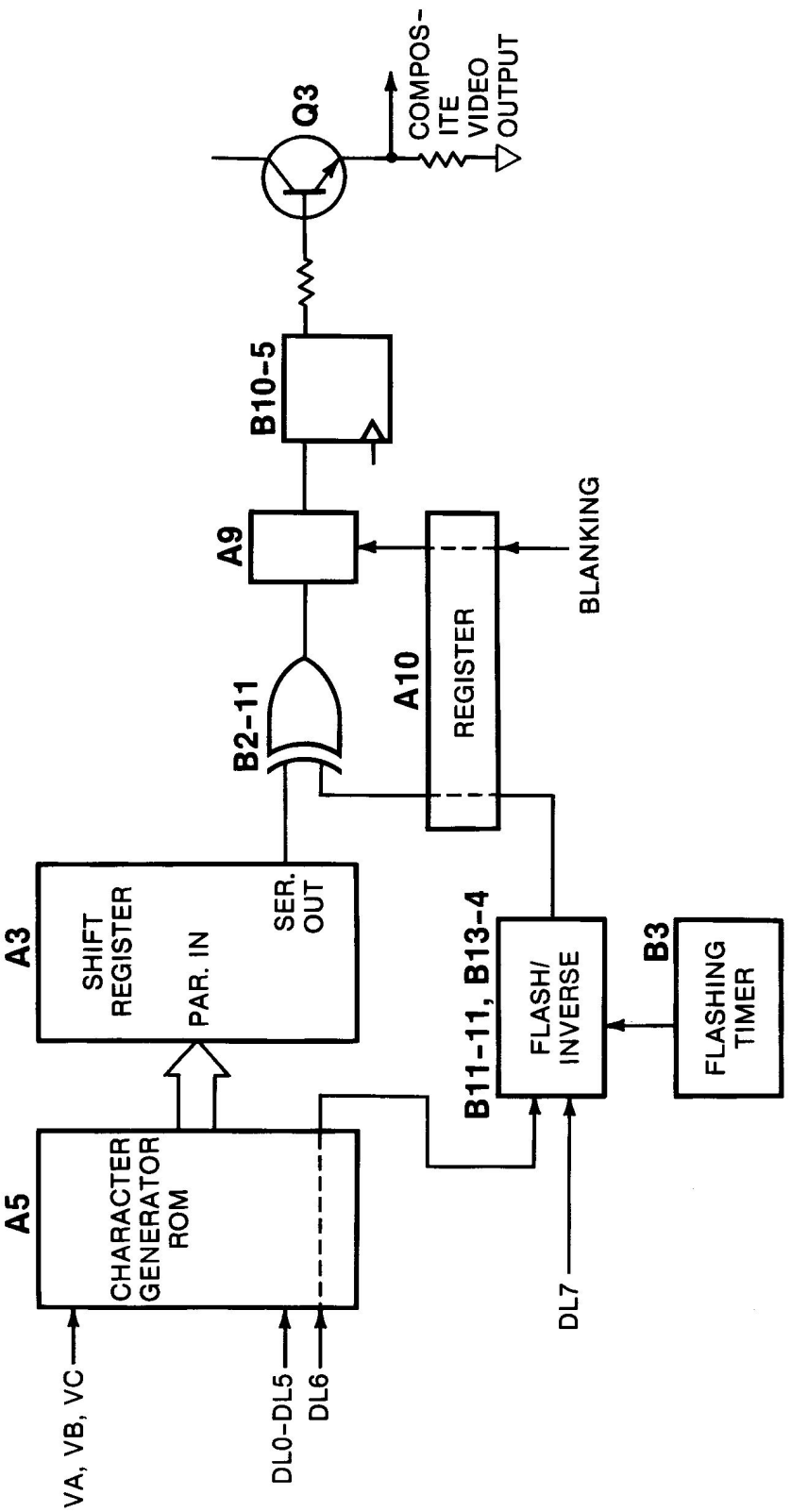


Fig. 8-3. Block diagram of text mode.

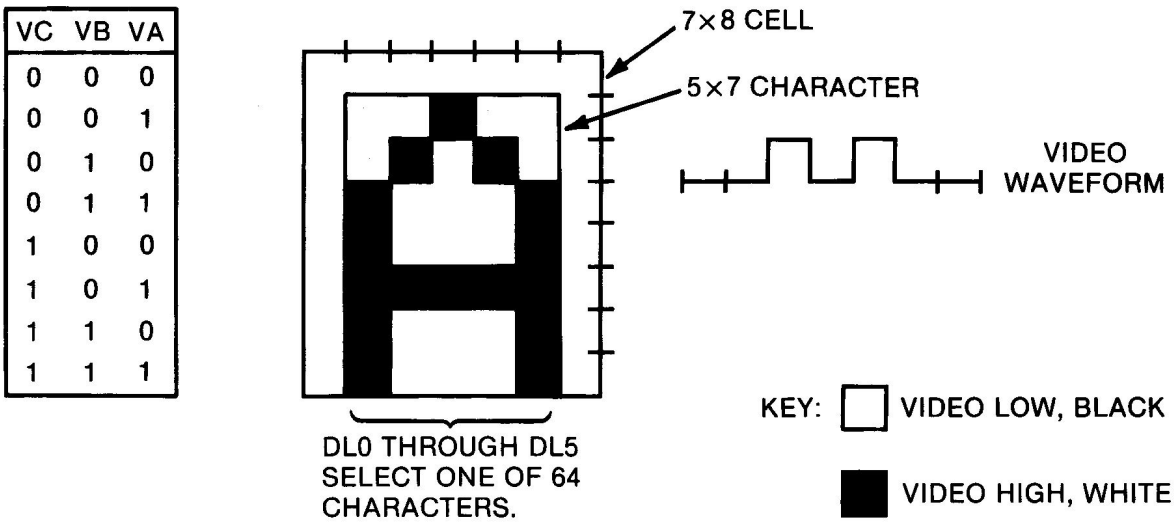


Fig. 8-4. Text character format.

Note that flashing characters are merely alternating quickly between normal and inverse.

The output of B13-4 contains the flash/inverse information for the current character. This information is stored in register A10, and is then presented to one input of exclusive OR gate B2-11. The exclusive OR gate inverts the serial data from the shift register under control of the flash/inverse signal. Fig. 8-5 shows the letter "A" as it would appear in inverse video. Note that the entire 7 by 8 cell is inverted, not just the 5 by 7 character. The video waveform for the third row in the cell is also shown in Fig. 8-5. The waveform is simply the inverse of the waveform from Fig. 8-4.

Returning to Fig. 8-3, the text video signal from B2-11 next passes through data selector A9. At A9, the video signal can be turned off by the blanking signal stored

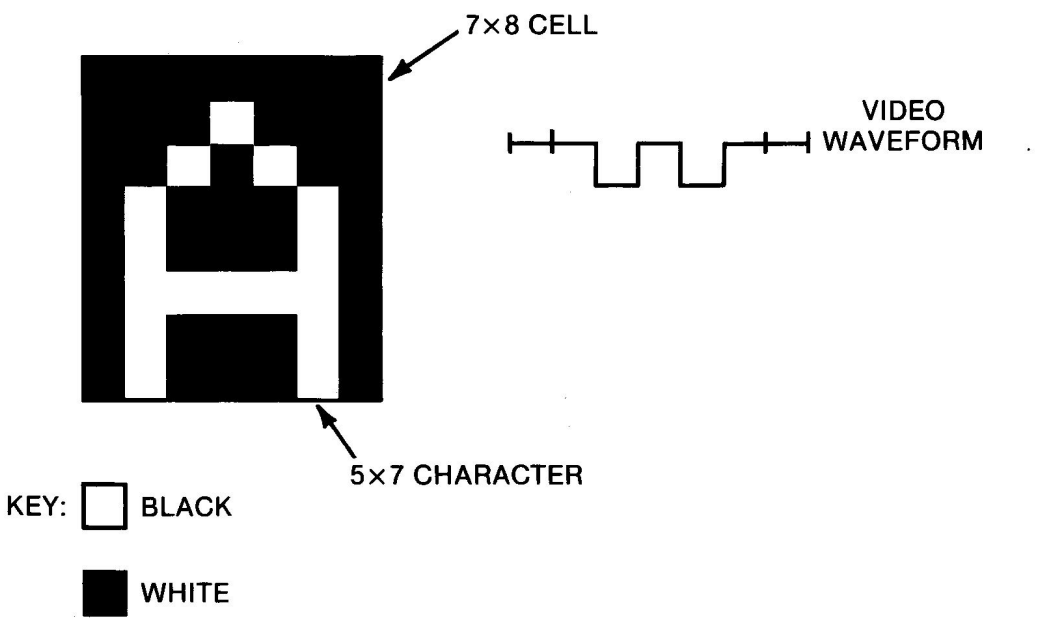


Fig. 8-5. Inverse text character.

in register A10. By this, we mean that during the blanking interval, the video signal is forced to 0 (low) to create a black portion of the screen. After A9, the video next passes through flip-flop B10-5 and then to Q3. At Q3, the video is mixed with SYNC and COLOR BURST to create the composite video output. (Blanking and the Q3 mixing process were described in Chapter 4.) This completes the path of the text characters through the video generator.

HIRES

The following soft switch settings configure the video generator for HIRES:

TEXT MODE = 0
MIX MODE = 0
PAGE 2 = 0 or 1
HIRES MODE = 1

Fig. 8-6 is the block diagram for HIRES mode. Data selector A8 configures 4-bit shift registers B4 and B9 into one long 8-bit shift register. Each byte to be displayed is loaded into the parallel input of this shift register. The contents are then shifted out one HIRES pixel at a time as the crt beam sweeps across the screen. If the byte contains green or violet pixels, the high order bit (DL7) will be 0, and data selector A9 will select the output of the shift register. If the byte contains blue or orange pixels, DL7 will be 1, and data selector A9 will select the output of flip-flop A11-9. The output of A11-9 is the shift register's output delayed by one 14M clock.

The output of data selector A9 passes through flip-flop B10-5 on its way to Q3. At Q3, it is mixed with SYNC and COLOR BURST to create the composite video output. Just as was the case in text mode, a blanking signal is applied to A9 to force the video to black during the blanking interval—DL7 and BLANKING are stored in register A10 on their way to A9. The key elements of Fig. 8-6 are the shift register

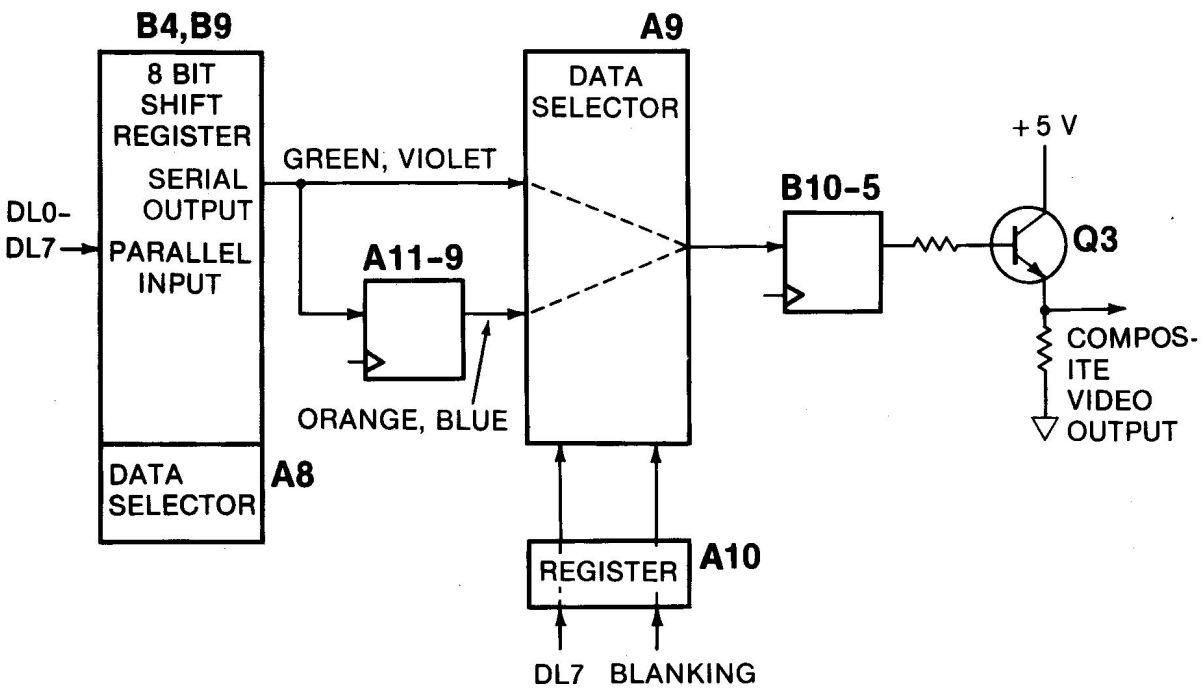


Fig. 8-6. Block diagram of HIRES generator.

and the ability of A9 to select the shift register's output either directly or delayed by A11-9. This selection is under control of DL7.

Now that we have the circuit arrangement in mind, let's see how it actually produces colors. Recall from color television theory that a video signal with a component at 3.58 MHz (the color burst frequency) will produce a color on the screen. The hue or tint of that color will be a function of the signal's phase relative to the phase of the color burst signal. (If you are not familiar with color video techniques, you should review Appendix A at this time.) Fig. 8-7 shows color hue as a function of phase angle relative to color burst.

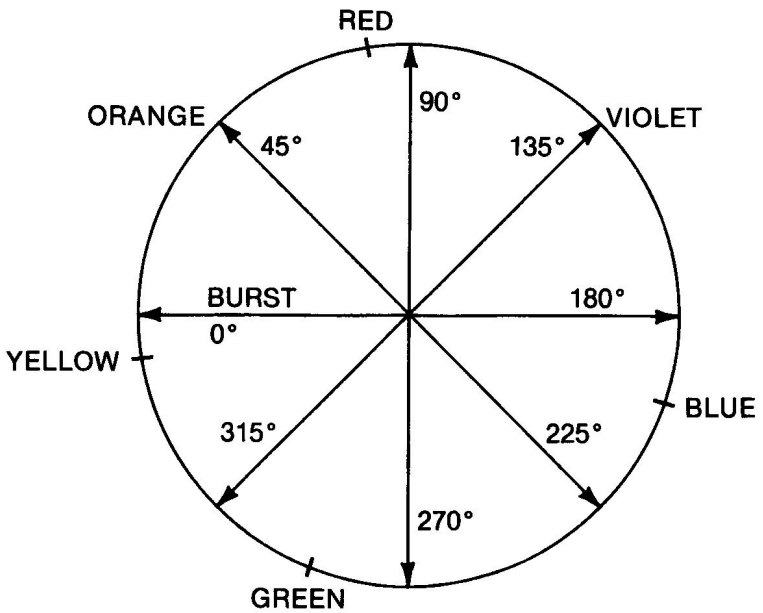


Fig. 8-7. Color hue as a function of phase angle relative to burst.

Let's take the case of one byte of violet in the upper left corner of the HIRES screen. We can do this with a "POKE 8192, 85" from BASIC. When the video scan is at the upper left of the screen, the decimal value 85 will load into shift register B4/B9 as shown in Fig. 8-8. Note that this value is an alternating pattern of 1s and 0s. The LSB (DL0) is immediately available (point A). The shift register then shifts six times, creating the video waveform shown, as the bits DL0 through DL6 shift out.

Above the video waveform we have shown COLOR BURST at the same time scale. Observe that the video waveform has the same frequency as COLOR BURST, but the video is delayed by 135 degrees. The color wheel (Fig. 8-7) shows us that a phase delay of 135 degrees from burst creates the color violet.

Fig. 8-9 shows how green is generated. This time we POKE 42 into location 8192. When the pattern shifts out of the shift register, we get the video waveform shown. Again it has the same frequency as COLOR BURST, but is delayed 315 degrees. This corresponds to the color green in Fig. 8-7.

Fig. 8-10 depicts the display method for blue. Here we POKE the value 213 into location 8192. The low order seven bits are the same as they were for violet. Thus, we get the same waveform at the shift register's output. The MSB (DL7) is 1, however, so data selector A9 selects the output of flip-flop A11-9. The flip-flop delays the shift register output by one 14M clock period. This is equivalent to 90 degrees at 3.58 MHz as shown in Fig. 8-10. The net result is to delay the video waveform 225 degrees relative to burst. This corresponds to the color blue in Fig. 8-7.

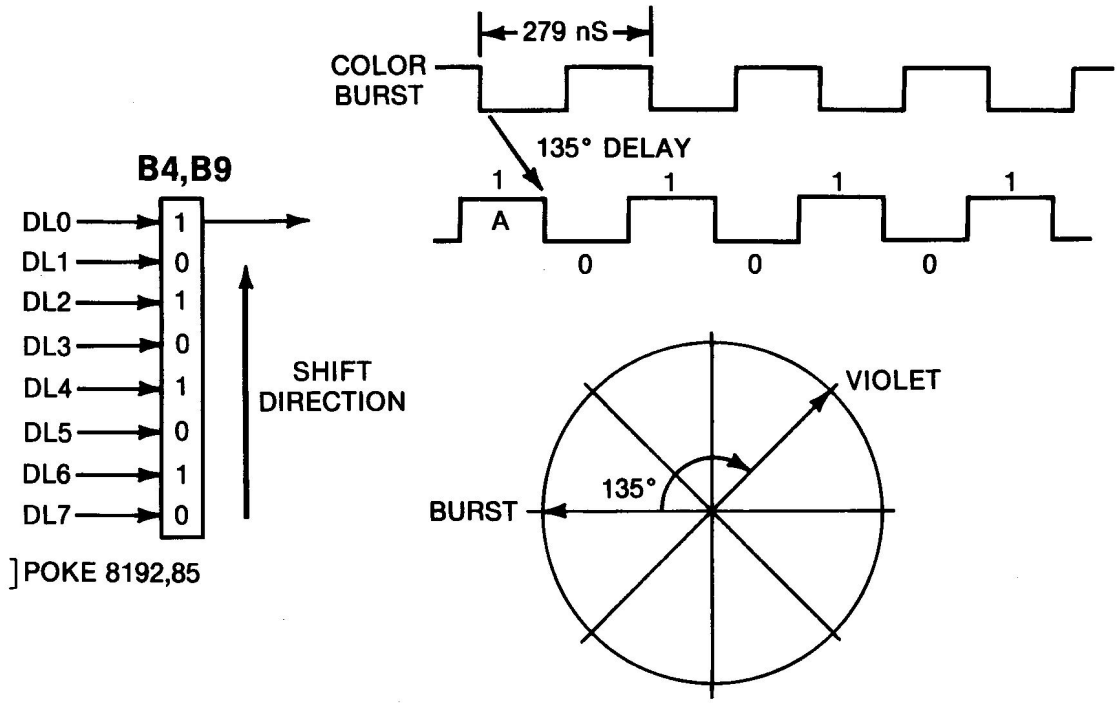


Fig. 8-8. HIRES violet.

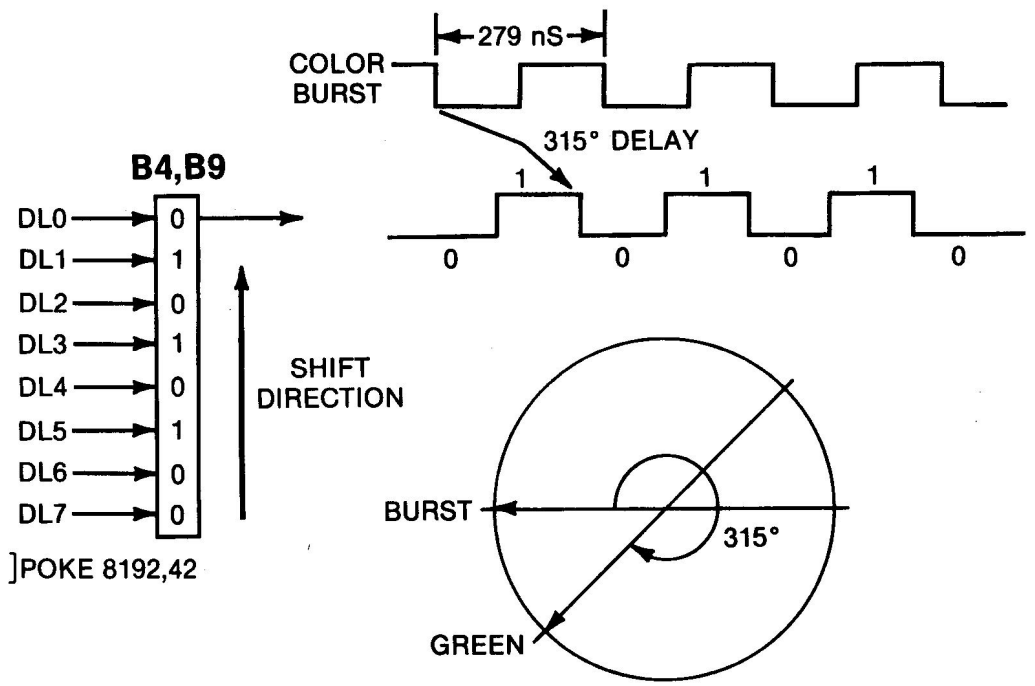


Fig. 8-9. HIRES green.

Finally, in Fig. 8-11, we show how orange is produced. We poke the value 170 into location 8192, and this generates the "green" waveform at the shift register's output. The MSB (DL7) is 1, so A9 selects the output from A11-9. A11-9 delays the "green" waveform by one 14M clock. The net result is a video waveform delayed 45 degrees from burst. This corresponds to the color orange in Fig. 8-7.

There now remain the two "colors" black and white. For black we simply store 0

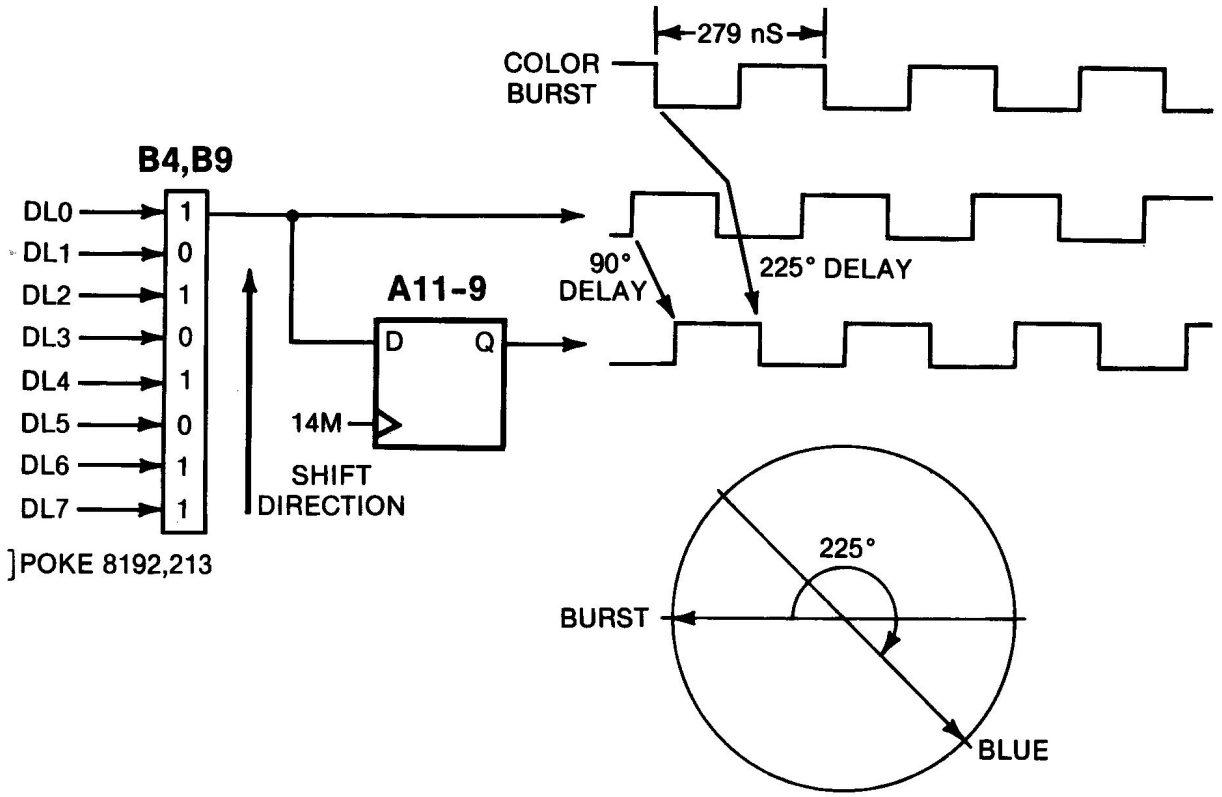


Fig. 8-10. HIRES blue.

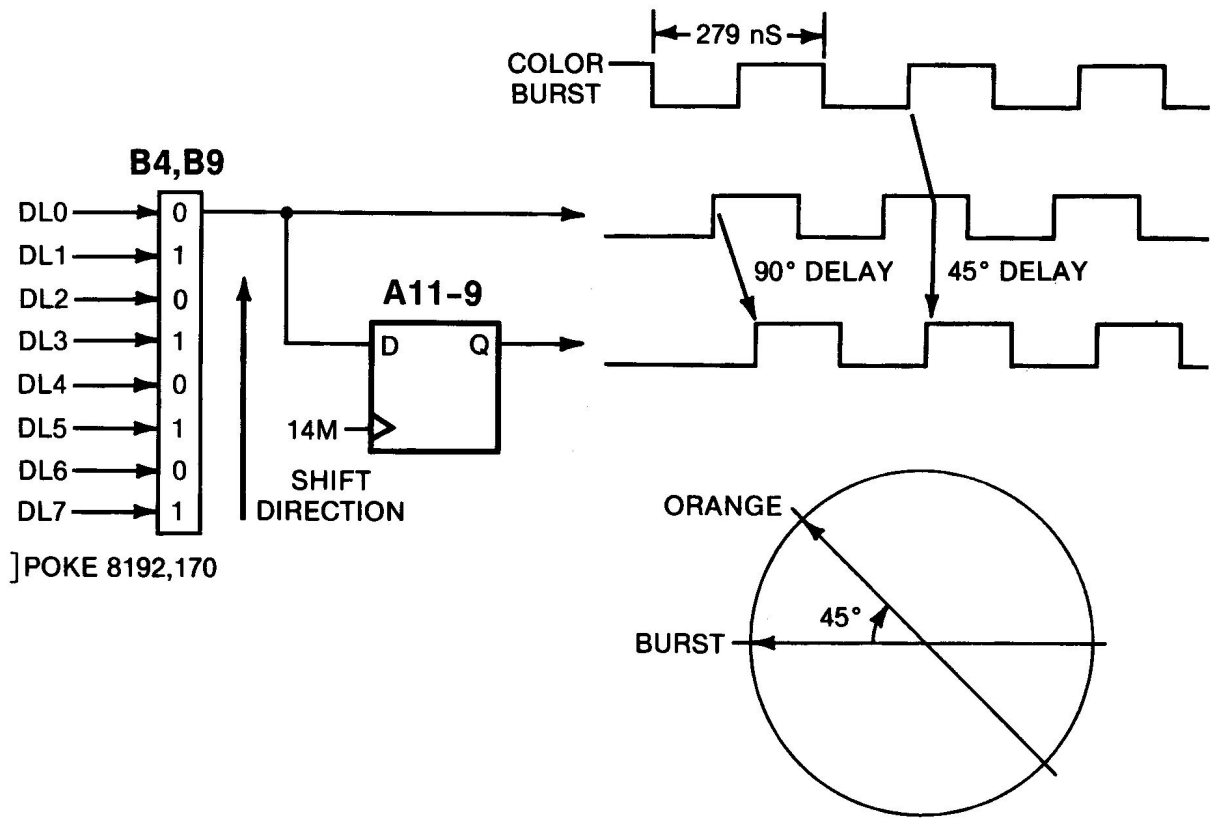


Fig. 8-11. HIRES orange.

or 128 in memory (Fig. 8-12). The seven 0s will shift out creating a continuous low signal. This corresponds to black. For white, we store 127 or 255 (Fig. 8-13). The seven 1s will shift out creating a continuous high signal. This signal has no component at 3.58 MHz, so there is no color. A high level corresponds to white, so that is what we get.

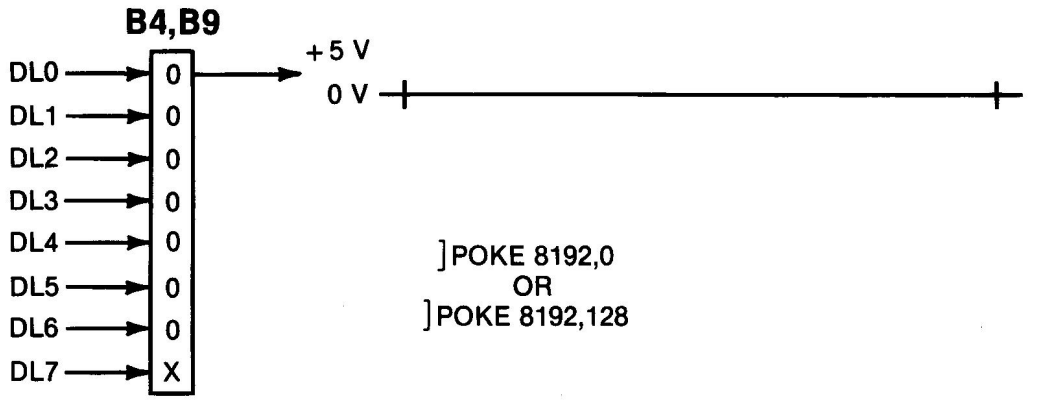


Fig. 8-12. HIRES black.

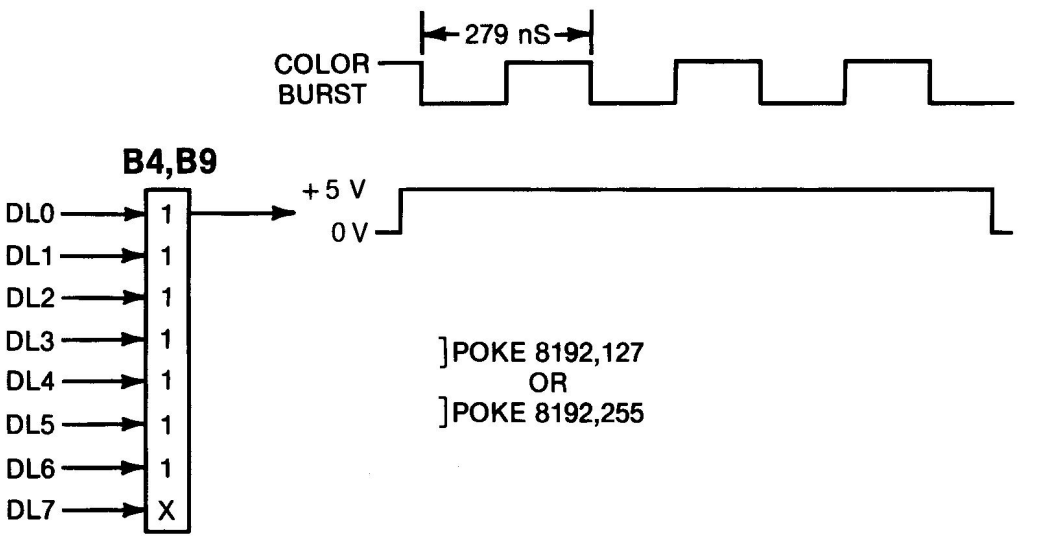


Fig. 8-13. HIRES white.

Before leaving our overview of the HIRES display, we must introduce the topic of even and odd bytes. The previous examples showing the six HIRES colors all used an even byte, location 8192. Fig. 8-14 shows what happens when we consider both even and odd bytes. At the top of the figure we have shown color burst. The second waveform is the video output for a violet line. It starts out just like the example in Fig. 8-8. The value 85 produces the alternating pattern 1010101.

The next segment of the line is stored in an odd location. If the line is to continue with the same color (same phase relative to burst), the odd segment must have the pattern 0101010. This is the value 42, the complement of 85. The same color is stored as *two different values,* depending on whether the byte is *even* or *odd.* This can be explained by noting that the time used to display one byte contains three and one-half cycles of 3.58 MHz. Thus, the relative phase of COLOR BURST changes by 180 degrees every byte. HIRES plotting routines take this into account.

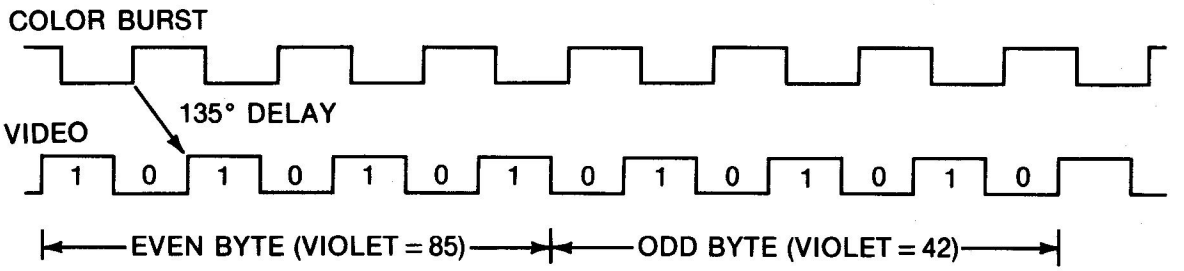


Fig. 8-14. Even and odd HIRES bytes.

We have now explained in concept how the HIRES colors are produced. There remain more details that will have to wait until later in this chapter.

LORES

The following soft switch settings configure the video generator for LORES.

TEXT MODE = 0
MIX MODE = 0
PAGE 2 = 0 or 1
HIRES MODE = 0

Fig. 8-15 shows the circuit configuration for this mode. Data selector A8 arranges shift registers B4 and B9 into two separate 4-bit shift registers. In LORES mode, each byte stores two pixels. Bits DL0–3 store the upper pixel; these bits parallel load into B4. Bits DL4–7 store the lower pixel and these bits parallel load into B9.

While the pixels are being displayed, the bits rotate in the shift registers in a manner similar to that of HIRES. In LORES, however, each shift register provides

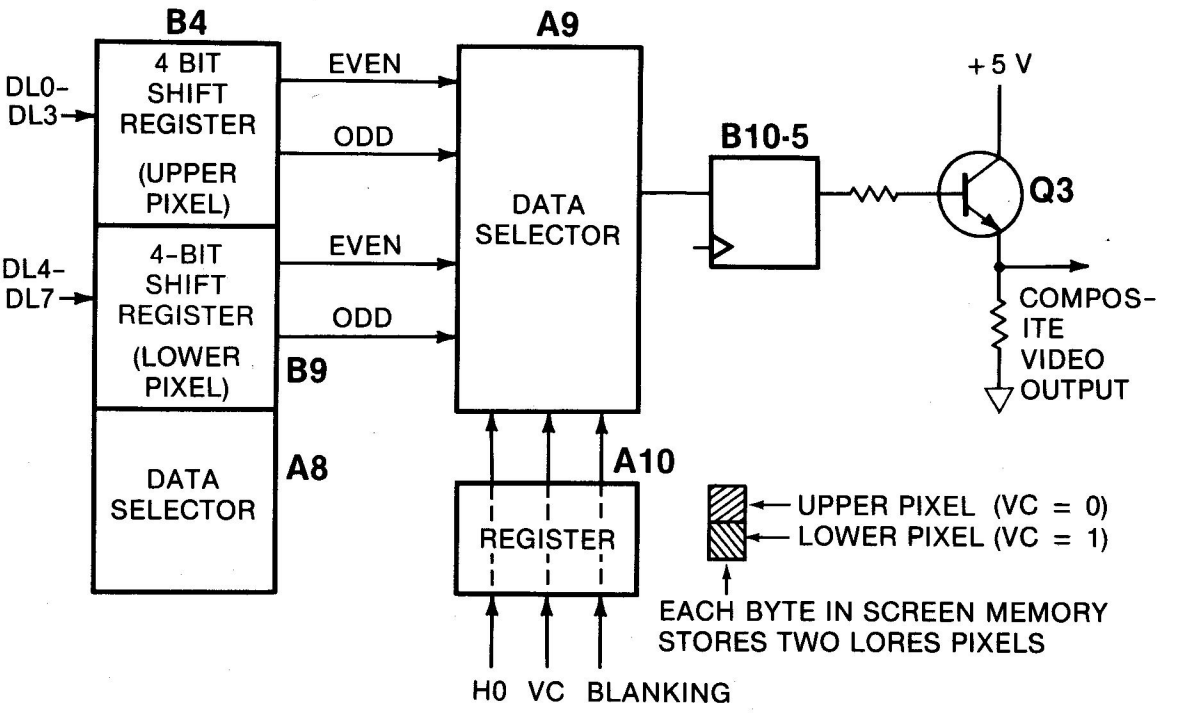


Fig. 8-15. Block diagram of LORES generator.

both an even and an odd output. Data selector A9 selects between even and odd outputs under control of signal H0. Signal H0 alternates between 0 and 1 as the crt beam scans over alternating even and odd bytes. You can refer to Fig. 3-4* to see the timing of H0. The purpose of this even/odd circuit configuration will be explained shortly.

Data selector A9 also selects between the upper and lower pixels under control of signal VC. Here you should refer to Fig. 4-1* to see the timing of VC. Signal VC alternates between 0 and 1 every four scan lines. The height of a LORES pixel is also four scan lines. When VC=0, A9 selects the upper pixel. When VC=1, A9 selects the lower pixel. This is depicted in Fig. 8-16 where we have shown signals VC and H0 coordinated with four LORES pixels.

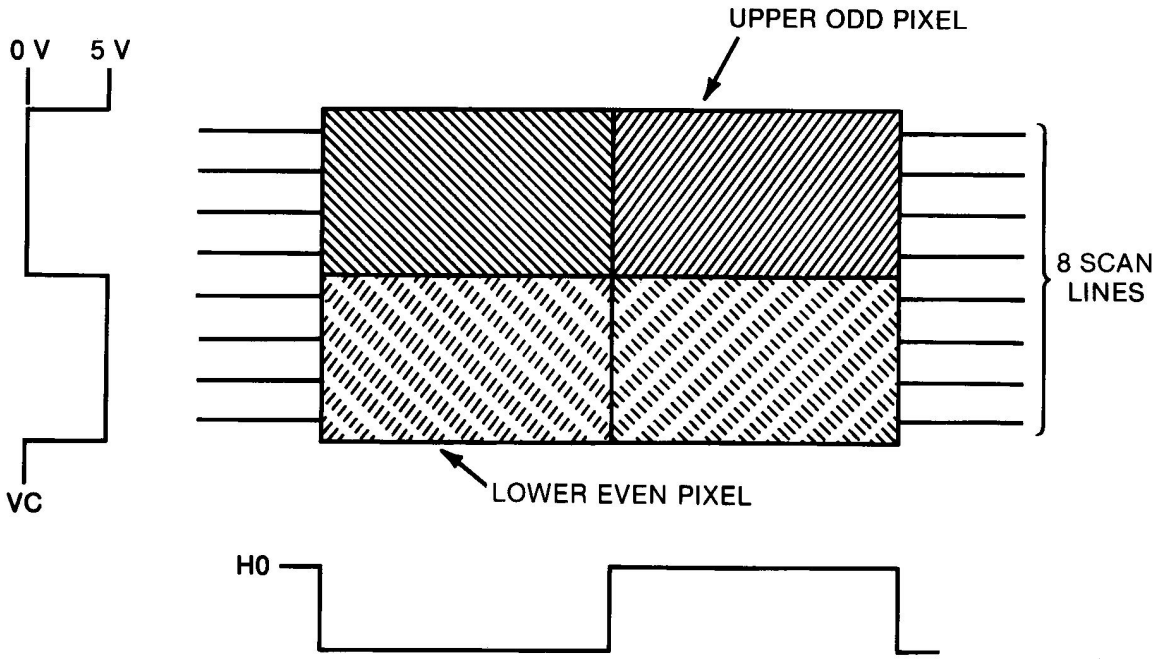


Fig. 8-16. LORES pixels.

The blanking signal at data selector A9 forces the video to black during the blanking interval. Signals H0, VC, and BLANKING are stored in register A10 before they go to A9. The video out of A9 passes through flip-flop B10-5 on its way to Q3. At Q3, the video is mixed with SYNC and COLOR BURST to create the composite video output.

Let's briefly review the LORES block diagram. The key elements are: (1) the shift registers with separate odd and even outputs, and (2) data selector A9 which selects upper and lower, even and odd inputs under control of VC and H0.

We now move from the block diagram to an explanation of LORES color generation. Each pixel is stored as four bits in memory. The pixel color is coded by these four bits as listed in Table 8-3. The table lists the color names used in Apple's documentation and alternative color names used in this book. The alternative names will aid our discussion, and you may assume that hereafter any LORES color mentioned in this book will use the alternative name.

In Table 8-4 the 16 colors are divided into four groups. We will examine one color from each of the first three groups and three colors from the miscellaneous group.

Table 8-3. LORES Colors

Bit Positions		Color Names	
Upper Pixel: 3 2 1 0 Lower Pixel: 7 6 5 4	Decimal	From *Apple II Reference Manual*	Alternative Names
0 0 0 0	0	Black	Black
0 0 0 1	1	Magenta	Dark Red
0 0 1 0	2	Dark Blue	Dark Blue
0 0 1 1	3	Purple	Violet
0 1 0 0	4	Dark Green	Dark Green
0 1 0 1	5	Gray 1	Gray 1
0 1 1 0	6	Medium Blue	Blue
0 1 1 1	7	Light Blue	Light Blue
1 0 0 0	8	Brown	Dark Orange
1 0 0 1	9	Orange	Orange
1 0 1 0	10	Gray 2	Gray 2
1 0 1 1	11	Pink	Light Red
1 1 0 0	12	Light Green	Green
1 1 0 1	13	Yellow	Light Orange
1 1 1 0	14	Aquamarine	Light Green
1 1 1 1	15	White	White

Table 8-4. LORES Groups

Medium Group	Light Group	Dark Group	Miscellaneous Group
9 Orange	13 Light Orange	8 Dark Orange	0 Black
3 Violet	11 Light Red	1 Dark Red	5 Gray 1
6 Blue	7 Light Blue	2 Dark Blue	10 Gray 2
12 Green	14 Light Green	4 Dark Green	15 White

Fig. 8-17 shows the generation of LORES blue. We start by poking a 6 into memory location 1024 (the upper left pixel on the screen). When the video address scans this pixel, the low order four bits load into shift register B4. Bit DL0 (a 0) is immediately available at the video output, point A. The shift register now shifts 13 times. On each shift, the serial output is loaded into the other end of the register. Thus, the same four bits *recirculate* through the shift register, generating the waveform shown.

COLOR BURST is drawn above the video waveform at the same time scale. Both signals have the same frequency, 3.58 MHz. We have drawn tic marks at the center of the positive portion of each signal so that we can measure their relative phase. As shown, the video signal is delayed 225 degrees relative to COLOR BURST. From Fig. 8-7 we see that 225 degrees corresponds to blue. You may also wish to note that the video signal for LORES blue is the same as that for HIRES blue, Fig. 8-10.

Fig. 8-18 shows the generation of LORES *light* blue. This time we POKE a 7 into location 1024. Again, when the upper left pixel is scanned, four bits load into the shift register then rotate to create a video waveform at the serial output. This waveform has the same frequency as burst and is delayed 180 degrees. From Fig. 8-7 we see that 180 degrees also corresponds to blue.

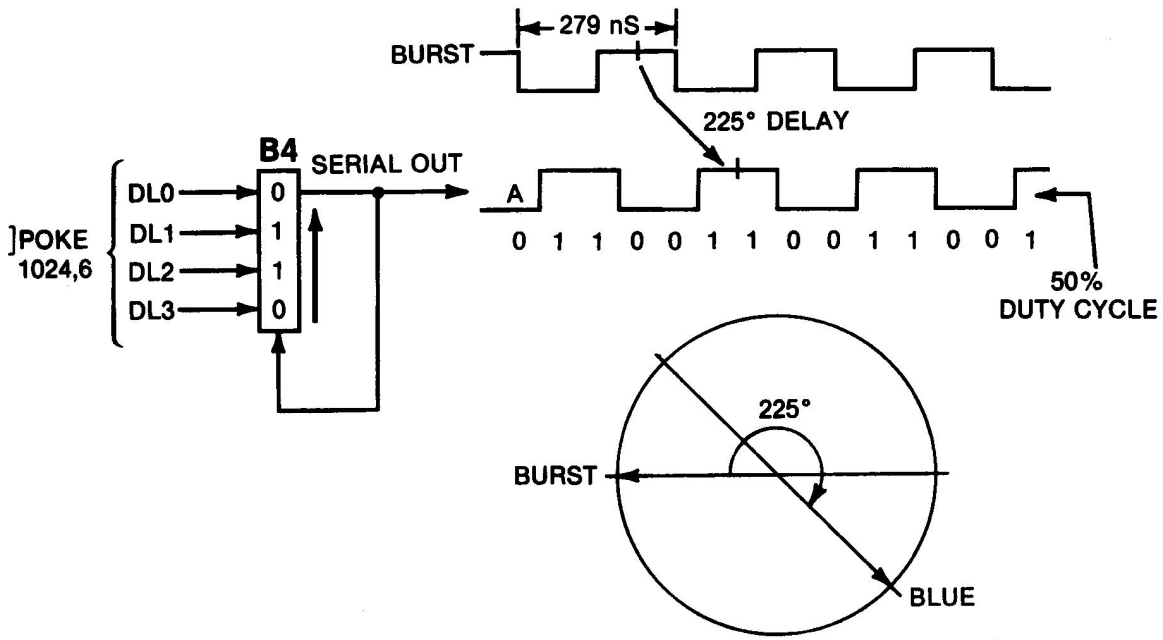


Fig. 8-17. LORES blue.

The color waveform in Fig. 8-18 differs from those presented up to now; it does not have a 50% duty cycle. This waveform is high for three out of four counts, so it has a duty cycle of 75%. Stated another way, its average value is 75% of the way from black (low video level) to white (high video level). This means the color display of this signal will be brighter than the display of the 50% duty cycle signal of Fig. 8-17. Thus, Fig. 8-17 shows "medium" blue and Fig. 8-18 shows "light" blue.

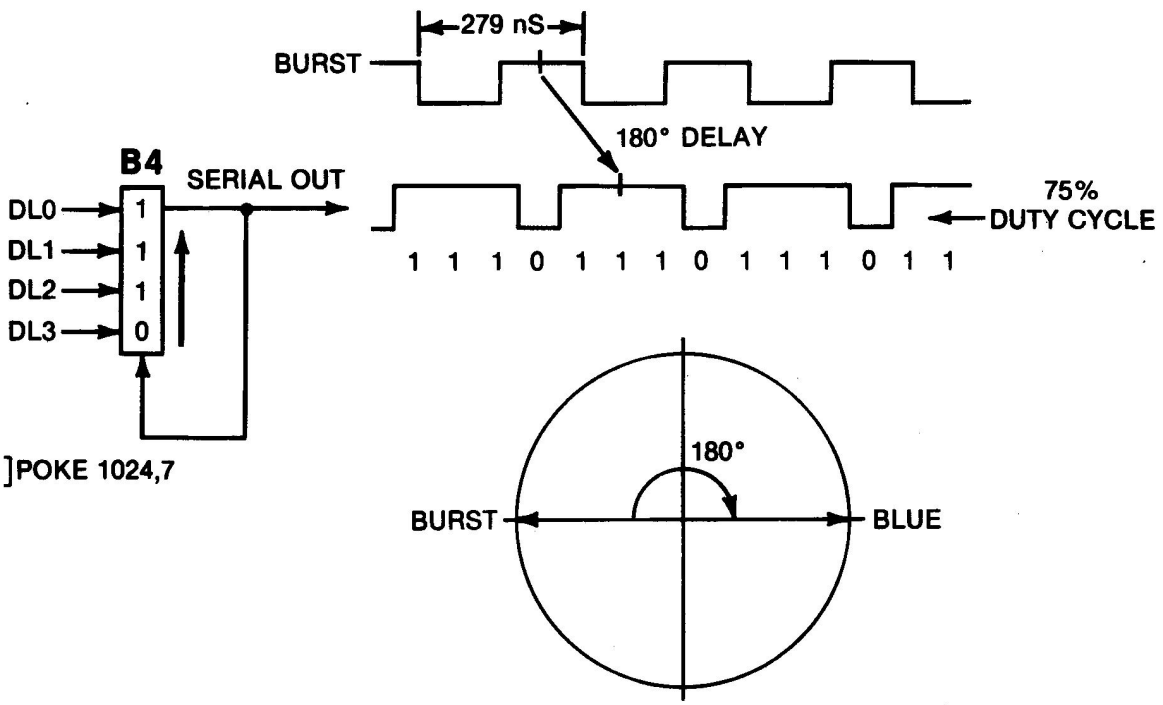


Fig. 8-18. LORES light blue.

Let's move to Fig. 8-19 which shows *dark* blue. This time we POKE a 2 into location 1024. The bits rotate and generate the waveform shown. This signal is delayed 180 degrees from burst, again blue. Note that this signal has a 25% duty cycle. Its average level is closer to black than to white. Thus, we get "dark" blue.

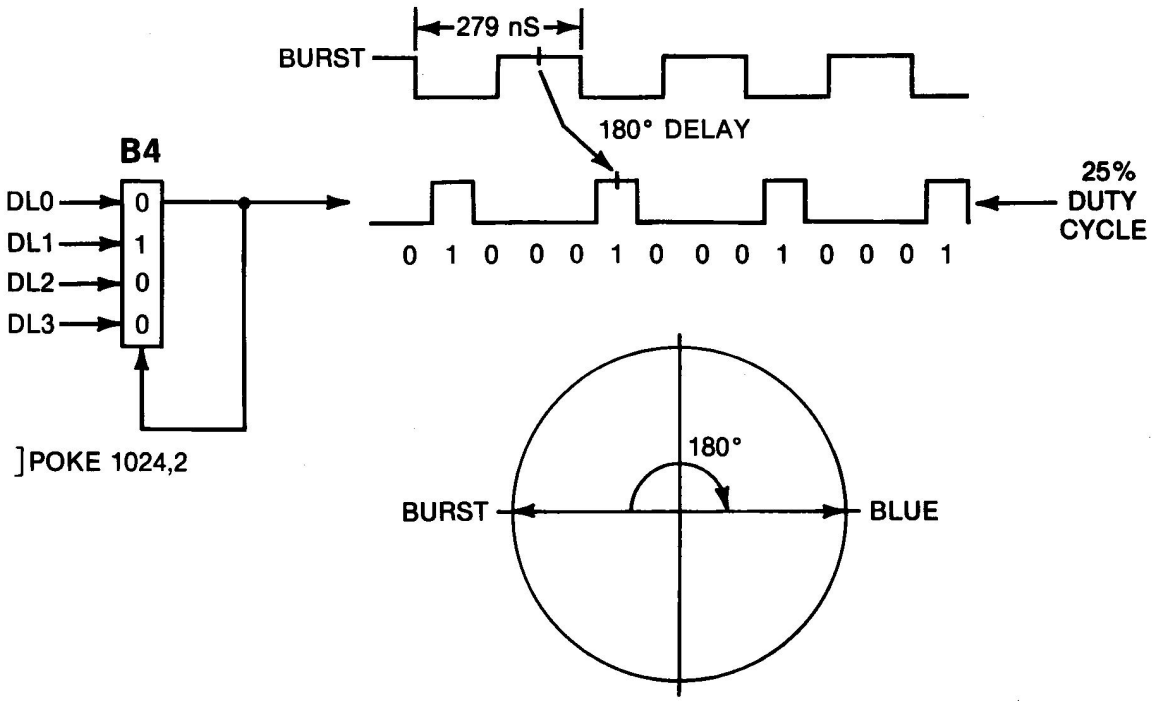


Fig. 8-19. LORES dark blue.

The other colors in the first three groups of Table 8-4 can be explained in a manner similar to the blue colors. The dark colors have duty cycles of 25%, the medium colors have duty cycles of 50%, and the light colors have duty cycles of 75%. The phase angles of the other hues will, of course, be different.

The black, white, and gray "colors" in the miscellaneous group contain no color since they do not contain a 3.58 MHz component. Black (all 0s) is a continuous low—a dc signal. White (all 1s) is a continuous high; another dc signal. A dc signal has no 3.58 MHz component, so we get no color, just black or white.

Gray is a little more interesting, and in Fig. 8-20 we show the generation of LORES gray 1. The value 5 is poked into location 1024. This is an alternating pattern, and when the bits rotate they produce the 50% duty cycle shown. This signal has a frequency twice that of COLOR BURST or 7.16 MHz. It is a square wave, and there will be harmonics above 7.16 MHz. However, there will be no components below 7.16 MHz. Specifically, there is no 3.58 MHz component. Without this component, there is no color. Since the duty cycle is 50%, the average level is half way between black and white—this is gray. Gray 2 is similar to gray 1; the two patterns are complements.

We have now shown in concept how six of the LORES colors are produced. These 6 are typical of the full set of 16 colors. The video waveforms and phase angles of *all* the LORES colors are summarized in Fig. 8-21.

Recall that in HIRES mode, data for the same color was stored as different values in even and odd bytes. You may also recall that we promised to explain the separate even and odd shift register outputs in the LORES block diagram, Fig. 8-15.

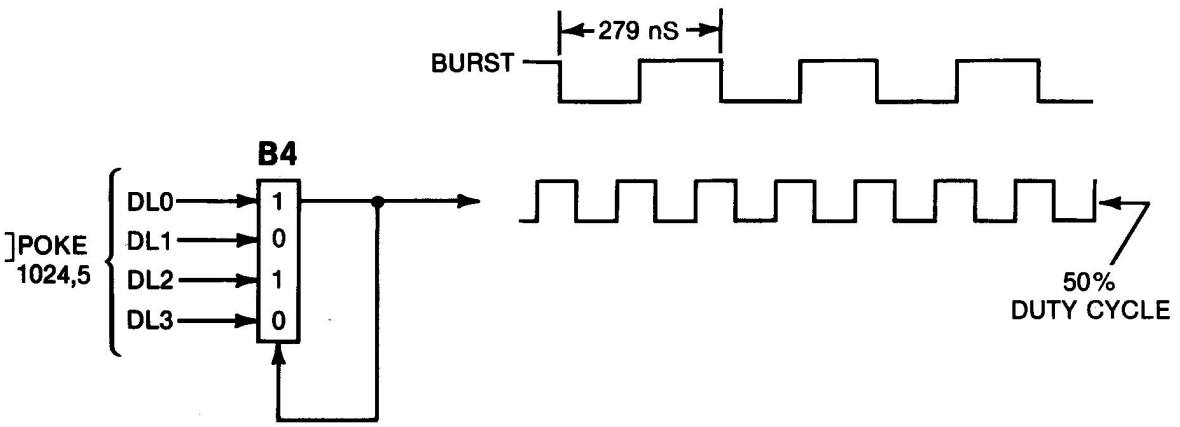


Fig. 8-20. LORES gray 1.

The Apple's method for handling the even/odd problem in LORES differs from the method used in HIRES. In HIRES, it is left to the software to sort out even and odd bytes. In LORES, the problem is solved in hardware. Fig. 8-22 shows the scheme, using blue as an example. First, data from an even byte is loaded into shift register B4. The data is decimal 6 which has the binary pattern 0110. The pattern rotates and generates the video waveform shown, starting at point A and ending at point B. This is the same waveform that was shown for the even byte blue of Fig. 8-17.

The next adjacent pixel is stored in an odd byte. Its display starts at point C. To continue the video signal at the same phase angle, the pattern starting at point C must be 1001. Here is how it works. We store the same decimal value (6) in the odd byte. When the pixel is scanned, the 6 loads into B4. Since this is an odd byte, signal H0 causes data selector A9 to select the "odd" tap of shift register B4. When B4 is loaded, the 1 at the odd tap is immediately available at the video signal, point C. The bits then rotate from this starting point, giving us a 1001 pattern.

In summary, A9 selects even and odd outputs of B4, allowing one data value to represent the same LORES color for both even and odd bytes. Since this is handled in the hardware, the programmer need not worry about it.

Our overview of LORES graphics has used shift register B4 as an example. The operation of shift register B9 is the same as that of B4. B9, however, operates on the high order four bits of data, and displays the lower pixel.

Mixed HIRES and Text

The following soft switch settings configure the video generator for mixed HIRES and text:

TEXT MODE =0
MIX MODE =1
PAGE 2 =0 or 1
HIRES MODE =1

Note that both the text and HIRES will come from the same page (determined by the PAGE 2 soft switch). Fig. 8-23 shows the block diagram for this configuration. Both the text generator and the 8-bit shift register operate on the latched data as previously described. While the crt scan is in the top portion of the screen, data selector A9 selects the HIRES from the 8-bit shift register. When the scan gets five-sixths of

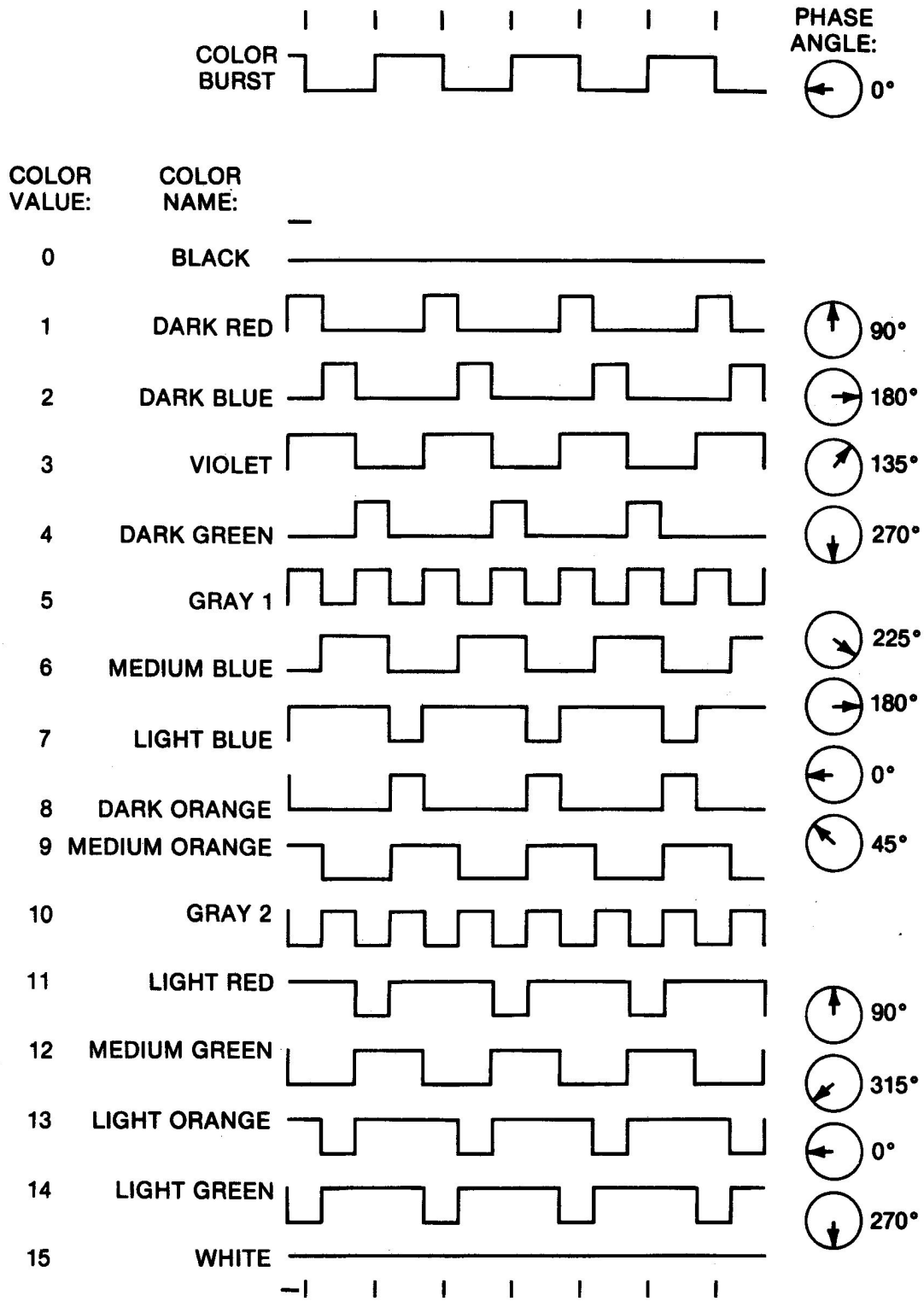


Fig. 8-21. LORES colors.

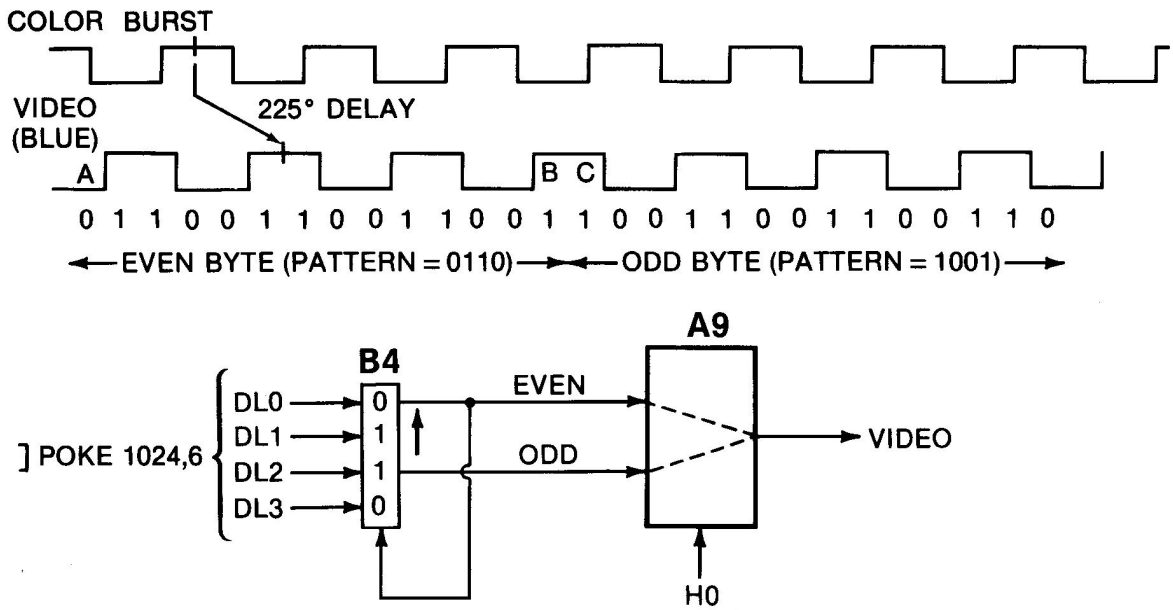


Fig. 8-22. Even and odd LORES bytes.

the way down the screen, video address lines V2 and V4 both go high. This causes A9 to select the text generator output. As a result, the bottom one-sixth of the screen contains four lines of text.

Mixed LORES and Text

The following soft switch settings configure the video generator for mixed LORES and text:

TEXT MODE	=0
MIX MODE	=1
PAGE 2	=0 or 1
HIRES MODE	=0

Both the text and LORES will come from the same page. Fig. 8-24 shows the block diagram for this configuration. Both the text generator and the two shift registers operate on the latched data. While the crt scan is in the top portion of the screen, data selector A9 selects LORES pixels from the shift registers. When the scan gets five-sixths of the way down the screen, video address lines V2 and V4 both go high. This causes A9 to select the text generator output. As a result, the bottom one-sixth of the screen contains four lines of text.

This concludes the overview of the Apple II video display. The block diagrams and signal waveforms in this section were drawn to illustrate only the concepts. As a result, some of the drawings may not correspond exactly to the actual circuit implementation or to waveforms actually observed on an oscilloscope. These details will be provided in the circuit analysis that follows.

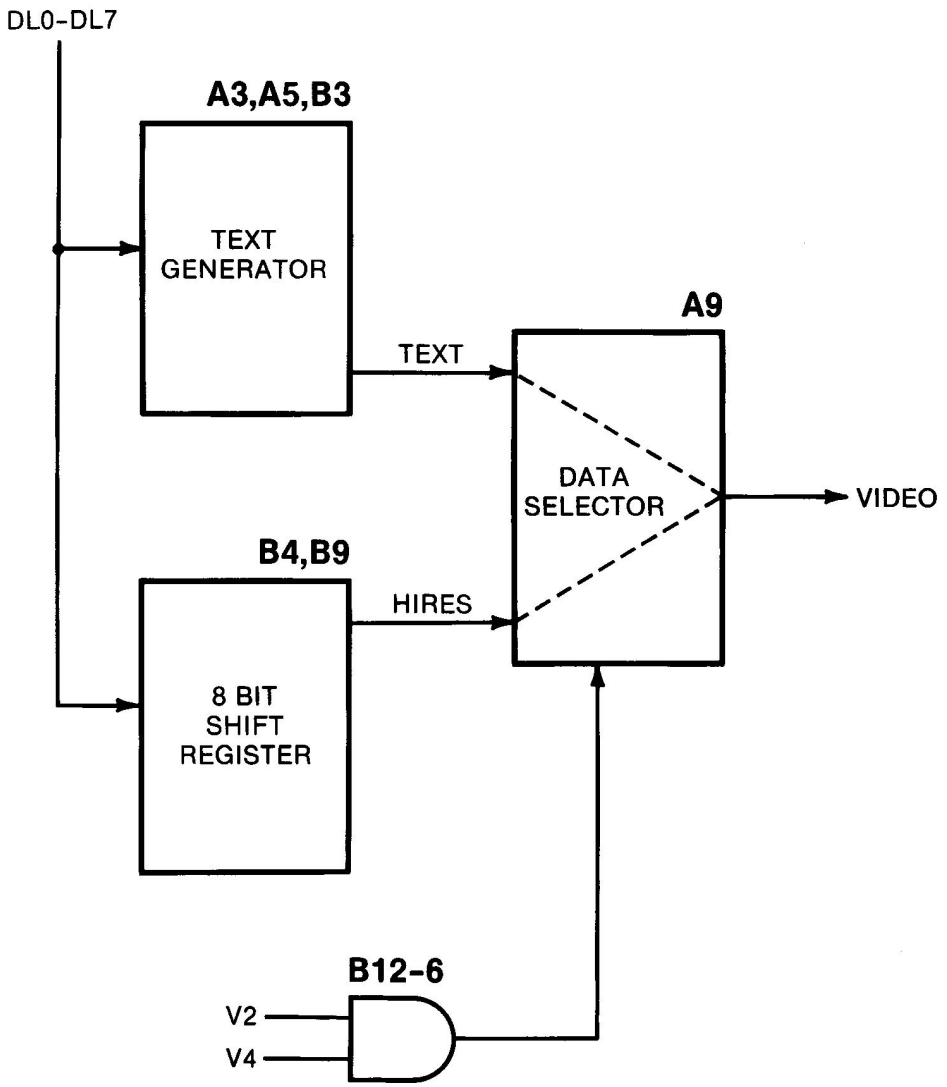


Fig. 8-23. Block diagram of mixed HIRES and text.

DETAILED CIRCUIT ANALYSIS

Text Mode

Our detailed analysis of text mode starts with character generator A5 and shift register A3 (Fig. C-16*). We then follow the signal through data selector A9 and flip-flop B10-5.

Character Generator—IC A5 is a type 2316B ROM that stores all the row and column dots for a set of 64 dot matrix characters. Inputs A3 through A8 are the character address. They receive latched data bits DL0 through DL5. These are the six bits of each byte in text memory of the Apple that store the character. Inputs A0, A1, and A2 of IC A5 are the row address. They receive the video address lines VA, VB, and VC. As the crt beam moves down the screen, these address lines select one of the eight rows that make up each character.

Input A9 of IC A5 connects to DL6. Character generator A5 is programmed so

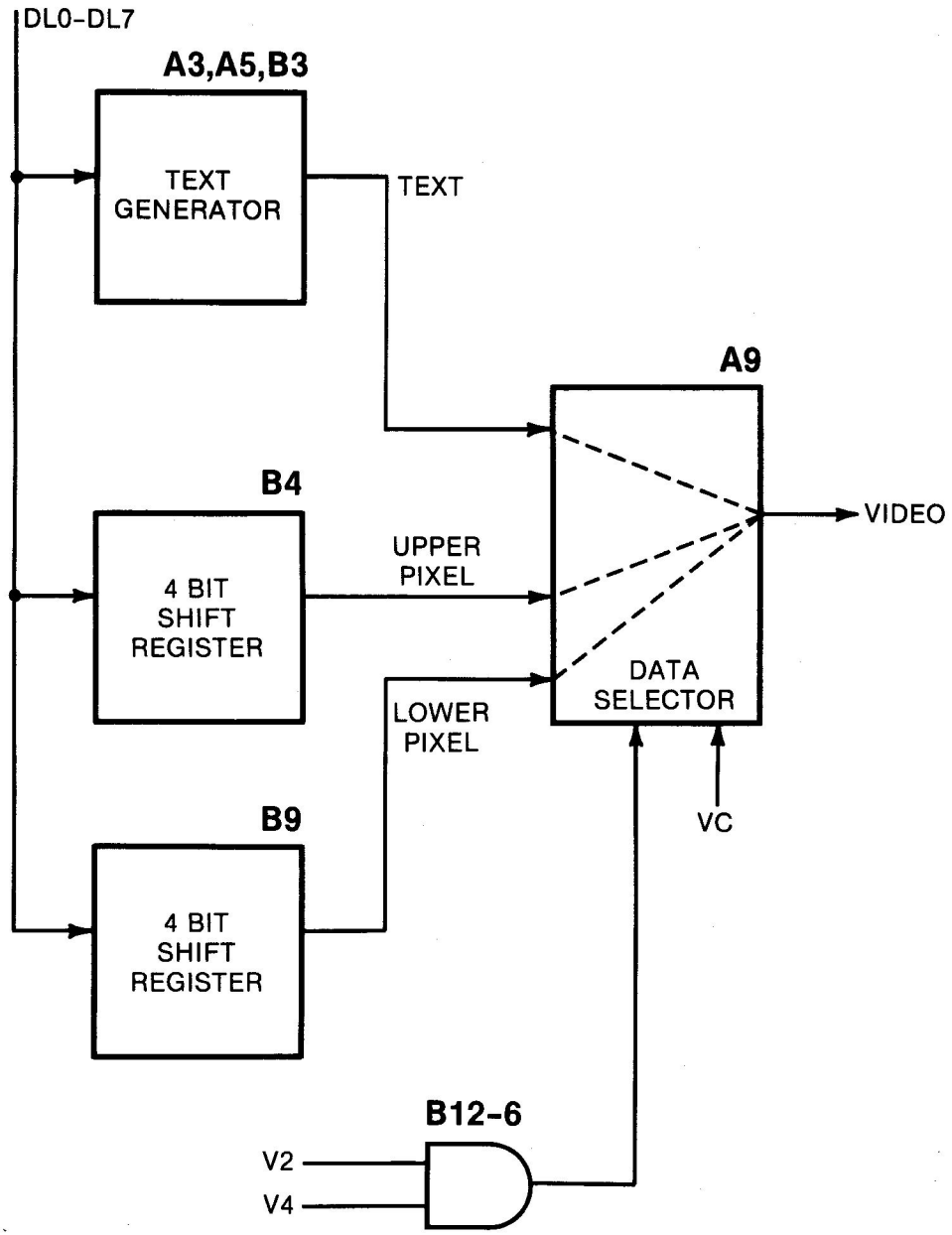


Fig. 8-24. Block diagram of mixed LORES and text.

that output O8 exactly follows address input A9. The effective result is to connect DL6 to B11-12.

Input A10 of IC A5 connects to DL7. A5 is programmed such that input A10 is a don't care.

As we have just described, the connection of DL6 and DL7 to A5 has no effect with the ROM supplied by Apple Computer. The purpose of the connections is so that you can provide your own ROM or EPROM containing additional characters. The additional characters would be accessed via DL6 and DL7. Additional flexibility is provided by bow ties 9, 10, and 11. These may be cut and connections made to A5 inputs A9 and A10 from external sources, such as the game I/O or a simple switch. The external source could then select a character set via A5 inputs A9 and A10. Specific implementation of user character sets is not covered in this book.

The Apple supplied A5 ROM is programmed so that outputs O1 and O7 are always low. This provides a blank dot on the left and right of each character. The result is the horizontal space between characters. In a user supplied ROM, these blank dots may be programmed to form part of a character. An example would be a graphics symbol.

Shift register A3 takes the parallel output of A5 and shifts it out one bit (dot) at a time. The resulting serial bit stream passes through exclusive OR gate B2-11 where it can be inverted to create inverse video. The inversion is controlled by the two latched data bits, DL6 and DL7. These two bits are decoded by gates B11-11 and B13-4. The resulting signal is applied to input pin 6 of register A10 where it is stored. The signal exits A10 on pin 12. The inverse signal has to be stored in A10 because DL6 and DL7 do not remain valid for the full character display time.

Normal and Inverse Text—When DL7 is high, B13-4 and A10-12 will be low. The resulting low at B2-12 allows the serial bit stream to pass noninverted. This gives us a character in *normal* video (white on black). Note that when DL7 is high, DL6 is a don't care. When DL6 and DL7 are both low, B13-4 and A10-12 will be high. This high at B2-12 inverts the serial bit stream, giving us a character in *inverse* video (black on white).

Flashing Text—When DL7 is low and DL6 is high, the signal at B11-11, B13-4, and A10-12 will follow the output of timer B3. B2-11 now alternately inverts or passes noninverted the serial bit stream. This gives us a *flashing* character at about 2 Hz.

Text Mode Circuit Configuration—Before moving on to the detail timing, we must explain how data selectors A8 and A9 configure the video generator for text. The TEXT MODE soft switch will be set, so B13-12 is high. NOR gate B13-13 is low, and after three $\overline{\text{RAS}}$ pulses, this low propagates through flip-flops B5-2 and B8 to make B8-2 low. Thus B11-6 (HIRES) is low, assuring that we will fetch characters from the text page and not the HIRES page.

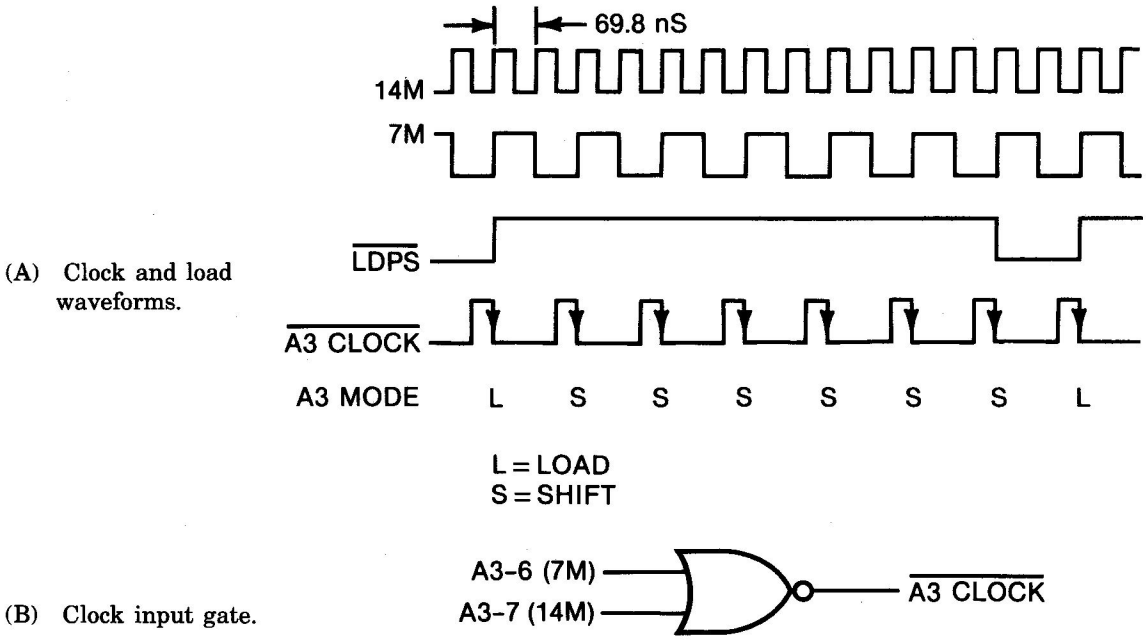
NOR gate A12-10 is high, A12-13 is low, and A11-6 is high. (A11-5 is used as an inverter.) This high at data selector A8 causes it to select its "1" inputs. Pin B5-2 is low, and this low passes through A8 to A8-7. The low is then stored in A10 on 14M rising while LD194 is high. The low next appears at A9-10. Recall that A12-13 is low. This low appears at A9-9.

If you have been following this discussion carefully, you will notice that A9 is now set up to select either input pin 3 or 4. These pins both connect to the video text from B2-11.

Shift Register A3—As we ease into the details of text generation, let's explore the clock and load inputs of shift register A3. These three inputs are driven from signals previously discussed and shown again at the top of Fig. 8-25A. Pins 6 and 7 are connected internally to a NOR gate that provides the IC's internal clock. The device is clocked on the falling edge as shown in the fourth line of Fig. 8-25A. On each clock, the IC either shifts (if pin 15 is high) or loads parallel data (if pin 15 is low). Signal $\overline{\text{LDPS}}$ controls the mode (shift or load) of A3.

Text Mode Timing Diagram—Fig. 8-26* shows the display timing of two text characters. Signals 14M and $\overline{\text{A3 CLOCK}}$ are shown at the top. The mode of A3 is shown next to each falling edge of $\overline{\text{A3 CLOCK}}$ (S = shift, L = load). (Remember that $\overline{\text{A3 CLOCK}}$ is an internal IC signal and cannot be viewed on an oscilloscope.) Also shown in the figure are LD194, the video address, and the latched data—all previously discussed.

The first character that we have shown is the left-most character on a line,



(A) Clock and load waveforms.

(B) Clock input gate.

Fig. 8-25. Clock and load inputs—shift register A3.

character 0. The second is the right-most character on a line, character 39. The 0 is an even character and 39 is an odd character. To add even more variety, we have made character 0 be in normal video and character 39 be in inverse video. We have drawn the timing as if these were the only two characters on a line. This is so the timing relationship between adjacent characters can be shown. There are, of course, really 40 characters per line.

Note that Fig. 8-26* shows one horizontal scan line or one row through the characters. The uppercase letters in the figure are references to the text (our normal practice in this book). The lowercase letters represent the data values of the waveform at various points. The seven signals A3-Q_A through A3-Q_G are shown as aids to understanding the timing. They are not available externally to IC A3.

Even Character Timing—At point A, the video address for character 0 becomes valid. Since we are at the left edge of the display, the video address is decoded to remove blanking at point B. After the video address maps to memory, the character appears as latched data at point C. During the time that the video address and the latched data are both valid (point D), VA, VB, VC, and DL0 through DL5 all address character generator A5 (Fig. C-16). Character generator A5 responds with seven horizontal dots on its outputs O1 through O7. On $\overline{\text{A3 CLOCK}}$ falling (point E in Fig. 8-26*), these seven bits load into shift register A3 (points F through M). The values of these seven bits are represented in Fig. 8-26* as b through h (corresponding to their 74166 signal names).

Also while the latched data is valid, DL6 and DL7 are decoded by B11-11 and B13-4 to become the inverse signal. It is loaded into A3 at point N. It is also loaded into register A10 on 14M rising pulse while LD194 is high, point O. The signal at A10-12 is low, so the output of A3 passes to B2-11 noninverted, point P. BLANKING (C14-6 in Fig. C-20*) is now low, so on 14M rising pulse while LD194 is high, A10-13 goes low (point Q in Fig. 8-26*). This enables data selector A9 (Fig. C-16*) via its pin 7. We already know that A9 is set to select input pin 3 or 4. Thus when enabled, A9 sends the text to B10-2. This flip-flop is clocked every 14M rising pulse, so the text

appears at the video output at point R in Fig. 8-26*. This first dot is actually a dark dot to the left of the character, so B10-5 is low. Each dot is 140 nS wide.

The purpose of flip-flop B10-5 is to *reclock* the video data. This is so that any glitches created by the preceding logic are eliminated. After the delay through B10-5, the video data will also have the correct phase relative to COLOR BURST. This is important for the color graphics modes.

On the next $\overline{\text{A3 CLOCK}}$ (point T), A3 is in shift mode and the eight bits stored in A3 shift as shown in the figure. Bit b shifts into Q_C, bit c shifts into Q_D, etc. The inverse signal at A10-12 shifts into A3-Q_A, and bit g shifts into Q_H. On the next 14M rising, bit g appears at the video output (point U). This process continues for four more shifts, giving us bits f, e, d, and c at the video output. One more shift gives us a dark dot to the right of the character, point V.

The connections at A3-1 and A3-2 do nothing. This is because A3 is always loaded with new data before the "a" bit shifts out.

Odd Character Timing—While the even character is shifting out, the video address and latched data for the odd character become valid (point W). Again, A3 is loaded with five character dots with a dark (low) dot on each side. This character is in inverse, however, so a high from B13-4 loads into A10-12 (point X). This high at B2-12 inverts the serial bit stream starting at point Y. This high appears at the video output starting at point Z. The high creates a lighted dot to the left of the inverse character. On the next five shifts of A3, the bits g through c appear inverted at the video output. One more shift creates a lighted dot to the right of the character, point AA.

Since this is the right-most character on the line (character 39), BLANKING goes high when the video address changes (point BB). On the next 14M rising pulse while LD194 is high, A10-13 goes high (point CC). This disables A9, and horizontal blanking starts after one more 14M clock pulse (point DD). This concludes our detailed analysis of text mode.

HIRES Mode

Our analysis of HIRES mode starts with the configuration of the video generator by the soft switches. Then we follow in detail the timing of the signals through the circuitry.

HIRES Mode Circuit Configuration—For HIRES graphics, the TEXT MODE and MIX MODE soft switches are low. This results in a high at B13-13 (Fig. C-16*). After three $\overline{\text{RAS}}$ pulses, this high propagates through flip-flops B5-2 and B8. This makes B11-5 high. Since the HIRES MODE soft switch is high at B11-4, B11-6 (HIRES) is high. This arranges the memory system to map video addresses to the HIRES page.

HIRES MODE high at A12-12 results in a high at A11-6. (Recall that A11-5 is used as an inverter.) This high at data selector A8 pin 1 causes it to select its "1" inputs. Thus $\overline{\text{7M}}$ at A8-10 passes through to A8-9 and pin 10 of shift register B4 and B9. Pins 9 and 11 of B4 and B9 connect to LD194 and 14M. Pin 11 is the clock input and pins 9 and 10 control the shift modes of B4 and B9 as shown in Table 8-5. B4 and B9 load on 14M rising while $\overline{\text{7M}}$ *and* LD194 are high. B4 and B9 shift left on 14M rising pulse while $\overline{\text{7M}}$ is high *and* LD194 is low. A left shift is a shift in the direction of D3 toward D0. The clock edges on which the loads and shifts occur are shown in Fig. 8-27. This figure also shows the clock edges on which A10 loads.

Returning to A8, pin 1 is high, so data input on pin 3 is output on pin 4. Thus,

Table 8-5. S1 and S0 Control Shift Modes

S1 (Pin 10)	S0 (Pin 9)	Mode
0	0	Do Nothing
0	1	Shift Right
1	0	Shift Left
1	1	Load

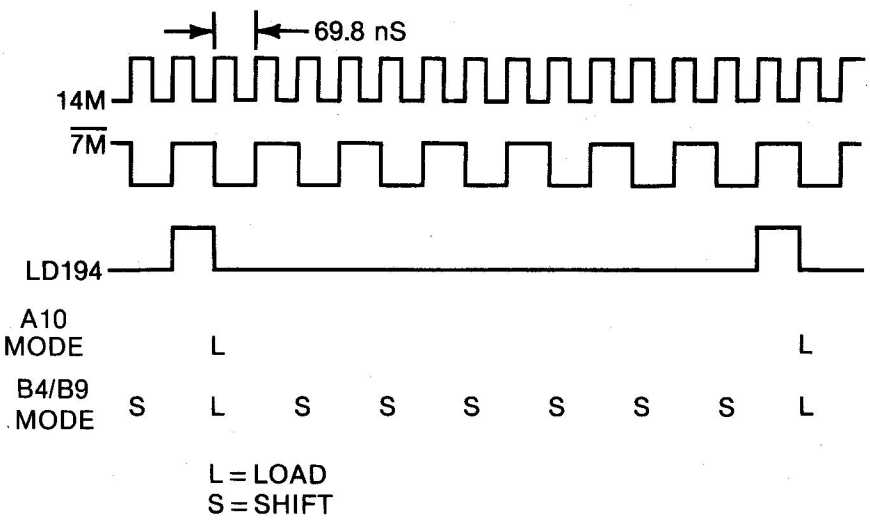


Fig. 8-27. Clock and shift inputs—ICs A10, B4, and B9.

bits that shift out of B9-15 shift into B4-7. This is how B4 and B9 become one long 8-bit shift register.

The signal at B5-2 is high as previously described. This high at A8-6 passes to A8-7, is latched in A10, then appears at A9-10. The signal at A12-13 is low (again as previously described), and this low appears at A9-9. Signal DL7 at A8-13 passes to A8-12, is latched in A10, then appears at A9-11. These three select inputs of A9 (pins 9, 10, and 11) cause it to select either input pin 1 or pin 2 under control of DL7. When DL7 is low, A9-2 is selected. This comes from B4-15, the direct output of the shift register. When DL7 is high, A9-1 is selected. This comes from A11-9, the shift register's output delayed by one 14M clock cycle.

HIRES Mode Timing Diagram—Fig. 8-28* shows the detail timing of HIRES data. The timing for 14M, the video address, and the latched data have been derived previously. The clock edges on which A10, B4, and B9 shift and load are also shown. This information has been duplicated from Fig. 8-27.

The two bytes shown are the left-most display byte (an even byte), and the right-most display byte (an odd byte). The waveforms represent one horizontal scan line. We have shown the timing as if only two bytes were displayed on a line. In practice, 40 bytes are displayed.

Even Byte Timing—The video address for the even byte becomes valid at point A. Since this is the left-most byte, BLANKING goes low at the same time (point B). The video address maps to the HIRES screen memory, and the memory returns the byte to be displayed as latched data, point C. While the latched data is still valid, it is loaded into B4 and B9, points D through K. The numbers (0 through 7) shown in the waveforms correspond to the bit's origin. For example, at point K, bit DL0 loads into B4 and appears at B4-15.

At the same time that B4 and B9 are loaded, DL7 loads into A10-14 and BLANKING loads into A10-13 (point M). Data selector A9 is enabled by the low on A10-13. Assume for now that DL7=0. This causes A9 to select input pin 2. Thus, bit DL0 from B4-15 appears immediately at A9-5, point N. On the next 14M rising pulse, bit DL0 appears at B10-5, the video output (point P). The bit is 140 nS wide when displayed.

On the next 14M rising pulse, B4 and B9 shift. This moves bit DL1 to B4-15, point Q. The other seven bits also shift as shown by the arrows. Note that since B9-15 connects to B9-7, DL4 shifts into B9-12. This is indicated by the arrow to point R. This arrangement serves no function in HIRES, nor does it do any harm.

The data shifts five more times during the display time of this byte. The latched data appears in the video output in the order DL0 to DL6 (point P to point T).

Odd Byte Timing—While the bits for an even byte are shifting out, the video address for the next odd byte becomes valid (point U). The corresponding latched data becomes valid at point V. The data from the odd byte loads into B4 and B9 in the same manner as just described for the even byte. Bit DL0 appears in the video output at point W.

Note that the loading of each byte prevents DL7 of the previous byte from reaching the video output. Since our odd byte is the right-most byte, BLANKING goes high at point X, and A9 is disabled at point Y.

Effects of DL7—Now let's assume that DL7 = 1. In this case, A9-11 is high, and A9 selects input pin 1. Pin A9-1 connects to A11-9 which is signal B4-15 delayed by one 14M clock cycle. Signal A11-9 is shown in Fig. 8-28* just below B4-15. The one clock delay should be evident. The delayed bit stream next appears at A9-5, then at B10-5 after another delay of one 14M clock cycle.

Compare the B10-5 waveforms in Fig. 8-28* for the cases where DL7 = 0 and DL7 = 1. You should see that under control of DL7, we can shift any pixel by 70 nS. This is a distance equal to half of the pixel's width. Thus, there are twice as many horizontal plotting locations on a line as there are pixel cells. This gives us 560 horizontal plotting locations for black and white HIRES, as mentioned in the overview.

We have now shown exactly how each bit of a typical HIRES byte appears in the video output. We will now use this new information to demonstrate the generation of HIRES colors.

HIRES Color Generation—Fig. 8-29 shows the video output for two adjacent HIRES bytes. The first four signals in Fig. 8-29 are simply duplicated from Fig. 8-28*. We have shown the even byte shifting out while the video address is odd, as is the case. COLOR REF is then added to the figure in the proper phase relationship.

Next we show COLOR BURST, but recall that COLOR BURST is active only for a short time following the sync pulse. The *phase* established by COLOR BURST is important, so we have shown that phase continuing into the visible portion of the display as a dotted line. COLOR BURST is delayed about 35 nS by the LC network (L1, C2, and C3 in Fig. C-20*), so we also show this delayed COLOR BURST at Q3.

The next signal in the figure (line A) shows the video output for two bytes, both with the value 85. Bits DL0, DL2, DL4, and DL6 are set when the data equals 85. Since DL7 is not set, the timing will correspond to the signal labeled "DL7 = 0." When the phase angle of the video is compared with that of burst, you can see that the value 85 in an *even* byte is violet while the same value in an *odd* byte is green. The next signal (line B) shows the same two bytes with a value of 213. Bits DL0, DL2, DL4, DL6, and DL7 are set when the data is 213. The timing of this signal will

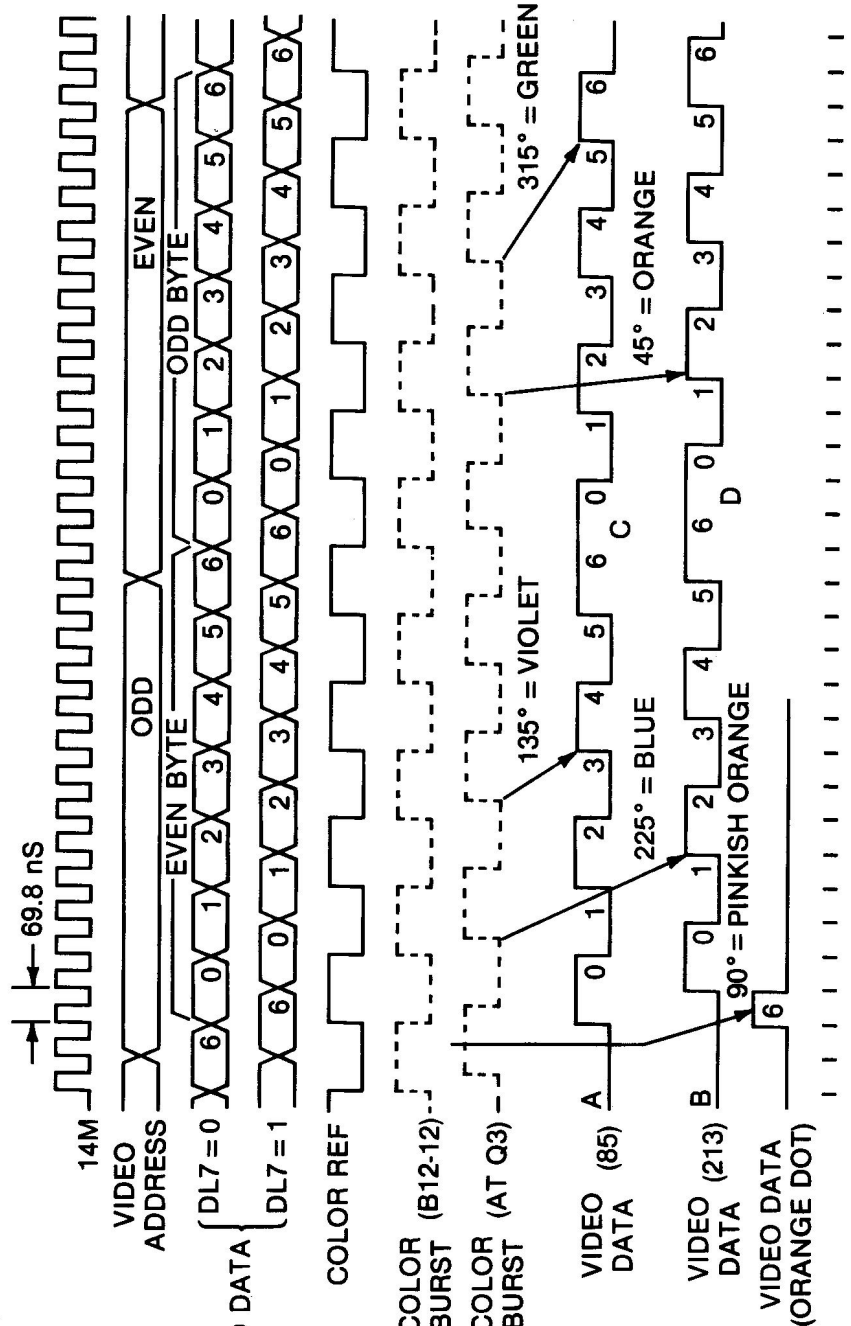


Fig. 8-29. HIRES colors.

correspond to the signal labeled "DL7 = 1." The phase angle results in blue for an even byte and orange for an odd byte.

At points C and D, two adjacent bits are both high. This results in a white dot on the screen. At this juncture in our discussion, you should be able to see how the circuit, as analyzed in this section, supports the theory of HIRES graphics as explained in the overview.

Mysterious Orange Line—Before leaving the subject of HIRES, we must discuss the "Mysterious Orange (or Pink) Line." This vertical line appears from time to time along the left edge of the screen (Reference 8.2). The phenomenon is rooted in the hardware. Notice in Fig. 8-28* that when the video is shifted one clock cycle by A11-9, 70 nS of garbage is shifted such that it falls within the unblanked portion of the screen (point Z). This unwanted half-dot ends up in the video output at point AA (the left edge of the screen).

Can we determine where this dot comes from and what its value is? It comes from bit 6 of the memory location that maps to the video address just to the left of the left-most byte displayed on the screen. If bit 6 of this critical location is set, *and* if bit 7 of the left-most byte is set, then there will be a half-dot at the left edge of the screen (see the last signal plotted in Fig. 8-29). The phase of this dot is such that it will be pinkish-orange in color. The critical addresses themselves can be determined from Fig. 5-8*. They are all within the 16K HIRES page. Of these locations, 128 are legitimate screen locations along the right edge of the screen. The remaining 64 are unused locations. The mapping is indicated in Fig. 8-30. You can see the Mysterious Line by running the following 20 second program.

```
>10 REM MYSTERIOUS ORANGE LINE
 20 POKE -16297,0
 30 POKE -16304,0
 40 POKE -16302,0
100 P = 8192
110 FOR I = P TO P + P
120 POKE I,128 : NEXT I
130 FOR N = 0 TO 7168 STEP 1024
140 FOR I =127 + P + N TO 1023 + P + N STEP 128
150 POKE I,64 : NEXT I
160 FOR I = 39 + P + N TO 935 + P + N STEP 128
170 POKE I,64 : NEXT I
180 FOR I = 79 + P + N TO 975 + P + N STEP 128
190 POKE I,64 : NEXT I
200 NEXT N
999 END
```

LORES Mode

Our detailed analysis of LORES mode starts with the configuration of the video generator by the soft switches. We then follow the timing of the video signals as they propagate through the circuit.

LORES Mode Circuit Configuration—For LORES, the TEXT MODE and MIX MODE soft switches are low. This results in a high at B13-13 (Fig. C-16*). After three $\overline{\text{RAS}}$ pulses, this high appears at B8-2. The HIRES MODE soft switch is low, so HIRES is low. This arranges the memory mapper to fetch LORES characters from the text page.

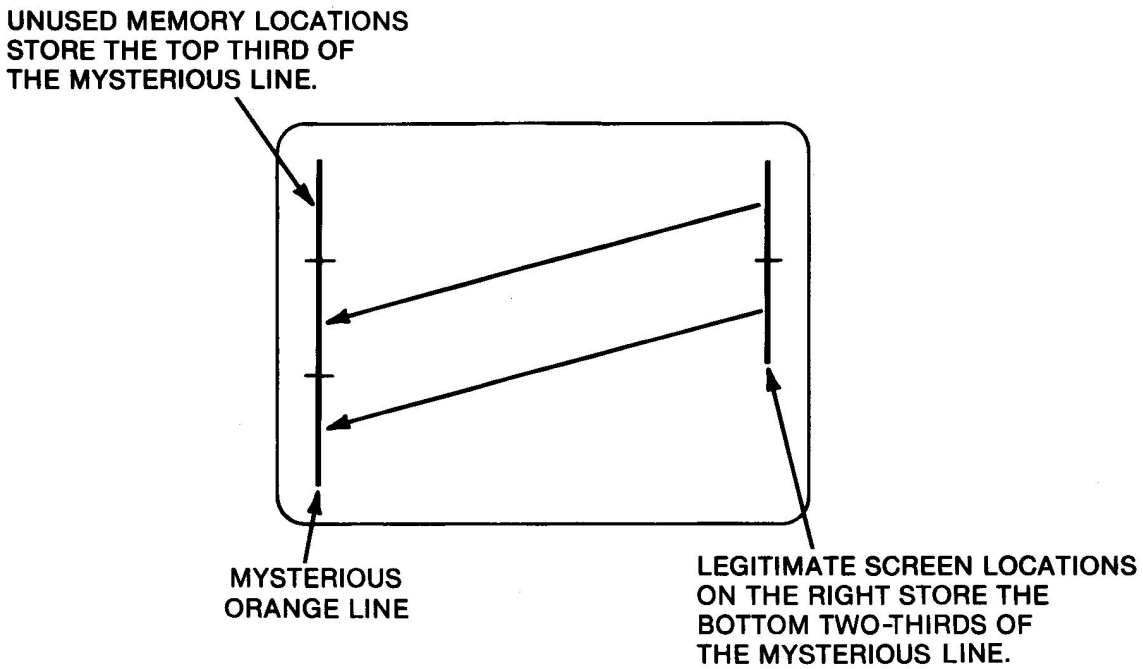


Fig. 8-30. Mysterious orange line.

The high from B8-2 makes A12-10 low. Since HIRES MODE is also low, A12-13 is high. Flip-flop A11-6 inverts this, and we get a low at data selector A8's input pin 1. This causes A8 to select its "0" inputs. Thus, the high pull-up at A8-11 appears at A8-9 which connects to B4-10 and B9-10. Since this line is a continuous high, B4 and B9 are limited to two modes: load and shift left. Signal LD194 connects to their pin 9, so B4 and B9 will load when LD194 is high, and shift when LD194 is low. Of course they only shift or load on 14M's rising edge. A left shift is a shift in the direction from Q3 toward Q0.

Bits shifting out of B4-15 will pass through A8 (A8-2 to A8-4) and back into B4 and pin 7. Bits shifting out of B9-15 will immediately shift back in on pin 7. Thus, the four bits loaded into each shift register rotate on each shift as described in the overview.

Signal VC passes through A8 (A8-5 to A8-7) and is stored in A10. It then appears at pin 10 of data selector A9. Signal H0 follows a similar path through A8 (A8-14 to A8-12) and is also stored in A10. It then appears at A9-11. NOR gate A12-13 is high as previously described, and this high appears at A9-9. These three most recently mentioned inputs to A9 are its *select* inputs. They cause A9 to select outputs from the shift registers as shown in Table 8-6.

Table 8-6. Signals VC and HO Influence the Output of A9

VC	HO	Selected Source	Description
0	0	B4-15	Upper Pixel, Even Byte
0	1	B4-13	Upper Pixel, Odd Byte
1	0	B9-15	Lower Pixel, Even Byte
1	1	B9-13	Lower Pixel, Odd Byte

LORES Timing Diagram—Fig. 8-31* shows the detailed timing of two bytes of LORES data. The timing for 14M, LD194, the video address, and the latched data have been previously derived. The 14M clock edges on which B4 and B9 load are also shown. On all other 14M rising clock edges, B4 and B9 shift. The two bytes shown are the left-most display byte (an even byte), and the right-most display byte (an odd byte). The waveforms represent one horizontal scan line through the pixels. The figure is drawn as if there were only two pixels displayed on the line. In practice, 40 pixels are displayed.

Even Byte Timing—The video address for the even byte becomes valid at point A. Since this is the left-most byte, BLANKING goes low at the same time, point B. The video address maps to the text/LORES screen memory, and the memory returns the byte to be displayed as latched data, point C.

The latched data is loaded into B4 and B9 in line with point D. All eight "Q" outputs of these shift registers are shown in the figure. The numbers shown in the waveforms correspond to the bit's origin. For example, at point E, DL0 loads into B4 and appears at B4-15. At the same time that B4 and B9 are loaded, H0, VC, and BLANKING load into A10 (points F, G, and H). This is an even byte, so A10-14 goes low at point F. Signal A10-15 will be low if we are scanning through an upper pixel. It will be high for a lower pixel. It does not change at point G since it assumed its correct value several microseconds before. Thus, A10-13 goes low at point H to enable A9.

We will assume in this example that VC is low (upper pixel). H0 (latched at A10-14) is also low, so A9 selects B4-15. Thus, bit DL0 appears at A9-5 (point I). On the next 13 rising edges of 14M the shift registers recirculate their contents as shown in the figure. Data selector A9 continues to select the bits at B4-15, and they appear at A9-5 as shown. After one 14M pulse delay, the bits at A9-5 appear at B10-5 (the video output). Each bit is 70 nS wide.

Odd Byte Timing—While the bits for an even byte shift out, the video address and latched data for the next adjacent odd byte become valid (points J and K). In line with point M, the odd data is loaded into B4 and B9. At the same time, H0 is loaded into A10. Signal A10-14 goes high (point N) since this is an odd byte.

For the next 13 clock pulses, the bits recirculate in the shift registers just as they did for the even byte. Signal A10-14 is high for the odd byte, however, so A9 selects B4-13 to output on A9-5. For example, bit DL2 at point P appears at point Q. After a one clock pulse delay, A9-5 appears at the video output, B10-5.

When the video address changes at point R, BLANKING goes high (point S) since we are at the right edge of the screen. At point T, A10 loads again, and A10-13 goes high to disable A9.

Note that the output bit sequence remains in order at the boundary between even and odd bytes (point Q). This is a result of H0's control over data selector A9. In this example, we assumed that VC is low. If VC is high, the video output contains bits 4, 5, 6, and 7 in that order.

LORES Color Generation—We have shown above bit for bit how the LORES data appears in the video output. In Fig. 8-32, we demonstrate by two examples how these bit positions create colors. The first four waveforms in the figure have been duplicated in the correct relative phase from Figs. 8-29 and 8-31*. We have shown two bytes. Remember that even bytes shift out while the video address is odd and vice versa.

The fifth signal in Fig. 8-32 is the video output that results if both bytes equal 2. Note that the DL1 bit is high. The phase delay from burst is 180 degrees (blue), and

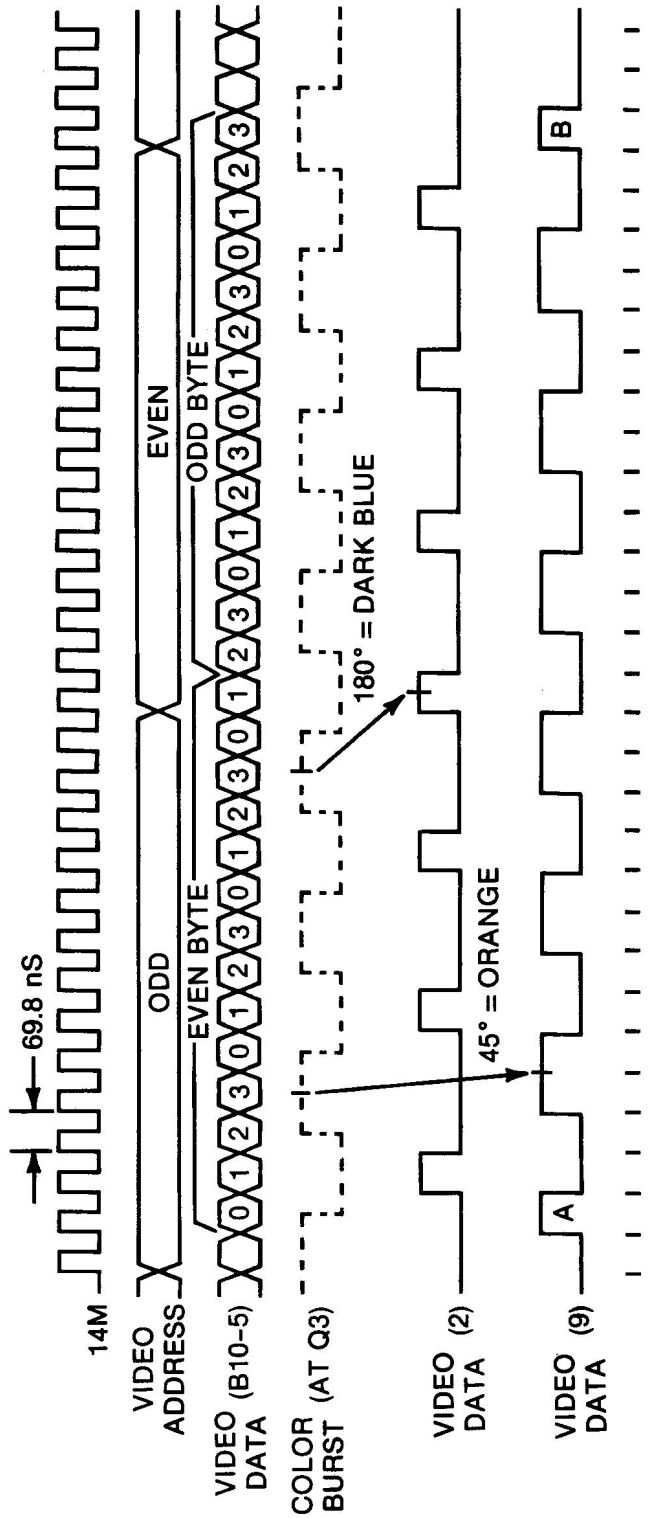


Fig. 8-32. LORES colors.

the duty cycle is 25% (dark). Thus, the color is *dark blue*. The last signal in the figure shows both bytes equal to 9. Note that bits DL0 and DL3 are high. The phase delay from burst is 45 degrees, so the color is orange. Some colors (orange is an example) will have a slight waveform discontinuity at the edge of the pixel, points A and B.

By these two examples, we have shown how the circuit actually produces LORES colors. At this point, you should be able to tie together the LORES theory presented in the overview with the actual circuit implementation.

In the previous section, we described a recirculating shift register that can generate color video signals directly from digital data. A video signal generator using this technique is a major claim of U.S. Patent No. 4,278,972.

Mixed HIRES and Text

You may have wondered about the function of flip-flops B5-2 and B8 (Fig. C-16*). They provide an orderly timing sequence for the transition between graphics and text displays.

Fig. 8-33 shows the transition between HIRES and text when these two modes are mixed on the screen. We have shown the right-most byte of the HIRES scan line that is just above the four lines of text. The first six waveforms in the figure have been previously derived. The clock edges on which A10, B4, and B9 load are indicated under the 14M waveform.

The video address of the right-most byte becomes valid at point A. The memory system provides valid data at point B. At point C, this data is loaded into shift registers B4 and B9 as previously described. Bit DL0 appears at B4-15. We will assume that DL7 is low, thus A9 selects B4-15 to output on A9-5 (point D). After one clock delay, bit DL0 appears at the video output (point E). The remaining bits then shift out as previously described.

Meanwhile, when the video address changes at point F, V2 and V4 both go high. (Refer to Chapter 4 for V2 and V4 timing.) The MIX MODE soft switch is high, so B12-6 goes high (point G) and B13-13 goes low. Thus, B13-13 is the signal that tells the video generator to switch its configuration from graphics to text. The circuit can't reconfigure immediately, however, since it is still shifting out the last HIRES byte.

The low at B13-13 is clocked into B5-2 on $\overline{\text{RAS}}$ rising at point H. The next $\overline{\text{RAS}}$ rising edge brings us to point I where B8-15 goes low. Meanwhile, the low at B5-2 has propagated through A8 (A8-6 to A8-7) and appears at A10-3. The next time A10 loads, A10-15 goes low (point J). This causes A9 to deselect HIRES signals from B4-15, and to select text signals from B2-11. Since this switch is coincident with the beginning of blanking (point K), we get a clean transition in the video output.

On the next $\overline{\text{RAS}}$ rising edge, B8-2 and HIRES both go low (points M and N). HIRES must of course go low so that the next 32 scan lines will map to the text page of memory and not the HIRES page. HIRES is made synchronous with $\overline{\text{RAS}}$ so that it (HIRES) will change only while $\overline{\text{CAS}}$ is inactive (point P). Recall that HIRES is decoded by F2 to help select one of three individual $\overline{\text{CAS}}$ lines. If HIRES were allowed to change while $\overline{\text{CAS}}$ is active, it could shorten the $\overline{\text{CAS}}$ for one memory block and produce a glitch on the $\overline{\text{CAS}}$ of another block. Either event would most likely cause loss of data.

While on the subject of shortened $\overline{\text{CAS}}$ pulses, you may have noted that the HIRES soft switch can directly change HIRES via B11-6. Fortunately the soft switches change state on $\phi 0$ rising, which occurs while $\overline{\text{CAS}}$ is inactive.

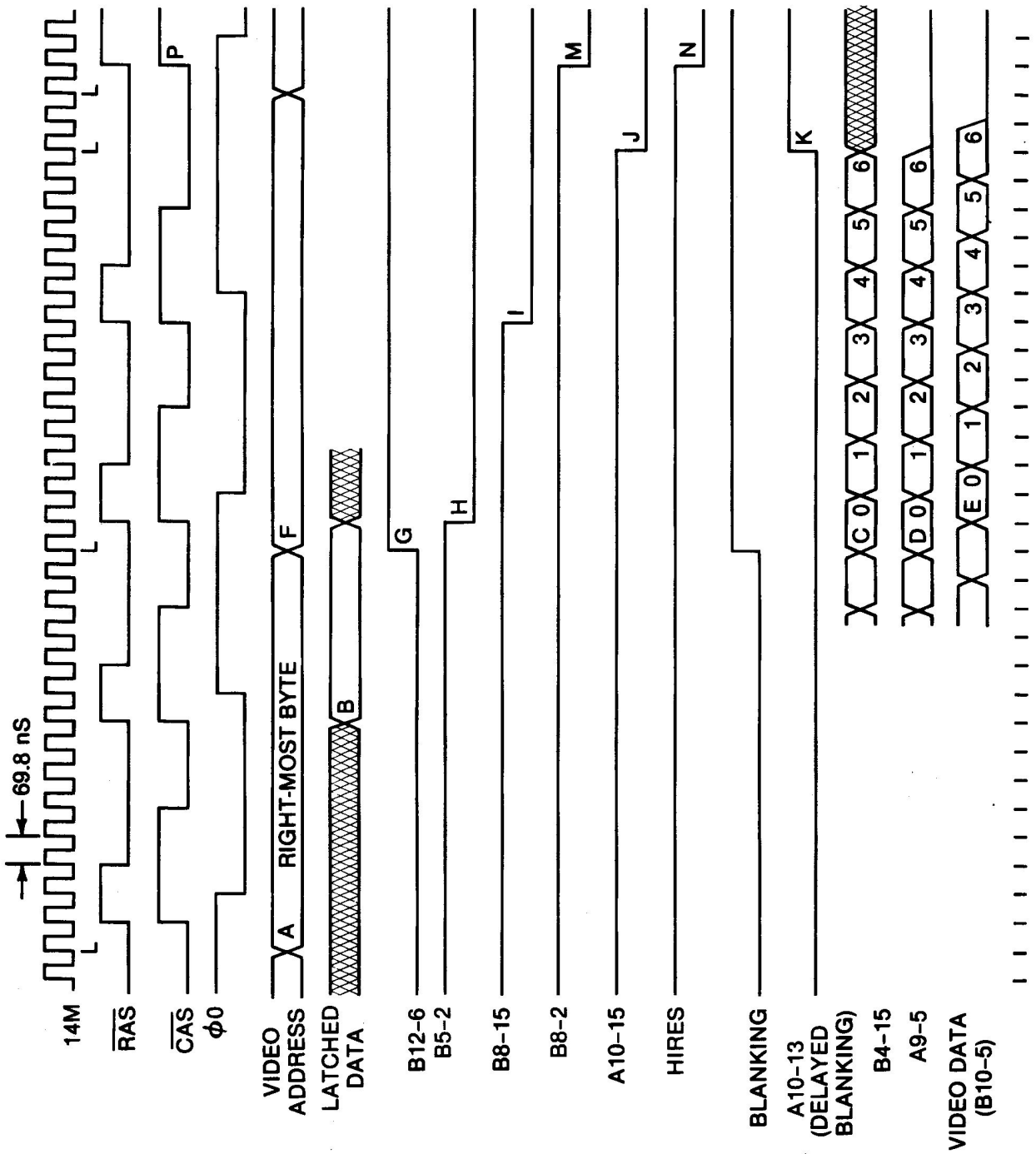


Fig. 8-33. Mixed HIRES and text.

Following the four lines of text, the video generator has to reconfigure itself to display graphics. This transition occurs during vertical blanking with HIRES being synchronized to $\overline{RAS}$ in a manner similar to that shown in Fig. 8-33.

In summary, flip-flops B5-2 and B8 delay the transition from graphics to text until blanking. The flip-flops also synchronize HIRES to $\overline{RAS}$ so that there will be no shortened $\overline{CAS}$ pulses. This concludes our discussion of mixed HIRES and text.

Mixed LORES and Text

The transition between LORES and text is depicted in Fig. 8-34*. The figure shows the right-most byte of the LORES scan line that is just above the four lines of text. Most of the waveforms in the figure have been previously presented, so we will concentrate on new information.

When the video address changes at point A, V2 and V4 both go high. The MIX MODE soft switch is high, so B12-6 goes high (point B) and B13-13 goes low. This low is clocked into B5-2 on $\overline{RAS}$ rising at point C. On the next $\overline{RAS}$ rising edge, B8-15 goes low (point D). Signal VC also goes low at point A, so when A10 next loads, A10-15 goes low (point E). In a similar manner, H0 goes low at point A, so A10-14 goes low at point F.

On the next $\overline{RAS}$ rising edge, B8-2 goes low (point G). This causes A12-13 to go low, point H. Since A12-13 connects to one of A9's select inputs (Fig. C-16*), A9 deselects the LORES graphics coming from the shift registers. This is a satisfactory point for the transition since we are now into blanking (which started at point I).

The low at A12-13 makes A11-6 go high, point J. This high appears at A8-1 causing A8 to reconfigure the circuit. The low at B5-2 passes through A8 (A8-6 to A8-7) and A10 (A10-3 to A10-15), then finally appears at A9-10. This causes A9 to select text signals from B2-11.

The circuit is now in text mode so the four lines of text can be displayed. During vertical blanking, the circuit switches back to LORES mode in a manner the reverse of that shown in Fig. 8-34.* This concludes the discussion of mixed LORES and text.

SUMMARY

In this chapter we have explained both the concepts and circuit details of the video modes of the Apple II. The Apple video generator consists of shift registers and data selectors. The data selectors configure the shift registers into circuit arrangements that differ for each mode. In each arrangement, the shift registers convert parallel graphics data into a serial bit stream. This serial data then becomes the video component of the composite video output of the Apple II.

This chapter also marks the conclusion of *The Apple II Circuit Description*. If you are designing peripherals or modifications for the Apple II, this book should have given you a major start. If your goal is repair of the mother board, the waveform drawings and large schematics should be a valuable maintenance aid. For students of digital design, a very popular and practical product has been described. And finally, if you were simply curious about the operation of Apple's hardware, that curiosity should now be satisfied.

APPENDIX A

Video Techniques

The Apple II has a composite video output that may be connected to a video monitor or to the antenna terminals of a television receiver via an rf (radio frequency) modulator. In order to understand the Apple video circuitry, it is first necessary to understand some basics about television. In this appendix we provide a brief introduction to video, using actual parameter values from the applicable U.S. standards. We also compare the video of the Apple II with these standards.

The signal we are discussing is called *composite video*. Such a signal is found at the video output of the Apple, at the video input of a video monitor, at the video input of an rf modulator, or inside a tv receiver at the *output* of the tuner. This signal has information content from near dc to several megahertz. It has an amplitude of about 1 volt. For our purposes, there is no audio in this signal.

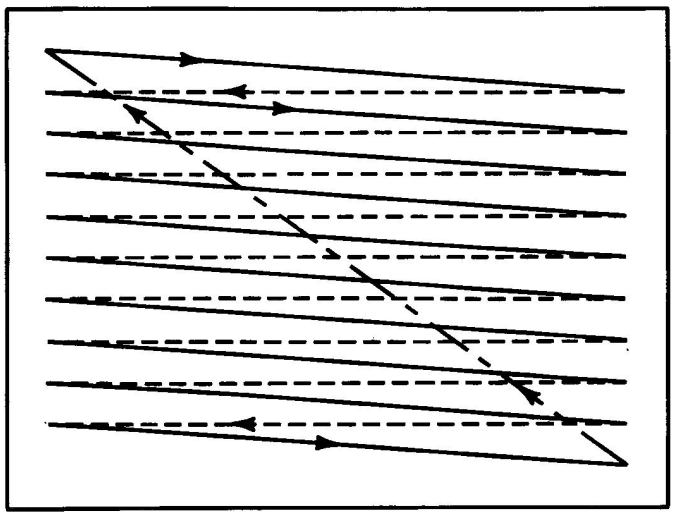
The composite video signal should not be confused with an *rf* signal. An rf signal as used in television is a carrier in the range from 54 MHz to 900 MHz that has been modulated by a composite video signal. An rf signal would be found at the *output* of an rf modulator or at the antenna terminals of a tv set. Such an rf signal usually ranges from 300 to 2000 microvolts.

THE BASIC VIDEO DISPLAY

A video display is created by sweeping an electron beam across the face of a crt (cathode ray tube) as in Fig. A-1. Of course in practice, there are many more lines than shown. The solid lines are visible and form the image. The dashed lines are not visible. The dashed lines show the path of the beam as it retraces back to the left at the end of each line and as it retraces to the top after each field. A *field* is one complete journey of the beam from top to bottom including retrace back to the top. The rapid motion of the beam (about 60 fields/sec) makes the screen appear to the eye to be lighted at all points simultaneously.

The video display of Fig. A-1 is *noninterlaced*. This term will have more meaning shortly when we show an interlaced video display. Noninterlaced video has many applications, but it is not the broadcast standard used for transmitting television signals over the air.

The image you see on the screen is formed by modulating the intensity of the



CRT FRONT VIEW

KEY:

——— VISIBLE TRACE

- - - - - INVISIBLE HORIZONTAL RETRACE

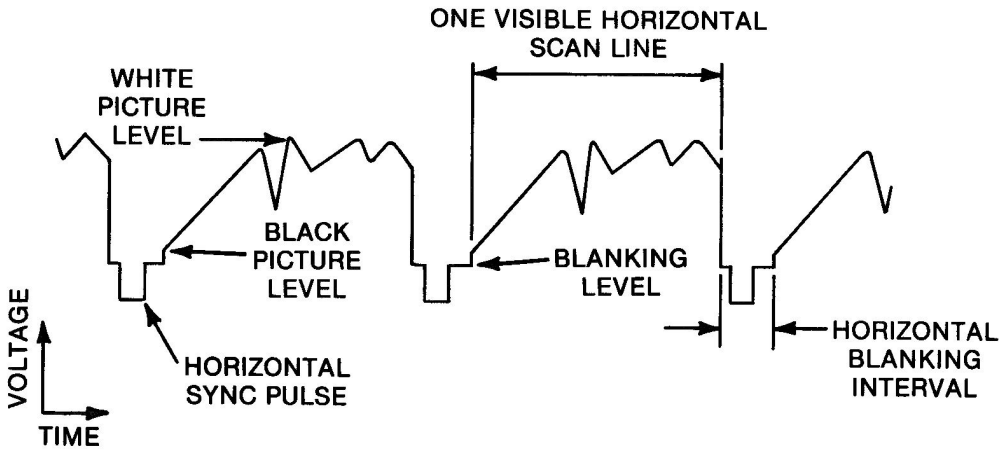
——- - —— INVISIBLE VERTICAL RETRACE

Fig. A-1. Basic video display (noninterlaced).

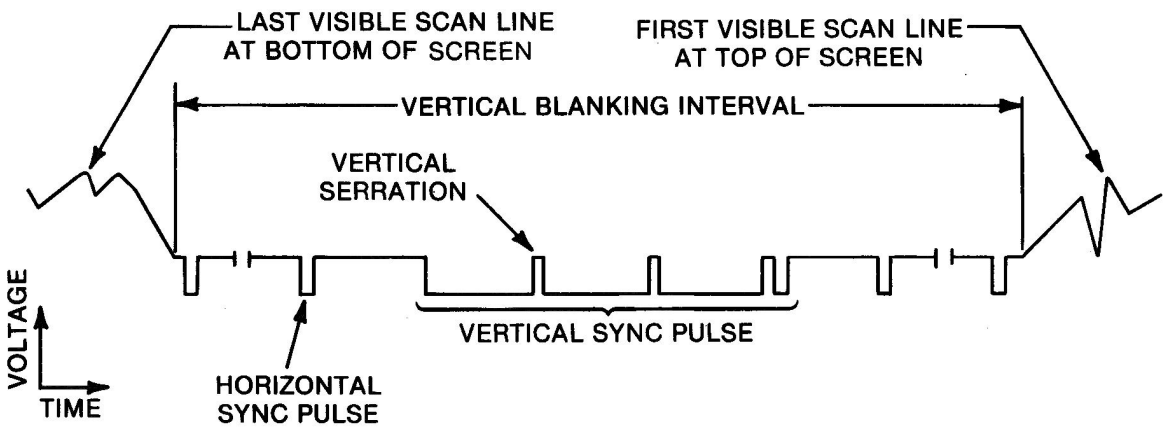
beam. The beam is *blanked* during retrace, causing the retrace lines to be invisible. Blanking is accomplished by applying a modulation that corresponds to a level slightly darker than the blackest picture elements. The horizontal and vertical retrace times are shorter than the display times. This is so the beam can spend most of its time displaying picture material.

A video system must provide a synchronization means so that the beam in the crt will sweep in step with the beam in the transmitting device (often a tv camera tube). Both beams must return to the left simultaneously at the end of each horizontal line. Both beams must also return to the top simultaneously after each field. To accomplish this, the video signal contains a *horizontal sync* (synchronization) pulse between lines and a *vertical sync* pulse between fields. A composite video signal consists of the picture modulation for the beam, the blanking signals and the sync signals.

In Fig. A-2A we show the composite video signal for two complete horizontal lines. *White* picture level is in the positive direction and *black* is in the negative direction. Blanking level is slightly more negative than black, and sync pulses are more negative still. In Fig. A-2B we see the signal in the area of a vertical sync pulse. Horizontal sync pulses continue throughout vertical blanking and vertical retrace. This is to maintain the horizontal oscillator of the receiver on frequency during this interval. Since there are no horizontal sync pulses during the vertical sync pulse, the vertical sync pulse is interrupted at the horizontal rate by *serrations*. The serrations thus provide horizontal timing information during the vertical sync pulse. Note that the horizontal and vertical blanking intervals extend beyond each edge of their respective sync pulses.



(A) Typical horizontal scan lines.



(B) Vertical blanking interval.

Fig. A-2. Composite video waveforms (noninterlaced).

BROADCAST STANDARDS

Broadcast television uses a technique called *2:1 interlace*. For a fixed bandwidth and line quantity, interlacing reduces visible flicker. In 2:1 interlace, each complete picture or *frame* is divided into two fields with the lines of one field interlaced between the lines of the other field, Fig. A-3. The figure shows fewer lines than actually exist in practice. In the U.S. there are 262.5 lines per field and 525 lines per frame. Each frame consists of two interlaced fields, one named *even* and one named *odd*. For black and white tv, the frame rate is 30 Hz and the field rate is 60 Hz. The horizontal line rate for black and white television is 525 lines/frame × 30 frames/sec = 15,750 lines/sec, or 15.750 kHz.

The horizontal sync and blanking signals are the same for both interlaced and noninterlaced tv, so Fig. A-2A still applies. However, the vertical signals differ (see Fig. A-4). Note that after each field, the vertical sync pulse alternately advances and delays one-half horizontal line. This is how the interlaced scans are generated. Note also the addition of the *equalizing interval*. This allows the sync separator in the tv

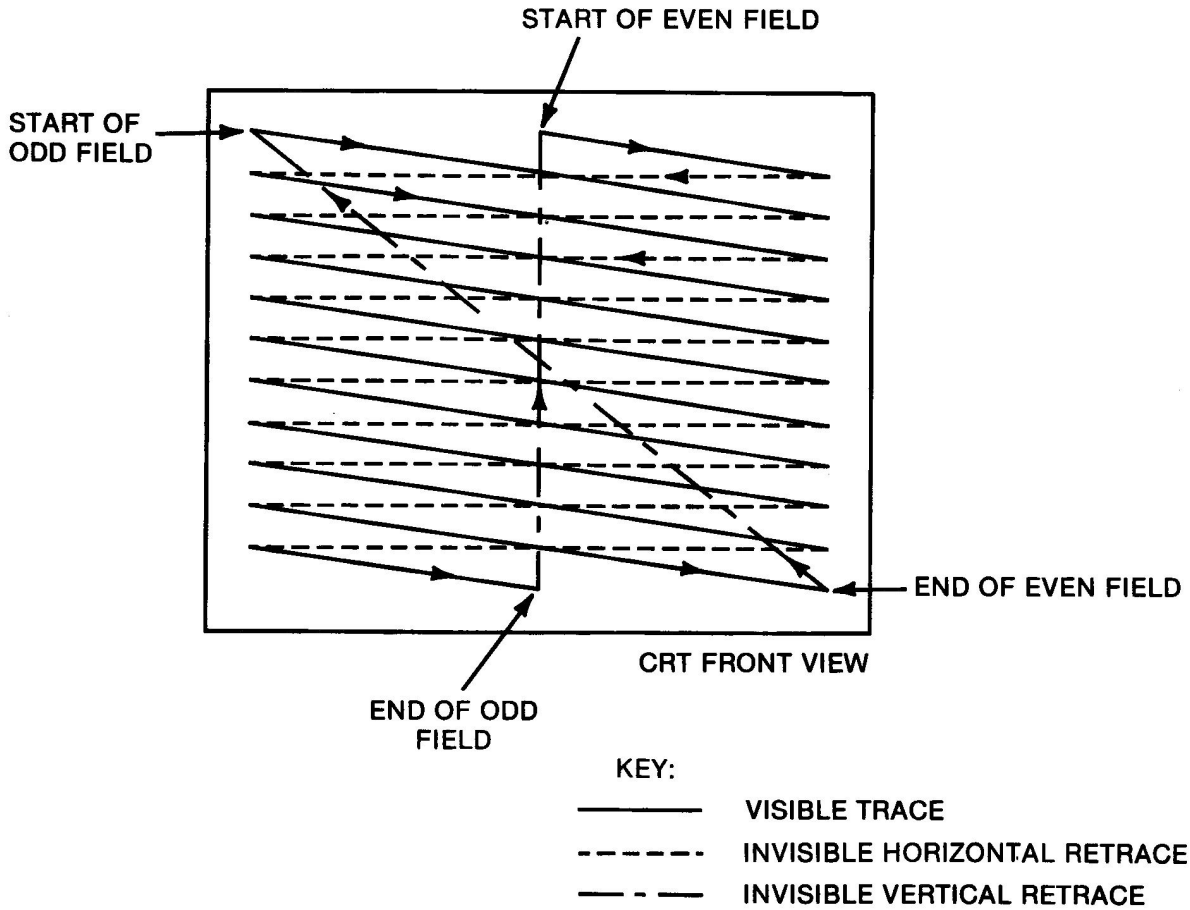


Fig. A-3. Video display (2:1 interlace).

set to trigger the vertical retrace at the same time (relative to the vertical sync pulse) on both odd and even fields.

We do not need to explore further the interlaced signal to understand the Apple's video; the Apple II's output is noninterlaced. Our object is to simply introduce the topic and point out that the Apple II does not output a broadcast standard 2:1 interlaced signal.

COLOR

The color tv process starts by dividing the image into three primary colors: red, blue, and green. This is often done with a tv camera that contains three camera tubes, one for each color. The three signals for the three colors are then combined in the proper proportions to create a brightness, or *luminance*, signal. The luminance is the same signal described previously for black and white tv. It is the signal that modulates the beam of a black and white crt to produce a picture. One purpose of including the luminance signal in the color broadcast signal is compatibility. This compatibility allows a black and white set to display a signal that is broadcast in color.

What about color tv sets? For the benefit of color sets, the red, blue, and green signals are used to modulate a 3.579545 MHz *subcarrier* that is added to the composite video. A color receiver demodulates the subcarrier and uses the resulting information along with the luminance to create a color image. A color crt displays the

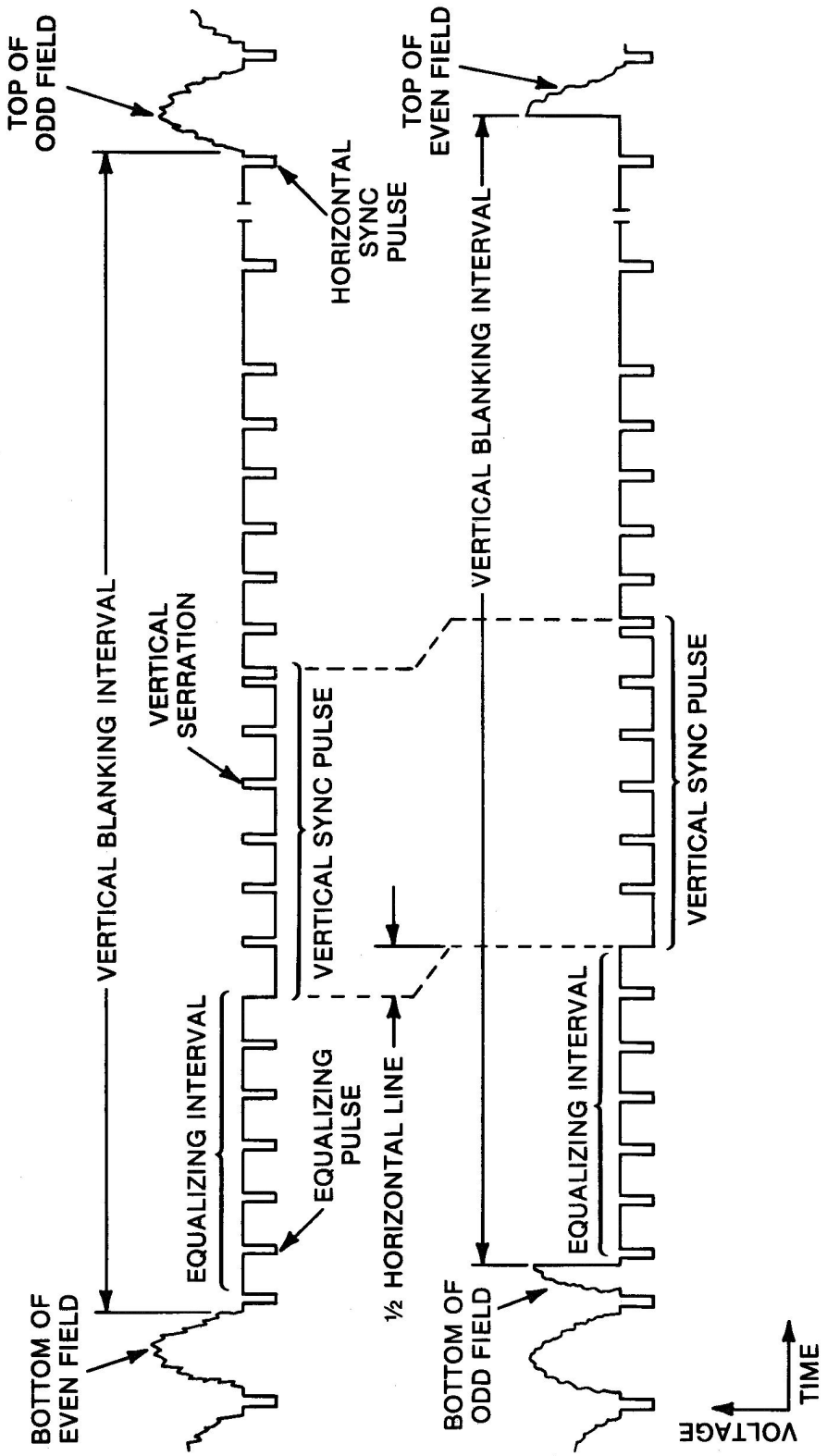


Fig. A-4. Vertical blanking interval (2:1 interlace).

image as many small dots or rectangles of the primary colors. Your eye blends the small dots together to create a continuous color image.

In Fig. A-5 we introduce the color circle as an aid to understanding how the subcarrier can contain the color information. Note that all colors can be represented by points around the circle. After modulation at the transmitting end, the phase of the 3.579545 MHz component determines the *hue* and its amplitude determines the intensity or *saturation*. In Fig. A-5, 0 degrees is the phase of a *burst* of subcarrier that is transmitted after each horizontal sync pulse. There are typically 9 cycles of the subcarrier in the burst. A color tv set uses this color burst to synchronize its internal 3.579545 MHz oscillator. It is with respect to this internal oscillator that the color set interprets the phase of the subcarrier to determine the hue.

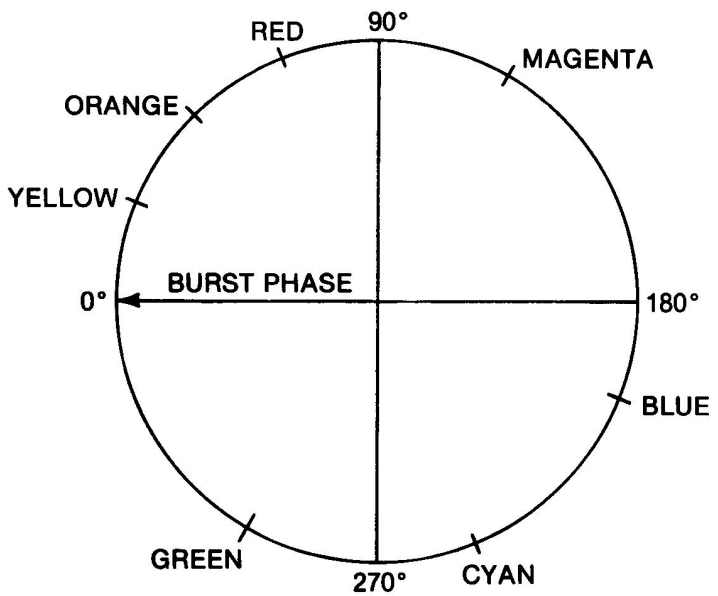


Fig. A-5. Color circle.

In summary, the two important parts of the color signal are the color burst (which sets a reference) and the subcarrier (whose phase relative to the burst determines the hue). Recall that the luminance signal is still present and its importance is in determining the brightness. For example, a subcarrier at "red" phase added to a high luminance level will be light-red or pink.

Fig. A-6 shows a color burst signal followed by a video signal that creates a blue horizontal line. The line is blue since its phase lags the burst phase by 180 degrees (refer to the color circle).

When the standards for color television were formed, the frequency of the color subcarrier was selected to be toward the high end of the video band. This would reduce interference with the luminance signal. The color subcarrier also had to be below the 4.5 MHz sound subcarrier used in broadcast tv. The exact frequency was selected to reduce interference, such as beat patterns between the video, color, and sound signals. This was accomplished by adjusting some of the frequencies to be exact multiples of other frequencies contained in the composite video.

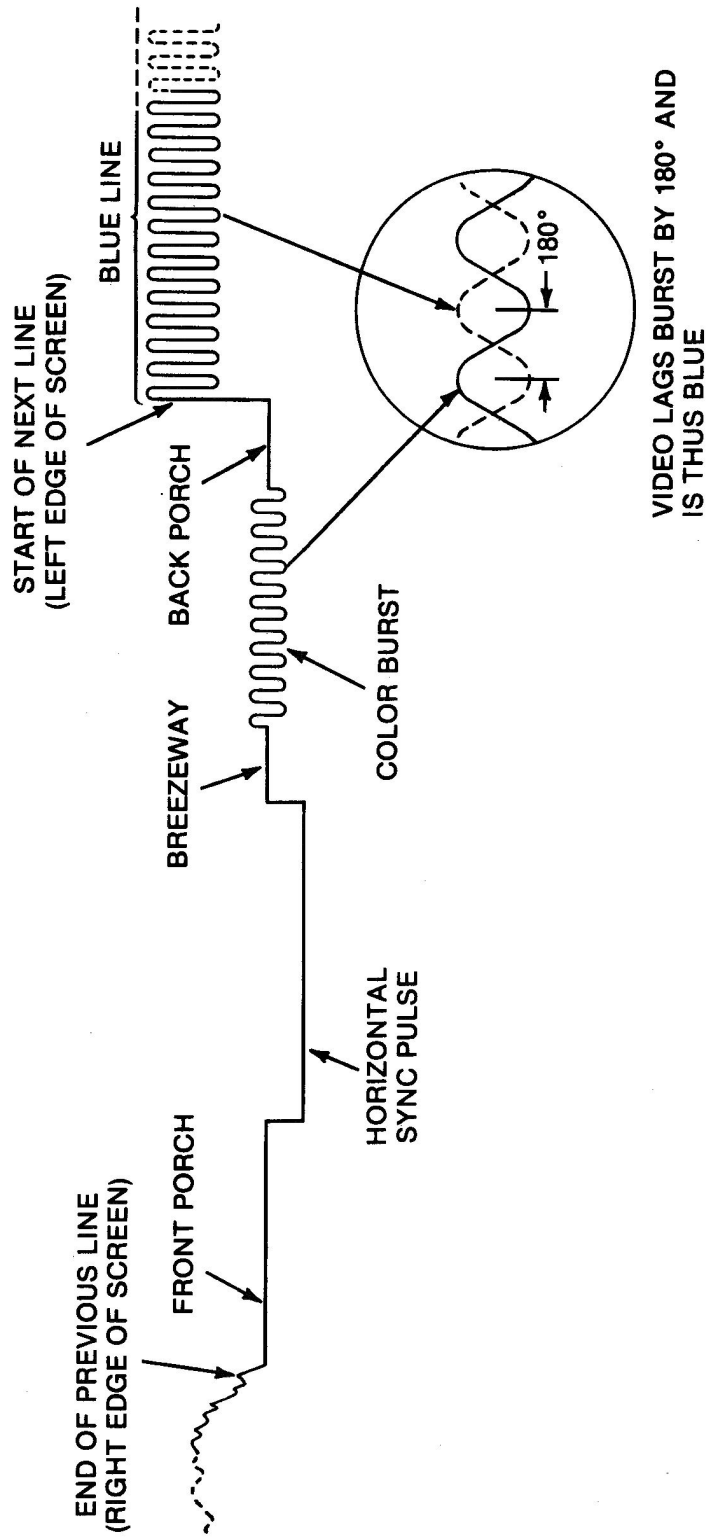


Fig. A-6. Color burst.

First the horizontal rate was adjusted slightly so that the sound would be an even multiple (572) of half the horizontal rate. Therefore, the horizontal line rate for color tv is

$$2 \times \frac{4.5 \text{ MHz}}{572} = 15.734266 \text{ kHz}$$

Next the color subcarrier was selected to be an odd multiple (455) of half the horizontal rate, or

$$\frac{15.734266 \text{ kHz}}{2} \times 455 = 3.579545 \text{ MHz}$$

The horizontal rate is usually rounded to 15.734 kHz. Note that the vertical field rate for color tv is

$$\frac{15.734 \text{ kHz}}{262.5 \text{ lines}} = 59.94 \text{ Hz}$$

The differences between scan rates for black and white and color broadcasts are so small that a tv set can sync to either one.

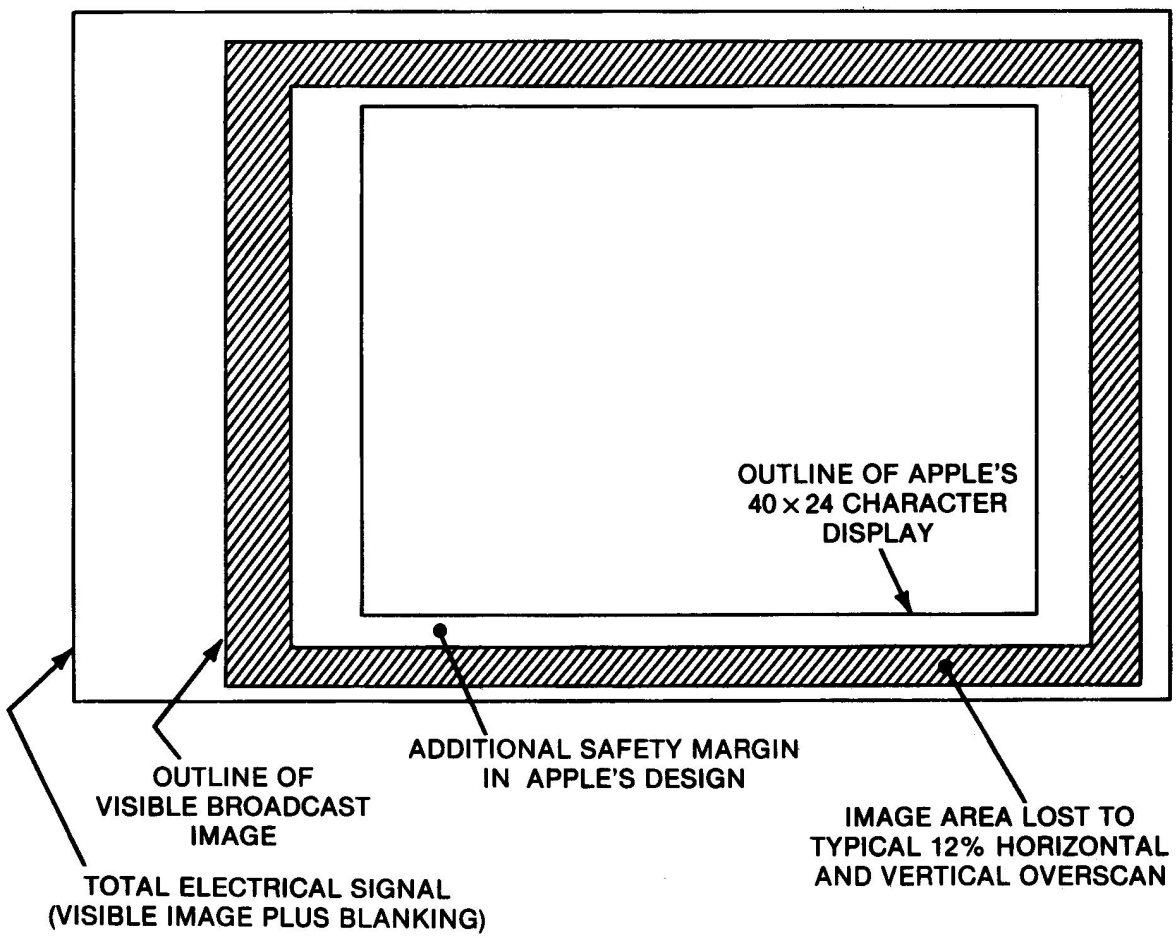


Fig. A-7. The video area of the Apple compared with the standard broadcast image.

Table A-1 Apple's Video Parameters

Parameter	Broadcast Standard	Apple II		
		Rev. 0	Rev. 1	Rev. 7 and RFI
Interlacing	2:1	No		
Lines per Field	262.5	262		
Line Rate	15.734 kHz (Color) or 15.750 kHz (B&W)	15.700 kHz		
Field Rate	59.94 Hz (Color) or 60 Hz (B&W)	59.92 Hz		
Color Subcarrier	3.579545 MHz	3.579545 MHz		
Burst Length	9 Cycles[1] (2.24 μS min)	14 Cycles (3.9 μS)		
Front Porch	1.27 μS min	7.9 μS		
Hor Sync Pulse	4.45–5.08 μS	7.8 μS		3.9 μS
Breezeway	381–735 nS	70 nS		
Color Backporch	1.27 μS min	5.0 μS		8.9 μS
B & W Backporch	4.25 μS min	8.9 μS		12.8 μS
Sync to Blanking End	9.2 μS min	16.7 μS		
Sync to Burst End	7.78–7.94 μS	11.7 μS		7.8 μS
Hor Blanking	10.5 μS min	24.6 μS		
Sync to Burst Start	5.3 μS[1]	7.9 μS		4.0 μS
Vert Serration	4.5 μS	None		8.9 μS[2]
Vert Blanking	1.14 mS min	4.48 mS		
Vertical Sync Pulse	190 μS (3H)[3]	1.02 mS (16H)		255 μS (4H)
Intended Output Load Impedance	75 Ω	75 Ω		
Peak-Peak Level into 75 Ω	1.0 V	0–1.4 V (Adjustable)		
Black Level	+7.5 Units[4]	0 Units		
White Level	+100 Units	+110 Units		
Sync Level	−40 Units	−30 Units		
P–P Burst Level	20 Units	25 Units		

Notes: [1] Recommendation.

[2] Rev. 7 vertical serration is a double pulse, see Fig. B-2*.

[3] H = 1 horizontal line.

[4] Level relative to blanking. 140 Units = peak-peak signal amplitude.

OVERSCAN

The sweep circuits in a standard tv set are designed to *overscan* the face of the picture tube. As a result, parts of the picture as broadcast are lost around the edges. This is done so that even with component aging or line voltage fluctuations that tend to shrink the picture, the screen will still be full. A 12% horizontal overscan is typical. This means that the picture area available for Apple's output is limited if we are to avoid losing part of the display due to overscan. Fig. A-7 shows (to scale) Apple's video area relative to that of a standard broadcast image.

SUMMARY

The parameters of the Apple II video output are listed in Table A-1 next to the applicable broadcast standard. Even though it does not meet the standards in all ways, the Apple II works well with most tv sets and monitors. However, there have been cases of partial incompatibility with some color sets, projection tv sets, video cassette recorders, and broadcast switching equipment. Table A-1 contains information for those researching such problems. Note that later revisions of the Apple II conform more closely to broadcast standards.

APPENDIX B

Apple's Revisions

Like many electronic products that are manufactured for several years, the Apple II has undergone revision from time to time. Some revisions are product enhancements, such as adding more graphics colors. Some changes, like adjusting the cassette input gain, improve performance or reliability. Other revisions reduce cost by taking advantage of changing technology. An example is elimination of the ability to use the old 4K RAMs. One revision expands the market by providing optional compatibility with European television. Finally, some changes are mandated by government regulation. Here we speak of the redesign required for compliance with FCC rules on radio frequency interference.

The list of revisions may seem long, but the Apple II has actually sustained little functional change since its initial release in 1977. This can be credited to the foresight and innovation of the original design.

In this appendix we describe mother board and keyboard circuit changes that have occurred since the Apple II's initial design. We also note which changes occurred at which revision level and how to determine the revision of a particular Apple.

The circuit description in the body of this book covers the current Revision D RFI mother board and the two-piece keyboard. When earlier revisions require different descriptions, those descriptions are provided in this appendix.

OVERVIEW OF REVISIONS

Mother Board

The Apple II mother board has been manufactured in two basic varieties: *Non-RFI* and *RFI*. The non-RFI board was produced until 1981 when Apple switched to the RFI board. Radio frequency interference (RFI) refers to the electromagnetic noise given off by high-speed circuits such as computers. This noise often interferes with radio and tv sets. The FCC can restrict the sale of computers that exceed certain RFI limits. For this reason, Apple redesigned the mother board to reduce RFI. This redesigned board is referred to as the "RFI Mother Board." Earlier boards are referred to as "Non-RFI Mother Boards."

The RFI redesign involved such things as interchanging the location of + 5 volt and ground buses, isolating some circuit grounds, and adding ferrite beads. There were few functional changes. (Note that the Apple II RFI redesign involved more than the mother board. Part of the change involved shielding the plastic case and using shielded cable to the peripherals.)

Within the two major categories of RFI and non-RFI, there are additional revisions. We will list these in later sections.

Non-RFI—The non-RFI mother boards are designated by the nine digit part number 820-0001-XX. The "XX" is an incrementing revision number that starts with 00. The part number is found near the F1 location (left edge of board) on later revisions, and under the 6502 microprocessor IC on earlier revisions. (In this latter case, it is necessary to remove the 6502 to see the part number.) On the first or "Rev. 0" board, there may be no part number at all.

It is possible to determine the revision of a non-RFI mother board by examining certain features on the board. This is handy if the part number is not visible. A Rev. 0 board does not have an Auxiliary Video Pin. This pin is a single vertical pin located between the four-pin Auxiliary Video Connector and the video level adjust potentiometer (see Fig. B-1).

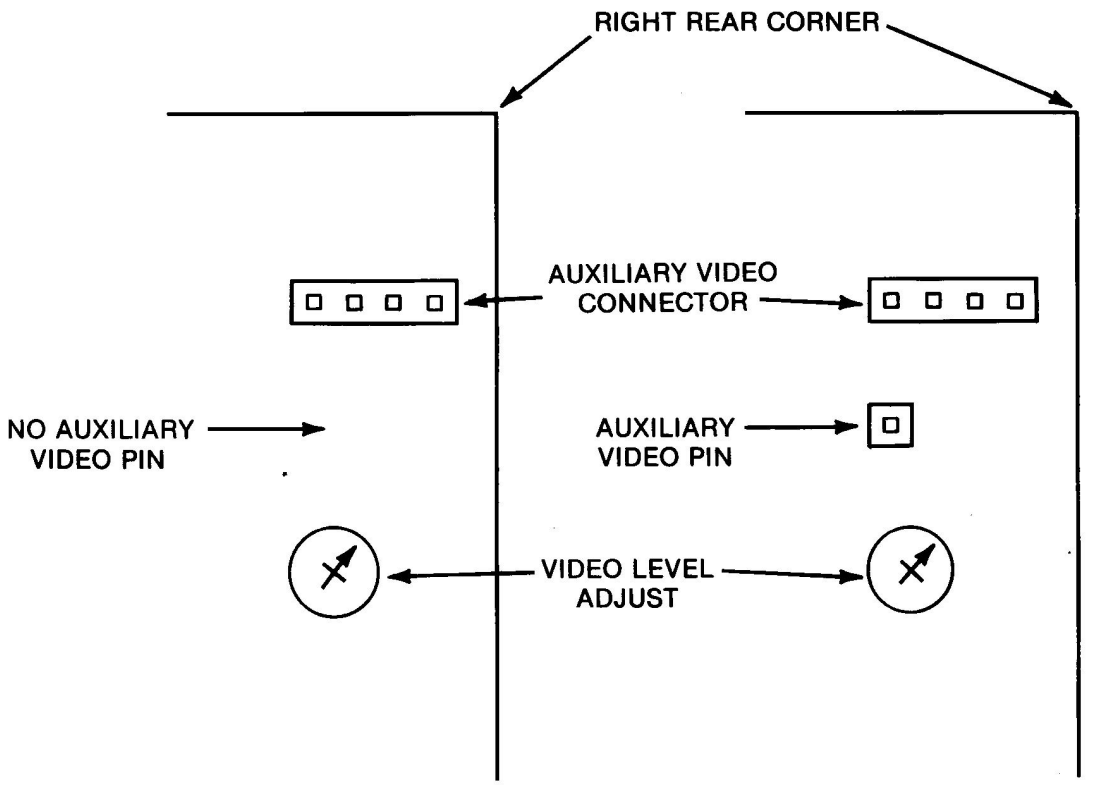


Fig. B-1. Identification of early non-RFI mother boards.

Non-RFI revisions 0 through 6 all have Memory Select Jumpers at locations D1, E1, and F1. Revision 7 does not have these jumpers.

There are no significant circuit differences between the revision 1 through 6 mother boards. Also, we have not found any distinguishing visible features (other than the part number) that can be used to differentiate these revisions. *In this book,*

we refer to the revision 1 through 6 non-RFI mother boards as simply "Rev. 1." Similarly, *revision 7 and higher non-RFI boards are referred to as "Rev. 7."*

RFI—The RFI mother boards are identified by the part number 820-0044-XX. The "XX" is the revision and to date (July 1982) the latest revision is "D." There have been previous revisions of "01" and "C." The part number is found near the F1 location (left edge of board). The differences between RFI revisions involve locations and quantities of ferrite beads and other changes intended to further suppress RFI emissions. To date there have been no functional circuit changes between RFI revisions. *We refer to all RFI mother boards (through revision D) as simply "RFI."*

Keyboard

Single-piece—The early Apple II computers contain a single-piece keyboard; the keys and electronic components are mounted on a common PC board. This keyboard uses the National Semiconductor MM5740-AAE encoder IC which is no longer manufactured. The single-piece keyboard is described in this appendix.

Two-piece—At about the time that Apple introduced the Rev. 7 non-RFI mother board, they also introduced a two-piece keyboard. In this design, the keys are mounted on one PC board, and the electronic components are mounted on another. The two boards then plug together. There have been revisions of the board containing the keys. These revisions allow for key-switches from different manufacturers. These different boards, however, are all plug-compatible with the electronics board. The two-piece keyboard uses the General Instrument AY-5-3600 encoder IC and is described in Chapter 7.

Apple II Plus

The Apple II Plus is not a revision in the same sense as the mother board circuit revisions listed in this appendix. The difference between the Apple II and the Apple II Plus is the choice of firmware plugged into the mother board. Integer BASIC is on the Apple II mother board, and Applesoft BASIC is on the Apple II Plus mother board. In this book we have made no distinction between the Apple II and the Apple II Plus.

DETAILED CIRCUIT ANALYSIS

In this section we work backwards in time, describing circuit differences between revisions of the Apple. Remember that the RFI mother boards (part numbers 820-0044-01, -C, and -D) are described in the body of the book. It is with respect to the RFI board that we describe circuit variations.

Revision 7 Non-RFI Mother Board

The revision 7 non-RFI mother boards are identified by part numbers 820-0001-07 and up.

Color Killer—Early Rev. 7 boards do not have IC A14 (Fig. C-19*). These boards use transistor Q6 as a color killer. When TEXT MODE is high, base current through R27 drives Q6 into saturation. This shunts COLOR BURST to ground. This use of Q6 is not fully effective and some tv sets still display tinted text characters. In later Rev. 7 boards, A14-1 and A14-4 are added to force COLOR BURST to a low when TEXT MODE is high. For these boards, the last waveform of Fig. 4-3 is always low. Note that A14 is inserted into the circuit differently on Rev. 7 boards than it is on RFI boards. Also note the use of COLOR REF versus $\overline{\text{COLOR REF}}$.

Soft 5—Rev. 7 and earlier boards use high outputs of gates A2-8 and A2-11 instead of R28 to pull up unused logic inputs (Fig. C-2*).

Data Transceiver—Rev. 7 and earlier boards use two 8T28 quad transceivers on the data bus (H10 and H11 in Fig. C-8) instead of a single 8304 octal transceiver.

H2 Inverter—Rev. 7 boards use Q7 and associated resistors (Fig. C-19*) instead of A14-13 (Fig. C-20*) to invert signal H2. Q7 is located near IC B13.

Vertical Sync—Rev. 7 boards do not use A14-4 and A14-10 in the vertical sync circuit (Fig. C-19*). This results in a slightly different vertical sync pulse waveform as shown in Fig. B-2*. The waveforms that differ between Rev. 7 and RFI boards are B14-1, C13-C, SYNC, and the COMPOSITE VIDEO OUTPUT. For Rev. 7, Fig. B-2* replaces the vertical sync portion of Fig. 4-2*.

Auxiliary Video Pin—The Auxiliary Video Pin on some Rev. 7 boards is mounted in a four-pin connector as shown in Fig. C-19*. Two pins in this connector connect to unused solder pads 3 and 4 nearby. The fourth pin connects to signal AN2 and solder pad 2.

Video Filter—Rev. 7 and earlier boards do not have the high frequency filter consisting of L7 and C16 in the composite video output (Fig. C-19* and C-20*).

Miscellaneous—On Rev. 7 boards, the Clear input for text mode shift register A3 connects to SOFT 5 via bow tie 5.

Revision 1 Non-RFI Mother Board

The revision 1 non-RFI mother boards are identified by part numbers 820-0001-01 through -06.

Color Killer—Rev. 1 boards use Q6 as their only color killer. The operation of Q6 was described previously under Rev. 7.

Text Points—Rev. 1 and Rev. 0 boards do not have test points 3 and 4 or solder pad 1 (Figs. C-2*, C-12*, and C-16*).

Memory Jumper Blocks—Rev. 1 and Rev. 0 boards have 16-pin Memory Jumper Blocks at locations D1, E1, and F1 (Fig. C-4*). These blocks allow 16K-bit *or* 4K-bit RAMs to be used. The jumper blocks in Fig. C-4* are drawn in the configuration for use of *all* 16K RAMs. In this configuration, the memory system operates as described in Chapter 5. The following ICs and gates are part of the 4K/16K memory arrangement of Rev. 1 and Rev. 0 boards: H1-11, H1-3, C14-3, and E2. The IC E2 may be removed if you are using all 16K RAMs in a Rev. 1 or Rev. 0 Apple.

RAM Address 6 Termination—R31 and R32 which terminate signal line RA6 are not present on Rev. 1 or Rev. 0 boards.

Solder Pads 7 and 8—Solder pads 7 and 8 (Fig. C-11*) are not present on Rev. 1 or Rev. 0 boards. These pads are used to connect signals to peripheral I/O slot 7 for use by European color video generators.

Cassette Input—The gain of the cassette input circuit is different on Rev. 0 and some Rev. 1 boards (Fig. C-12*).

Test Connector—Rev. 1 and Rev. 0 boards do not have a multipin test connector near IC B2 (Fig. C-3*).

Character Generator—On Rev. 1 and Rev. 0 boards, IC A5 is a 2513 character generator (Fig. C-15*). A 2513 is really a ROM containing the same set of 64 ASCII characters that have been programmed into the 2316B ROM of the Rev. 7 and RFI boards. The 2513 cannot be directly replaced by a user-supplied EPROM, however, due to its supply voltages and pinouts.

The 2513 supplies only five outputs; the five horizontal dots of a character. As a result, the B and H inputs of shift register A3 are tied to ground. Bow ties 9, 10, and

11 are not present on Rev. 1 or Rev. 0 boards. Apart from these differences, the text mode operation of Rev. 1 and Rev. 0 boards is the same as described in Chapter 7.

Video Sync—Rev. 1 mother boards do not have IC A14 (compare Figs. C-18* and C-20*.) The result is a horizontal sync pulse twice as wide on Rev. 1 as on Rev. 7 and RFI boards. Fig. B-3 shows as typical the horizontal sync pulse that follows the vertical sync pulse. When dealing with a Rev. 1 board, use Fig. B-3 to replace the appropriate parts of Fig. 4-2*.

Fig. B-4 *shows the wider sync pulse and COLOR BURST in detail. When dealing with Rev. 1, Fig. B-4 replaces Fig. 4-3. In summary, the video signals that differ between Rev. 1 and RFI boards are B14-1, C13-C, C13-D, SYNC, COLOR BURST, and the COMPOSITE VIDEO OUTPUT.

Miscellaneous—The following items also differ between Rev. 1 and RFI boards: SOFT 5, Data Transceiver, and Video Filter. These items were described previously under Rev. 7.

Revision 0 Non-RFI Mother Board

The revision 0 non-RFI mother boards are identified as shown in Fig. B-1.

Power-On Reset—The Rev. 0 mother board does not have a power-on reset circuit. This function is provided by type 555 timer A13 on Rev. 1 and later boards (Fig. C-13*). This means that Rev. 0 users may have to press the reset key to initialize the 6502. This is not necessary, however, if a peripheral is installed that drives a reset pulse onto the reset line at power up. The Apple II floppy disk controller card is such an example.

Without power-on reset, the keyboard strobe flip-flop (B10-9) may be set when power is applied to a Rev. 0 board. As a result, the first keyboard command may contain an extra leading character. Note: If an autostart monitor ROM is installed in a Rev. 0 board, reset routines in the ROM will reset the keyboard strobe.

Color Killer—Rev. 0 boards use neither IC A14 nor transistor Q6 for a color killer. Without a color killer, COLOR BURST is not suppressed in text mode. This results in text characters with a color tint or fringe when using a color tv set.

Four HIRES Colors—The Rev. 0 mother board displays only four HIRES colors: black, white, violet, and green. By comparing Figs. C-14* and C-15*, you can see that some components are absent from Rev. 0. The key component not present is flip-flop A11-9. Recall from the HIRES discussion in Chapter 8 that A11-9 provides a delayed version of the HIRES serial bit stream from B4-15. Data selector A9 then selects either the delayed or nondelayed bit stream under control of DL7. When the delayed bit stream is selected, blue and orange are provided.

On Rev. 0 boards, there is no delayed bit stream, so there are no blue and orange colors. Data bit DL7 is ignored in HIRES mode on a Rev. 0 mother board.

Also on Rev. 0 boards, DL7 does not connect to A9 (this connection is via A8 and A10 on Rev. 1 and later boards). The data and select inputs of A9 are also arranged differently on Rev. 0 boards, with A12-10, A12-13, and A11-5 absent. Apart from the two missing HIRES colors, the graphics and text operation of Rev. 0 is the same as that of the RFI mother board described in Chapter 8.

24K Memory Size Problem—There is a problem on the Rev. 0 mother board that affects Apples with 20K or 24K bytes of RAM. The problem causes BASIC to "think" that there is more RAM in the computer than there is. This problem is corrected on the Rev. 1 board by gate C14-3 (Fig. C-4*). See page 72 of the *Apple II Reference Manual* for more details.

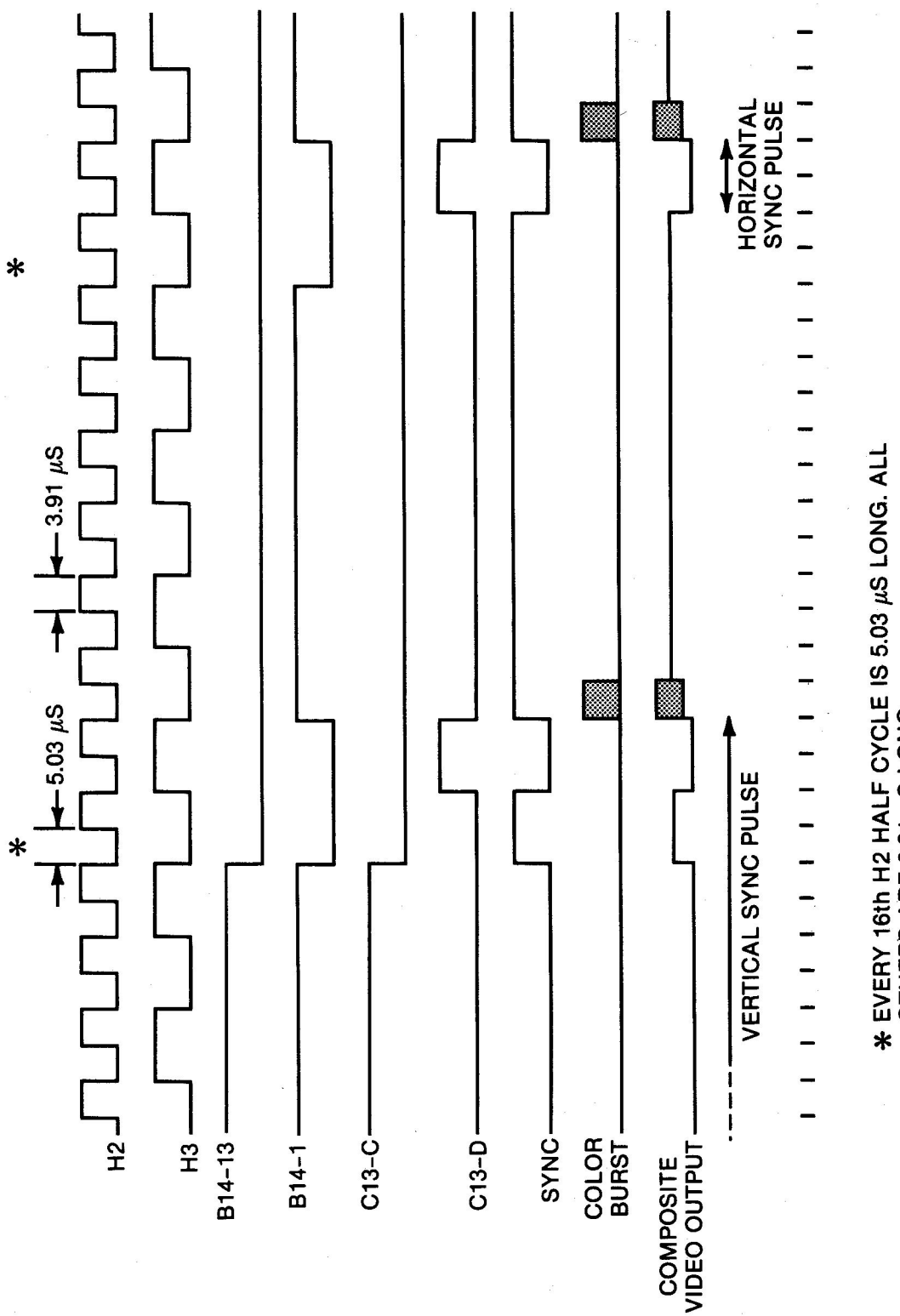


Fig. B-3. Vertical and horizontal sync—Rev. 1.

European TV Standards—Bow ties 12 and 16, and solder pads 13, 14, and 15 on Rev. 1 and later boards can be rearranged to convert the vertical frequency to 50 Hz for use with European tv sets. These pads and bow ties are not present on Rev. 0 boards. The circuit differences may be seen in Figs. C-3*, C-17*, and C-18*. The operation of the video address generator on a Rev. 0 board is the same as described in Chapters 3 and 4 for RFI boards.

Auxiliary Video Pin—Rev. 0 boards do not have an Auxiliary Video Pin (compare Figs. C-17* and C-18*).

Slot 7—Signals SYNC and COLOR REF do not connect to peripheral I/O slot 7 on Rev. 0 boards (Fig. C-11*).

Video Sync—Rev. 0 boards do not have ICs A14 and B14 (compare Figs. C-17* and C-20*). Since there is no A14, the Rev. 0 horizontal sync pulse is twice as long as the RFI horizontal sync pulse (see Figs. B-4 and B-6*). When dealing with the Rev. 0 mother board, substitute Fig. B-4 for Fig. 4-3.

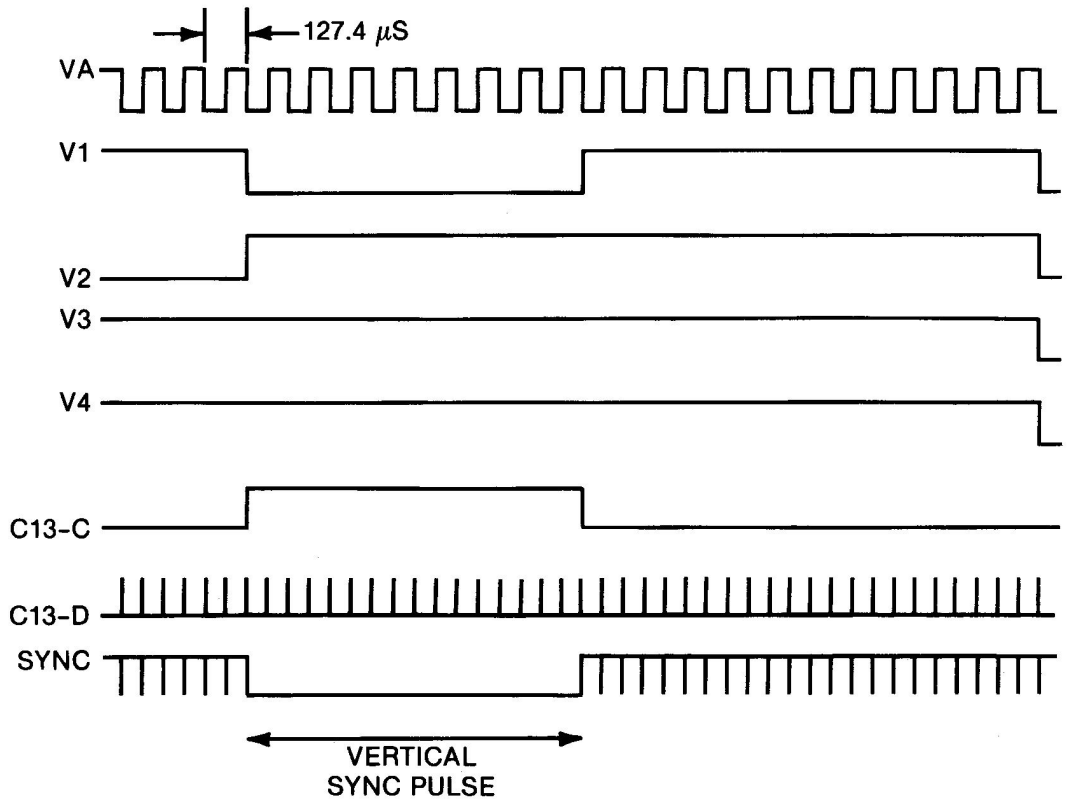


Fig. B-5. Vertical sync—Rev. 0.

Since there is no B14, the Rev. 0 vertical sync pulse has no serrations and is four times as long as the RFI vertical sync pulse (compare Figs. B-5 and 4-1*). Note that to the scale of the figures, the Rev. 0 and RFI vertical sync pulses both start at the same time relative to the video address. When dealing with the Rev. 0 mother board, substitute Figs. B-5 and B-6 for the appropriate sections of Figs. 4-1* and 4-2*.

Miscellaneous—The following items also differ between Rev. 0 and RFI boards (they were described previously under Rev. 1):

Test Points
Memory Jumper Blocks

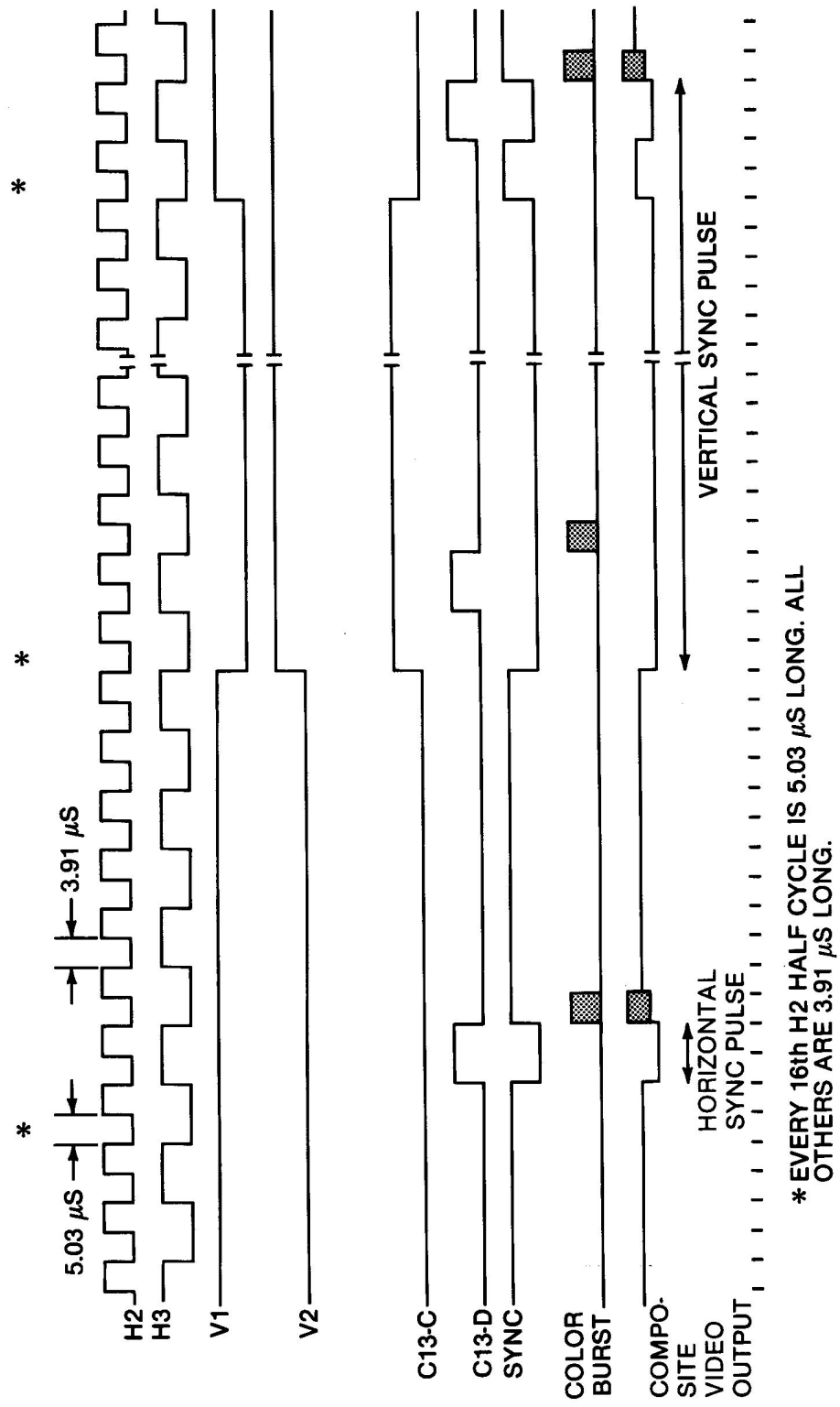


Fig. B-6. Vertical and horizontal sync—Rev. 0.

RA6 Terminator
Solder Pads 7 and 8
Cassette Input
Test Connector
Character Generator

The following additional items of difference between Rev. 0 and RFI boards were described previously under Rev. 7:

SOFT 5
Data Transceiver
Video Filter

Table B-1 lists a summary of the major differences between mother board revisions.

Table B.1 Summary of Major Mother Board Revisions

	Rev. 0	Rev. 1	Rev. 7	RFI
Color Killer	No	Q6	Q6 or A14-4	A14-1
Horizontal Sync Pulse	7.8 μS	7.8 μS	3.9 μS	3.9 μS
Vertical Sync Pulse	1.02 mS	255 μS	255 μS	255 μS
Vertical Serrations	No	Yes	Yes	Yes
H2 Inverter	None	None	Q7	A14-13
Auxiliary Video Pin	No	Yes	Yes	Yes
Optional 50 Hz Video	No	Yes	Yes	Yes
SYNC and COLOR REF on Slot 7	No	Yes	Yes	Yes
Solder Pads 7 and 8	No	No	Yes	Yes
HIRES Colors	4	6	6	6
Power-on Reset	No	Yes	Yes	Yes
Character Generator	2513	2513	2316B	2316B
Memory Jumper Blocks	Yes	Yes	No	No
Data Bus Transceiver	8T28	8T28	8T28	8304
SOFT 5	A2-8 & A2-11	A2-8 & A2-11	A2-8 & A2-11	R28

Single-Piece Keyboard

The single-piece keyboards used in early Apple II computers are identified by their physical arrangement. In the single-piece keyboard, the key switches and the ICs are all mounted on a common PC board. The functions performed on the single-piece keyboard are similar to those described in Chapter 7 for the two-piece keyboard. A different keyboard encoder IC is used, however, and there are other circuit differences. In this section, we provide a circuit description for the single-piece keyboard (Fig. C-21*).

(Note that *very early* single-piece keyboards use different reference designators than those shown in Fig. C-21*. These early keyboards use essentially the same parts and circuit, however. The early keyboards have a different PC layout that allows the ICs to be seen when the lid of the Apple is removed. This is how the early keyboards may be identified.)

Keyboard Encoder IC—The single-piece keyboard uses a type MM5740-AAE keyboard encoder IC, U5. The IC's 9 X outputs and 10 Y inputs are arranged in a matrix with the character keys at the cross points. Only 47 of the possible 90 cross points are occupied by keys. When a key is depressed, it connects the X line to the Y line.

Keyboard encoder U5 contains counters that count through an X sequence and a Y sequence as it scans the keyboard looking for a depressed key. These counters are driven from an external clock supplied to pin 3. An oscillator is formed by U4-6, U4-8, U4-11, C6, R6, and R7 that provides this clock. Its frequency (about 50 kHz) is not critical, but it must be between 10 kHz and 200 kHz for U5 to operate properly.

Fig. B-7 shows the clock and the sequence of pulses generated on U5's X outputs. When a key is depressed, one of the X outputs connects to a Y input. When this Y input is scanned, the key closure will be detected. Keyboard encoder U5 then looks at the SHIFT and CTRL key inputs and encodes the key into the proper ASCII character. The character is latched and appears inverted at U5's output (B1 through B9). The Apple II uses only the lower seven bits.

Gates U2 and U1-8 buffer the character and invert it to positive-true. The character then finds its way through the flat cable to location A7 on the mother board. Since the character was latched inside U5, it will remain at U5's output until another key is depressed.

Each time a key is closed, U5 outputs a pulse on its Data Strobe Output (pin 13). Pin 13 connects to pin 14 (Data Strobe Control), an arrangement that sets the pulse length at one clock period.

Sequence of Operation—Follow along on Fig. B-7 as we describe the timing. We will assume that the previous character was a "B." The ASCII LSB for "B" is 0, so U2-8 will be latched low from the previous character (point A). Now suppose you depress the "A" key. From the keyboard matrix we see that the "A" key connects X9 to Y4. Thus, as long as the "A" key is down, Y4 (point B) will be the same as X9 (point C). We will further assume that the Y scan is currently looking at Y4. This means the pulse on Y4 at point B will be detected by U5. Since the ASCII LSB for "A" is 1, U2-8 will go high (point D).

On the second clock falling pulse after detection (point E), U5 will generate a strobe pulse (point F). The pulse is one clock period long. The strobe is buffered by U1-6 and U1-3, then finds its way via the flat cable to location A7 on the mother board. There it sets flip-flop B10-9, point G. Chapter 6 covered the reading of the keyboard as far as circuitry on the mother board is concerned.

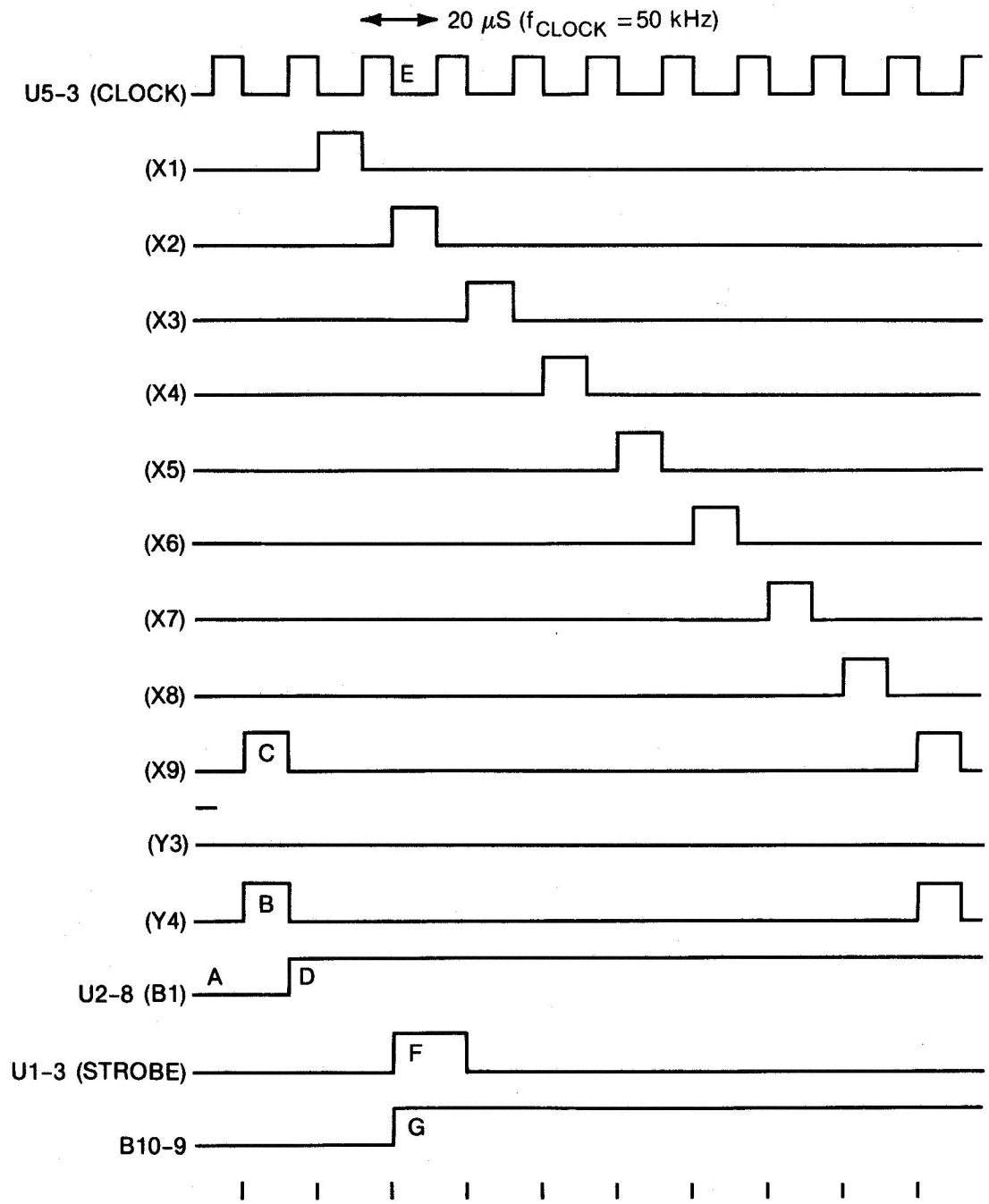


Fig. B-7. X scan—single-piece keyboard.

Fig. B-8* is another view of the signals shown expanded in Fig. B-7. In Fig. B-8* we can see one complete scan of the Y inputs. Again let's assume that "B" was the previous character so that B1 is low at point A. Sometime around point B we depress the "A" key. This connects Y4 to X9, and these two signals will be the same as long as the key is down. The key is not recognized until the Y scan reaches Y4 (point C). At this point, B1 goes high and there is a strobe pulse as was shown expanded in Fig. B-7.

The length of time it takes to detect a key closure is a function of where in the scan the key is depressed. The time ranges from 1 to 90 clock periods.

Non-TTL Levels—U5's X outputs and Y inputs are not at TTL levels. Their typical voltage levels are shown in Fig. B-9.

Contact Bounce—In most mechanical keyboards there is a possibility of contact bounce when a key is depressed and again when it is released. The bounce (rapid on and off fluctuations) can cause extra characters to be generated. Keyboard encoder U5 alleviates this problem by ignoring key closures for 8 mS after a key is first closed and again for 8 mS after a key is opened. This is accomplished by connecting C4 to U5's Key Bounce Mask pin. Fig. B-10 shows the timing and waveforms on C4.

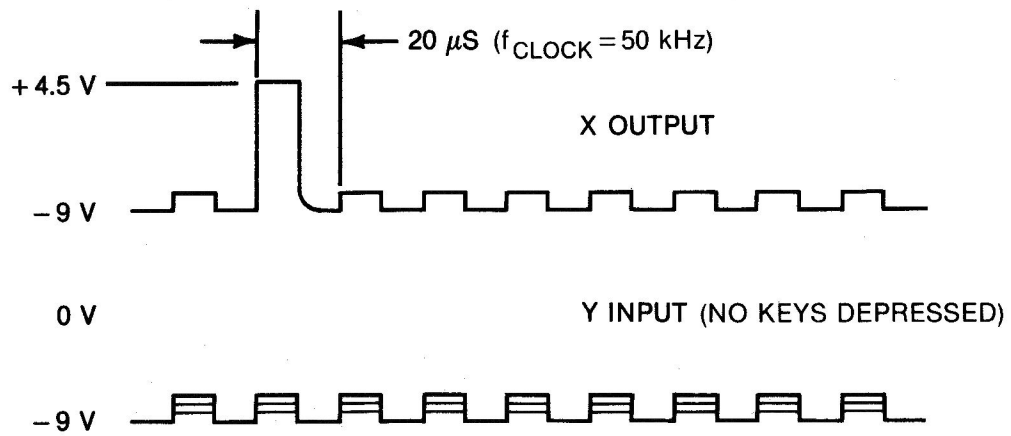


Fig. B-9. Encoder waveforms—single-piece keyboard.

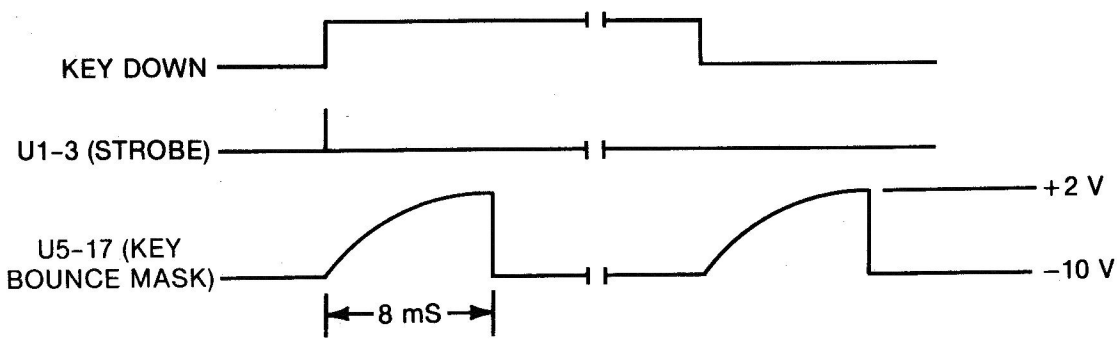


Fig. B-10. Key bounce mask—single-piece keyboard.

Repeat Function—U3 is a type 555 timer connected to oscillate at about 10 Hz as set by R3, R4, and C2 (Fig. C-21). U3 is normally held reset by R2. When the REPT key is depressed, the reset is removed and the resulting 10 Hz signal is applied to U5's Repeat input (pin 16). During each positive half cycle of the Repeat input, U5 will output a data strobe if a character key is also depressed.

Roll Over—The MM5740-AAE encoder IC is capable of *N-key roll over*. N-key roll over means that you can have any number of keys down simultaneously, and the encoder will correctly send the characters in the order keyed. This capability, however, can be used only if there is a diode at every cross point in the key matrix. The Apple II keyboard does not have these diodes. Without them, depressing three or more keys simultaneously may generate extraneous or *phantom* characters. As a result, the Apple II keyboard is described as having *2-key roll over*. No more than two keys may be depressed simultaneously if phantom characters are to be avoided.

The sneak paths that are possible in a matrix without diodes can be used in the

single-piece keyboard to generate characters not normally available from the keyboard as shown in Table B-2.

Table B-2. Generate Special Characters

Character	Hold down these keys simultaneously
_ (Underscore)	SHIFT, U, I, Y
[(Left Bracket)	SHIFT, U, I, J
\ (Backslash)	SHIFT, U, I, H

With the proper editing, you can insert these characters into programs directly from the keyboard.

Quad-Mode Operation—U5 is a quad-mode encoder. This means each key can generate up to four different characters as a function of the SHIFT and CTRL keys (see Table B-3).

Table B-3. CHARACTERS as a Function of Control and Shift

U5-19 (CTRL)	U5-21 (SHIFT)	Character Group
0	0	Unshift
0	1	Shift
1	0	Control
1	1	Shift-Control

The four characters for each key are shown in Fig. C-21*. Only the "P," "M," and "N" keys can generate four different characters. The rest generate one or two.

Initialization—C7 and R8 form a delay circuit that initializes U5 on power up by taking SHIFT high for about 5 mS. Diode CR1 provides a rapid discharge path for C7.

APPENDIX C

Apple II Schematics

This appendix contains schematics for the Apple II mother board, keyboard, and power supply. Mother board schematics are provided for circuits dating from the earliest Apple II to the most recent revision available for publication. The paragraphs that follow contain information intended to help you use the schematics.

REVISIONS

Mother Boards

As described in Appendix B, the earliest Apple II mother board is known as Rev. 0. A short time after the Apple's introduction, several circuit changes resulted in the Rev. 1 mother board. Rev. 1 includes a color killer and two more HIRES colors. There were no more significant circuit changes until the introduction of the Rev. 7 mother board. Rev. 7 deletes the Memory Jumper Blocks and changes the character generator ROM. The next significant change involved modification of the mother board and packaging to reduce RFI. Some minor circuit changes were made at the same time. The major revisions of the mother board are denoted in this book as Rev. 0, Rev. 1, Rev. 7, and RFI.

Appendix B contains additional information on the differences between revisions. It also describes how you can examine the mother board to determine the revision of a particular Apple II. Note that the circuit description in the body of this book corresponds to the RFI schematics. Material in Appendix B describes the differences in operation of Revisions 0, 1, and 7 relative to the operation of the RFI mother board.

A notation on each schematic indicates the revision to which the schematic applies. Most schematics apply to two or more revisions. Minor circuit differences between the revisions are indicated by notes or insets on the drawing. When there are major differences between revisions, separate schematics are used for each revision. As you trace a signal from schematic to schematic, be sure to use the drawing that corresponds to the revision of interest.

Keyboards

The original Apple II keyboard was constructed on a single printed circuit board and is thus referred to as the "single-piece keyboard." Apple later introduced a "two-piece keyboard." The two-piece keyboard contains one printed circuit board on which the keys are mounted. A second printed circuit board containing the ICs then plugs onto the first printed circuit board. This appendix contains schematics for both the single- and two-piece keyboards.

SYMBOLS

Fig. C-1 shows some of the symbols used on the schematics. When a signal enters or exits a schematic, the signal name is shown beside an arrow symbol (Examples a through d). The direction of the arrow indicates the direction of signal flow. A double-headed arrow indicates bidirectional signal flow, such as on a data bus (Example c). If the signal is part of the system bus, a number inside the arrow symbol gives the peripheral I/O connector pin on which the signal appears (Examples c and d). A "To" or "From" notation beside a signal name indicates the other schematics on which the signal appears. The designations "Part 1" and "Part 2" refer to the two separate drawings used to show some schematics. For example, the video generator schematic is divided into two parts.

Logic functions that are active on a low level are shown by a small circle or

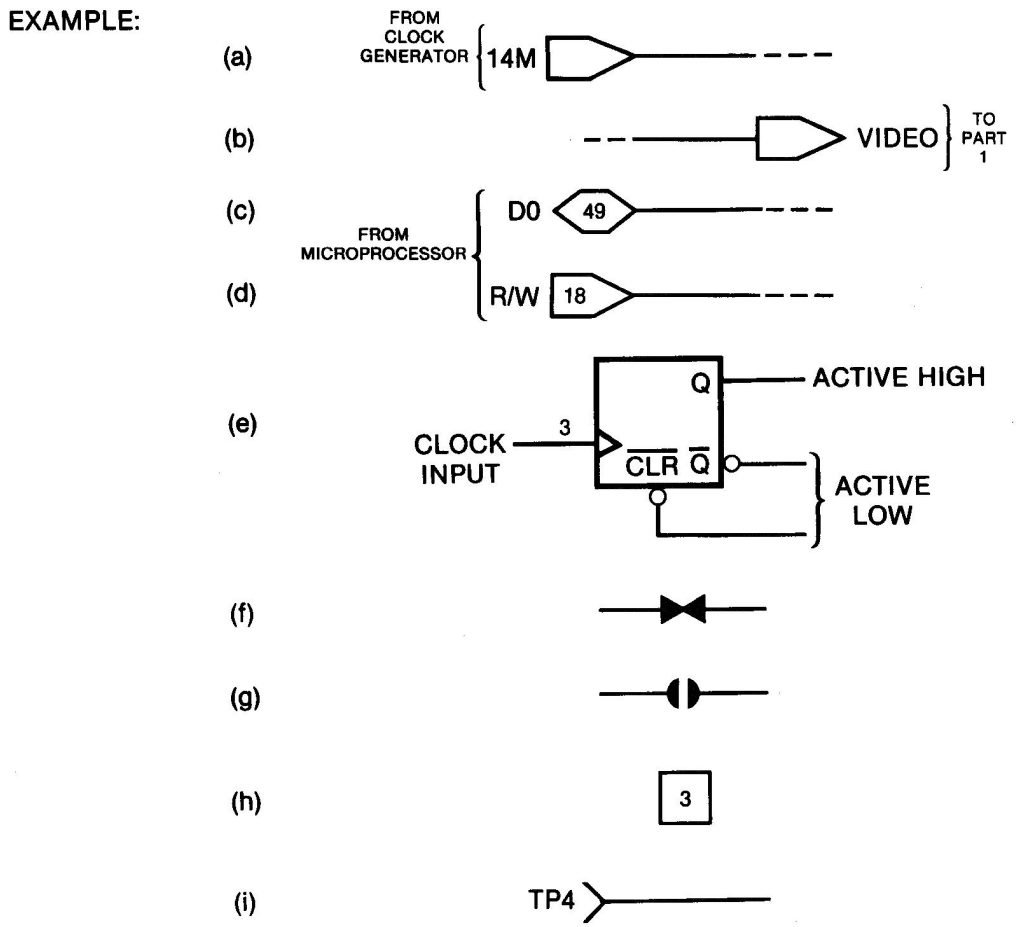


Fig. C-1. Schematic symbols.

"bubble." For example, see the $\overline{Q}$ output and $\overline{CLR}$ input in Example e. The symbol at pin 3 is a clock input.

The symbols in Examples f and g represent the shapes of printed circuit board traces that are meant to be altered in order to modify the circuit. The *bow tie* in Example f would be cut between the arrows to open the circuit. The semicircles in Example g would be bridged with solder to close the circuit.

A number in a square (Example h) refers to a note shown elsewhere on the drawing. The symbol in Example i represents a test point or other labeled solder point on the circuit board.

When an IC is powered from +5 volts and ground, and these voltages connect to the usual opposite corner pins of the IC, then the power pins are not shown on the schematic. If the +5 volts connects to an unusual pin or if the IC requires multiple voltages, then the power pins *are* shown. In general, bypass capacitors, ferrite beads, and isolated grounds are not depicted in the drawings.

The Apple II schematics are shown in Figs. C-2* through C-23*.

ACKNOWLEDGMENTS

The schematics in this appendix have been used with the permission of Apple Computer, Inc. They have been redrawn from the following Apple Computer documents:

Revision 0 Mother board	*Apple II Reference Manual* ("Red Book"), Jan. 1978
Revision 1 Mother board, Power Supply, Single-piece Keyboard	*Apple II Reference Manual*, 1979
Revision 7 Mother board	*The Apple II Revision 07 Main Board*
RFI Mother board	*Addendum to the Apple II Reference Manual*
Two-piece Keyboard	*Schematic, A-2 Keyboard & Encoder*, Drawing No. 050-0021-D

APPENDIX D

References

Chapter 1

1.1 Espinosa, Christopher. *Apple II Reference Manual*, Apple Computer, Inc., 1979.
1.2 *The TTL Data Book for Design Engineers*, 2nd ed., Texas Instruments, 1976.

Chapter 2

2.1 United States Patent No. 4,130,862, "DC Power Supply," Inventor: Frederick R. Holt, Assignee: Apple Computer, Inc., 19 Dec., 1978.
2.2 Wozniak, Stephen. "System Description—The Apple II," *Byte,* May 1977.

Chapter 3

3.1 United States Patent No. 4,136,359, "Microcomputer for use with Video Display," Inventor: Stephen G. Wozniak, Assignee: Apple Computer, Inc., 23 Jan., 1979.
3.2 Wozniak, Stephen. "The Impossible Dream: Computing *e* to 116,000 Places with a Personal Computer, "*Byte*, June 1981.

Chapter 5

5.1 *Memory Data Book and Designers Guide*, Mostek, 1979.

Chapter 6

6.1 "Apple II Product Specifications, Hobby/Prototyping Board," Apple Computer, Inc.
6.2 "Applications Information AN2—SY6500 Microprocessor Family," Microprocessor Products, Synertek, Jan. 1980.
6.3 DeJong, Marvin L. *Programming & Interfacing the 6502, with Experiments*, Howard W. Sams & Co., Inc., 1980.
6.4 Fischer, Dan and Caffrey, Morgan P. "Go Ahead and Interrupt your Apple," Parts 1 and 2, *Softalk,* Mar. and Apr. 1982.
6.5 Holland, John M. *Advanced 6502 Interfacing*, Howard W. Sams & Co., Inc., 1982.

6.6 *MCS 6500 Microcomputer Family Hardware Manual*, Commodore Business Machines, Inc. (MOS Technology), Jan. 1976.
6.7 *MCS 6500 Microcomputer Family Programming Manual*, Commodore Business Machines, Inc. (MOS Technology), Jan. 1976.
6.8 *R6500 Microcomputer System Hardware Manual*, Rockwell International, Aug. 1978.
6.9 "R6500 Microcomputer System Data Sheet," Rockwell International, Nov. 1981.
6.10 *R6500 Microcomputer System Programming Manual*, Rockwell International, Feb. 1979.
6.11 Scanlon, Leo J. *6502 Software Design*, Howard W. Sams & Co., Inc., 1980.
6.12 "6500 Microprocessors" (Data Sheet), Commodore Semiconductor Group, Feb. 1981.
6.13 *SY6500/MCS6500 Microcomputer Family Hardware Manual*, Synertek, Aug. 1976.
6.14 *SY6500/MCS6500 Microcomputer Family Programming Manual*, Synertek, Aug. 1976.
6.15 SY6500 Data Sheet, "8-Bit Microprocessor Family," Synertek, Apr. 1980.
6.16 Titus, Jonathan S., Larsen, David G., and Titus, Christopher A. *Apple Interfacing*, Howard W. Sams & Co., Inc., 1981.
6.17 White, George M. "Using Interrupts on the Apple II System," *Byte*, May 1981.

Chapter 7

7.1 *Apple II Monitors Peeled*, Apple Computer, Inc., 1981.
7.2 "AY-5-3600 Keyboard Encoder Operation," Bulletin 1402, General Instrument Corp.
7.3 *Data Catalog*, Standard Microsystems Corp., 1981 (KR3600 Keyboard Encoder).
7.4 Lancaster, Don. *TV Typewriter Cookbook*, Howard W. Sams & Co., Inc., 1976.
7.5 *Microelectronics Product Catalog*, General Instrument Corp., 1980 (AY-5-3600 Data Sheet).
7.6 "The Apple II Cassette Interface," *Apple Orchard*, Spring 1981.

Chapter 8

8.1 Bishop, Bob. "Apple II HIRES Graphics: Resolving the Resolution Myth," *Apple Orchard*, Fall 1980.
8.2 Rowe, Pete. "The Mysterious Orange Vertical Line," *Apple Orchard*, Fall 1980.
8.3 United States Patent No. 4,278,972, "Digitally Controlled Color Signal Generation Means for use with Display," Inventor: Stephen G. Wozniak, Assignee: Apple Computer, Inc., 14 July, 1981.

Appendix A

A.1 Bierman, Howard and Bierman, Marvin. *Color Television Principles and Servicing*, Hayden Book Co., Inc., 1973.
A.2 "EIA Television Systems Bulletin No. 4, EIA Recommended Practice for Horizontal Sync, Horizontal Blanking and Burst Timing in Television Broadcasting," EIA, Mar. 1976.
A.3 "EIA Standard RS-170, Electrical Performance Standards—Monochrome Television Studio Facilities," EIA, Nov. 1957.
A.4 "EIA Standard RS-189-A, Encoded Color Bar Signal," EIA, July 1976.

A.5 "EIA Standard RS-330, Electrical Performance Standards for Closed Circuit Television Camera 525/60 Interlaced 2:1," EIA, Nov. 1966.
A.6 Grob, Bernard, *Basic Television Principles and Servicing*, McGraw-Hill, 1975.
A.7 "Industrial Electronics Bulletin No. 1, Closed Circuit Television—Definitions," Electronics Industries Association, Sept. 1962.
A.8 "19-Inch Color TV's," *Consumer Reports*, Jan. 1981.
A.9 Schure, Alexander. *Basic Television*, Vol. 6, Hayden Book Co., Inc., 1975.

Appendix B

B.1 "Addendum to the Apple II Reference Manual," Apple Part No. 031-0004-B, Apple Computer, Inc. (Reduced RFI Mother Board).
B.2 *Apple II Reference Manual* (Red Book), Apple Computer, Inc., Jan. 1978.
B.3 "Application Note AN-80, MOS Keyboard Encoding," National Semiconductor Corp., April 1973.
B.4 "Data Sheet, MM5740 90-Key Keyboard Encoder," National Semiconductor Corp., 1977.
B.5 "The Apple II Revision 07 Main Board," Apple Part No. 031-0072-00, Apple Computer, Inc.
B.6 "2513 Data Sheet—High Speed 64 × 7 × 5 Character Generator," Signetics.

Index

TO THE READER

Sams Computer books cover Fundamentals — Programming — Interfacing — Technology written to meet the needs of computer engineers, professionals, scientists, technicians, students, educators, business owners, personal computerists and home hobbyists.

Our Tradition is to meet your needs and in so doing we invite you to tell us what your needs and interests are by completing the following:

1. I need books on the following topics:

_____

_____

_____

_____

_____

_____

_____

_____

2. I have the following Sams titles:

_____

_____

_____

_____

_____

_____

_____

_____

3. My occupation is:

_____ Scientist, Engineer
_____ Personal computerist
_____ Technician, Serviceman
_____ Educator
_____ Student
_____ D P Professional
_____ Business owner
_____ Computer store owner
_____ Home hobbyist
Other _____

_____

Name (print)_____

Address_____

City _____ State _____ Zip _____

Mail to: **Howard W. Sams & Co., Inc.**
Marketing Dept. #CBS1/80
4300 W. 62nd St., P.O. Box 7092
Indianapolis, Indiana 46206

21959

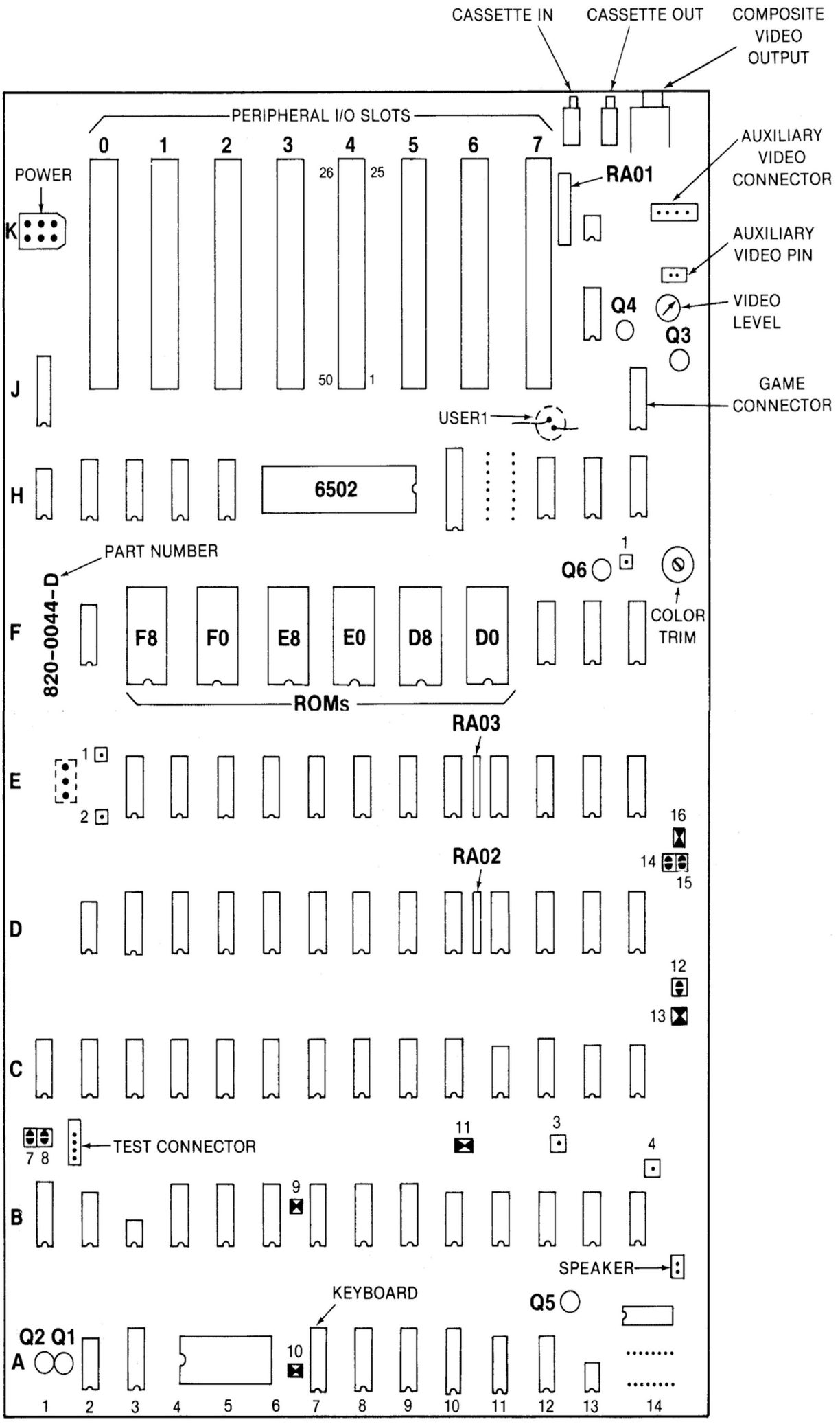


Fig. 1-1. Diagram of the Apple II RFI motherboard. (Courtesy Apple Computer Inc.)

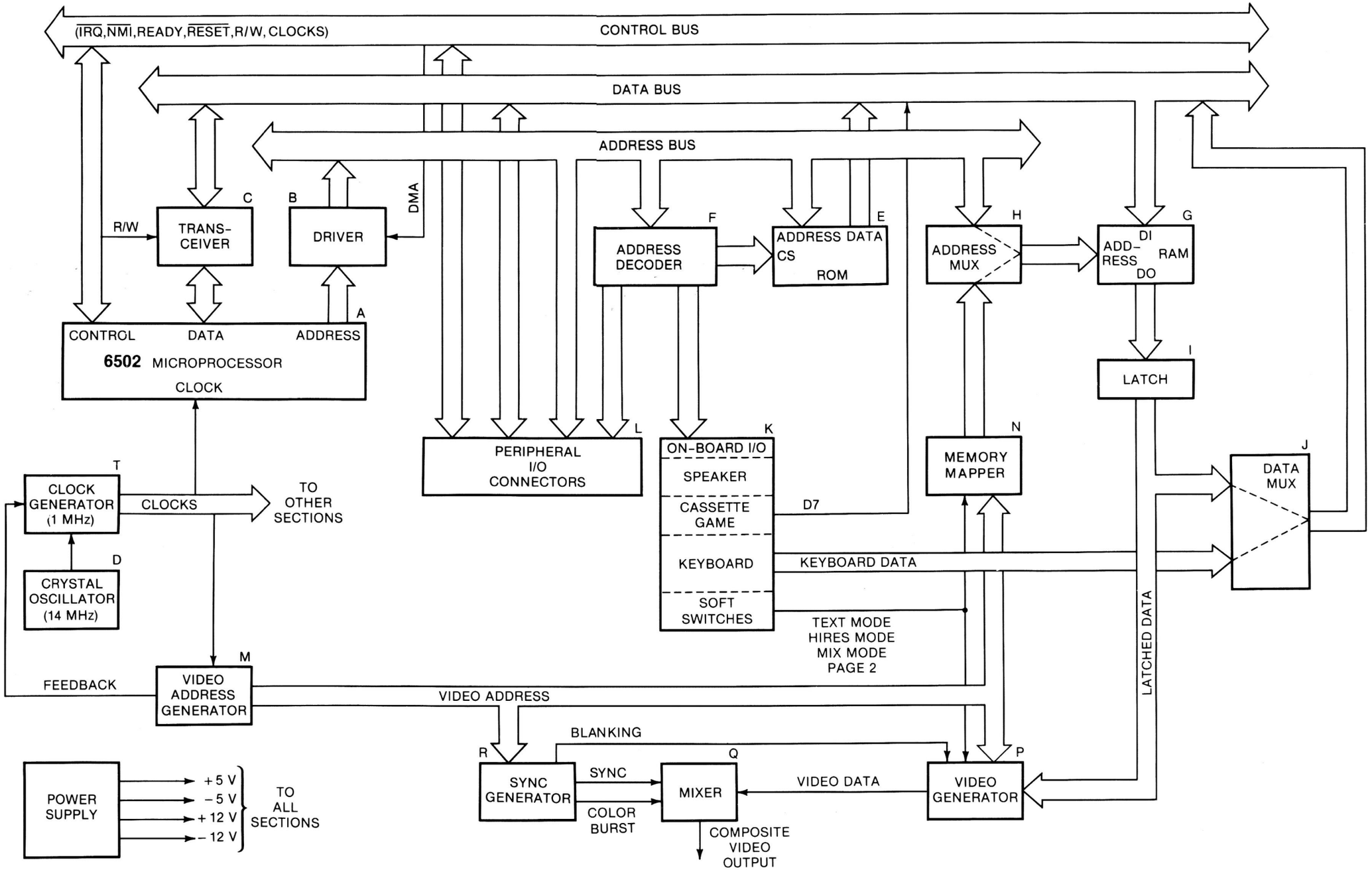


Fig. 2-1. Apple II block diagram.

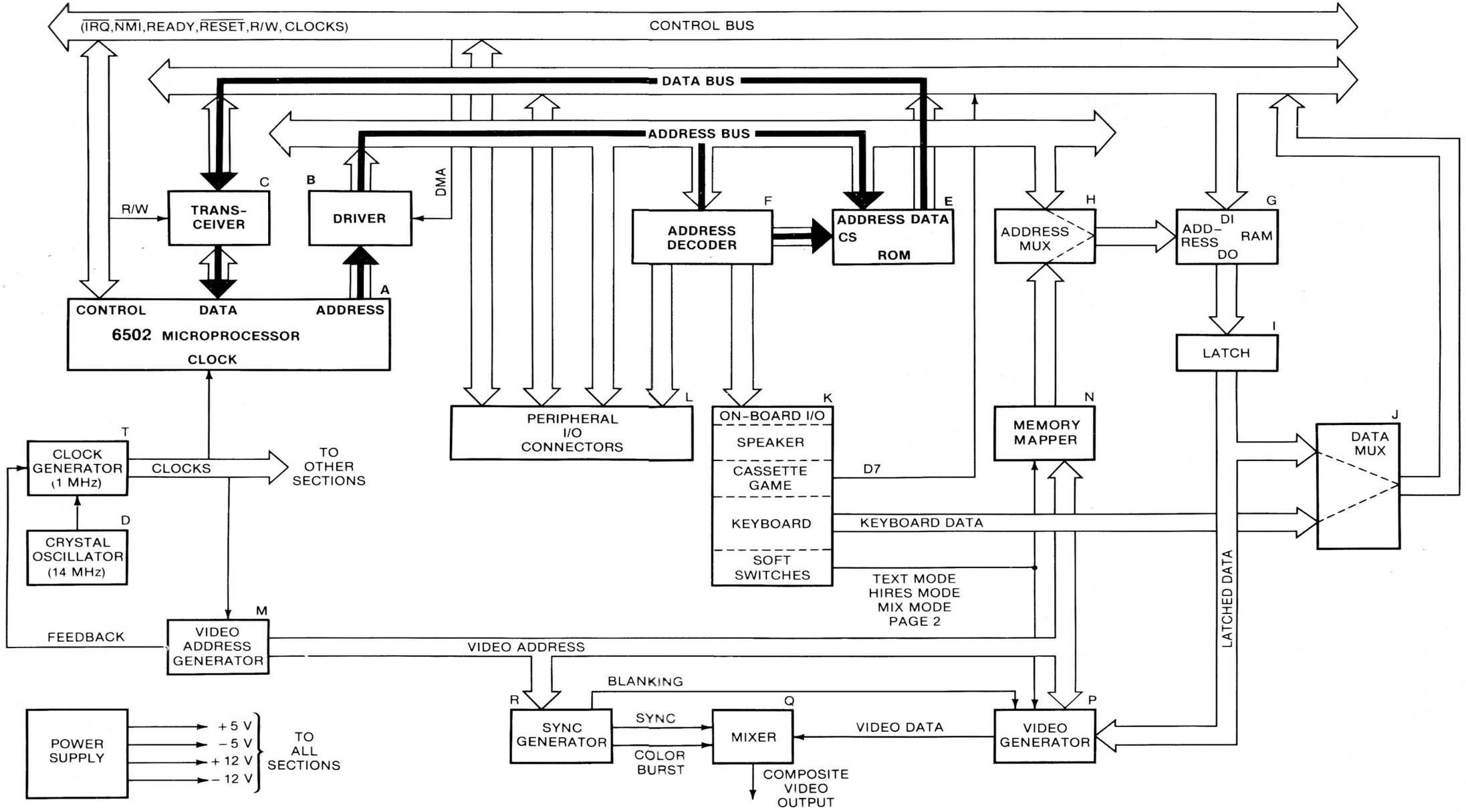


Fig. 2-2. ROM read-cycle path.

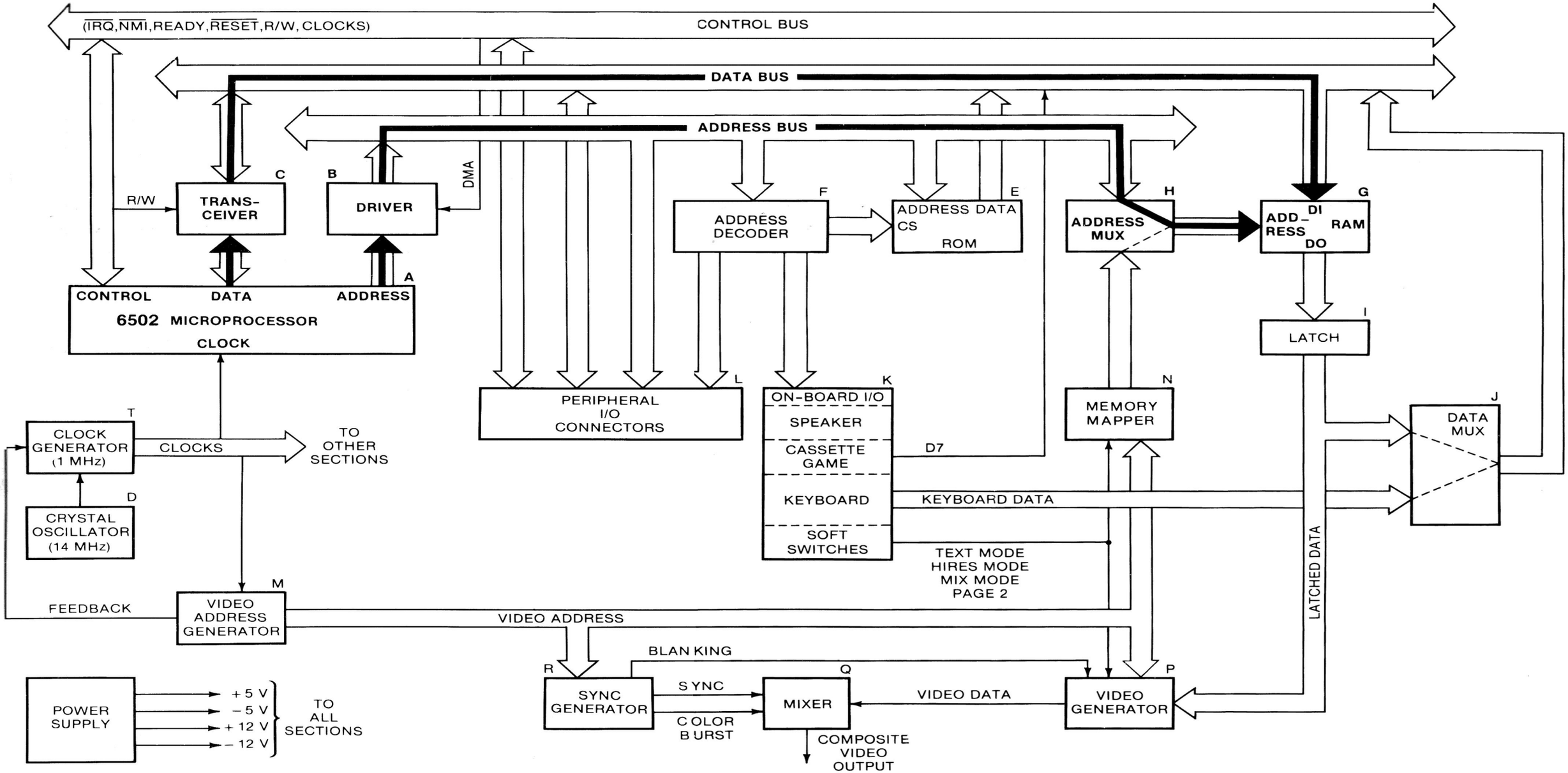


Fig. 2-3. RAM write-cycle path.

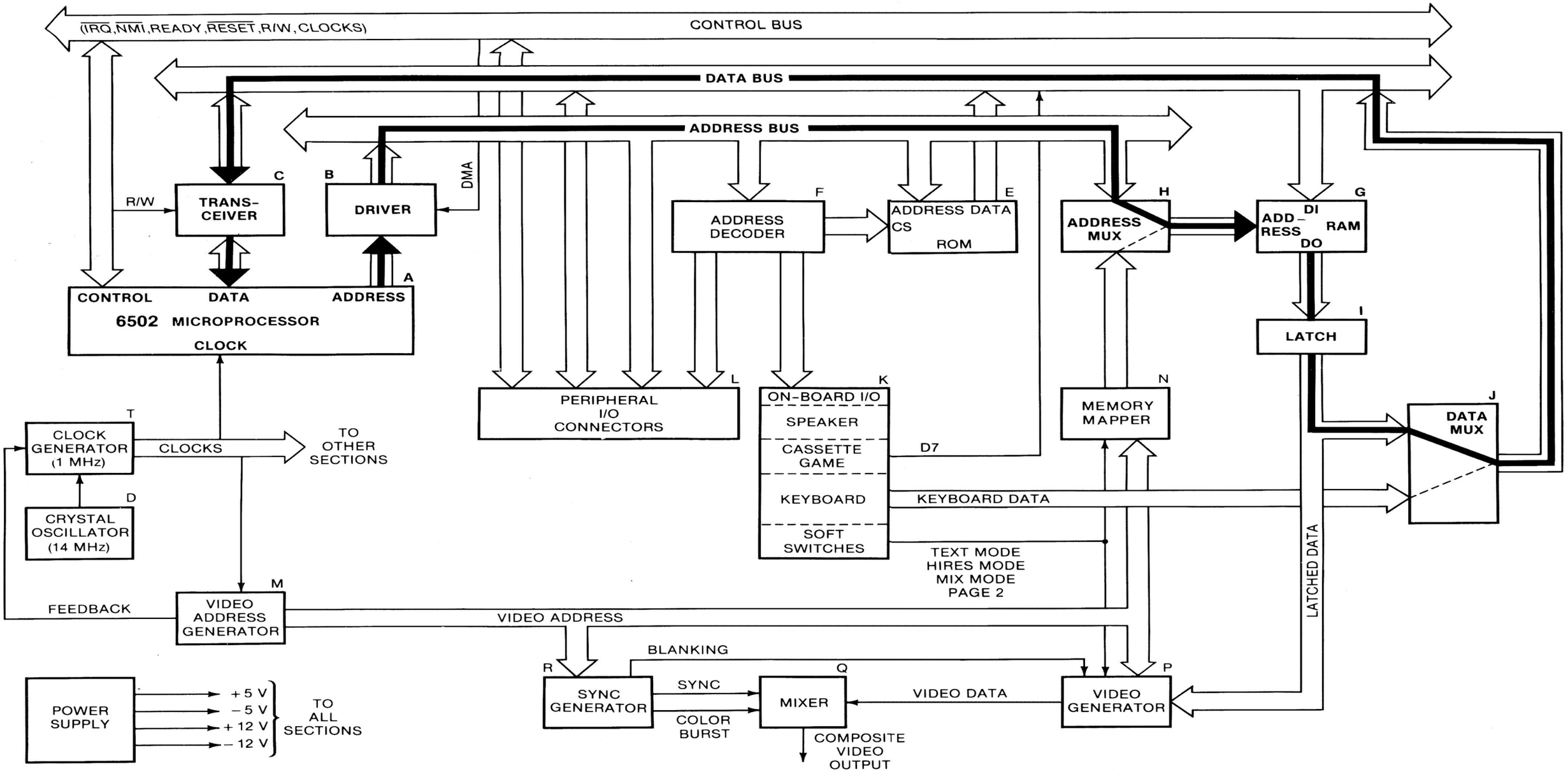


Fig. 2-4. RAM read-cycle path.

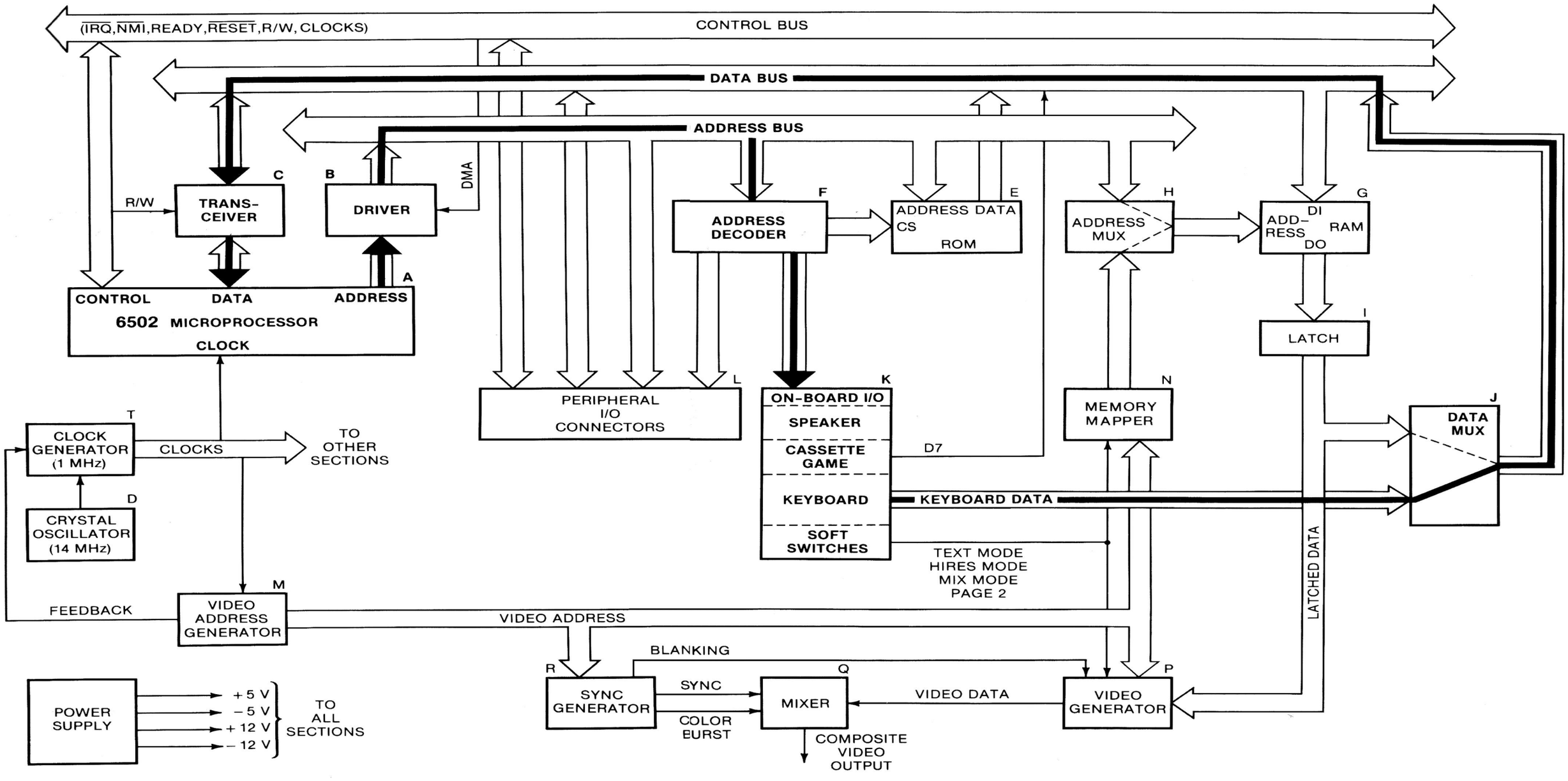


Fig. 2-5. Keyboard read-cycle path.

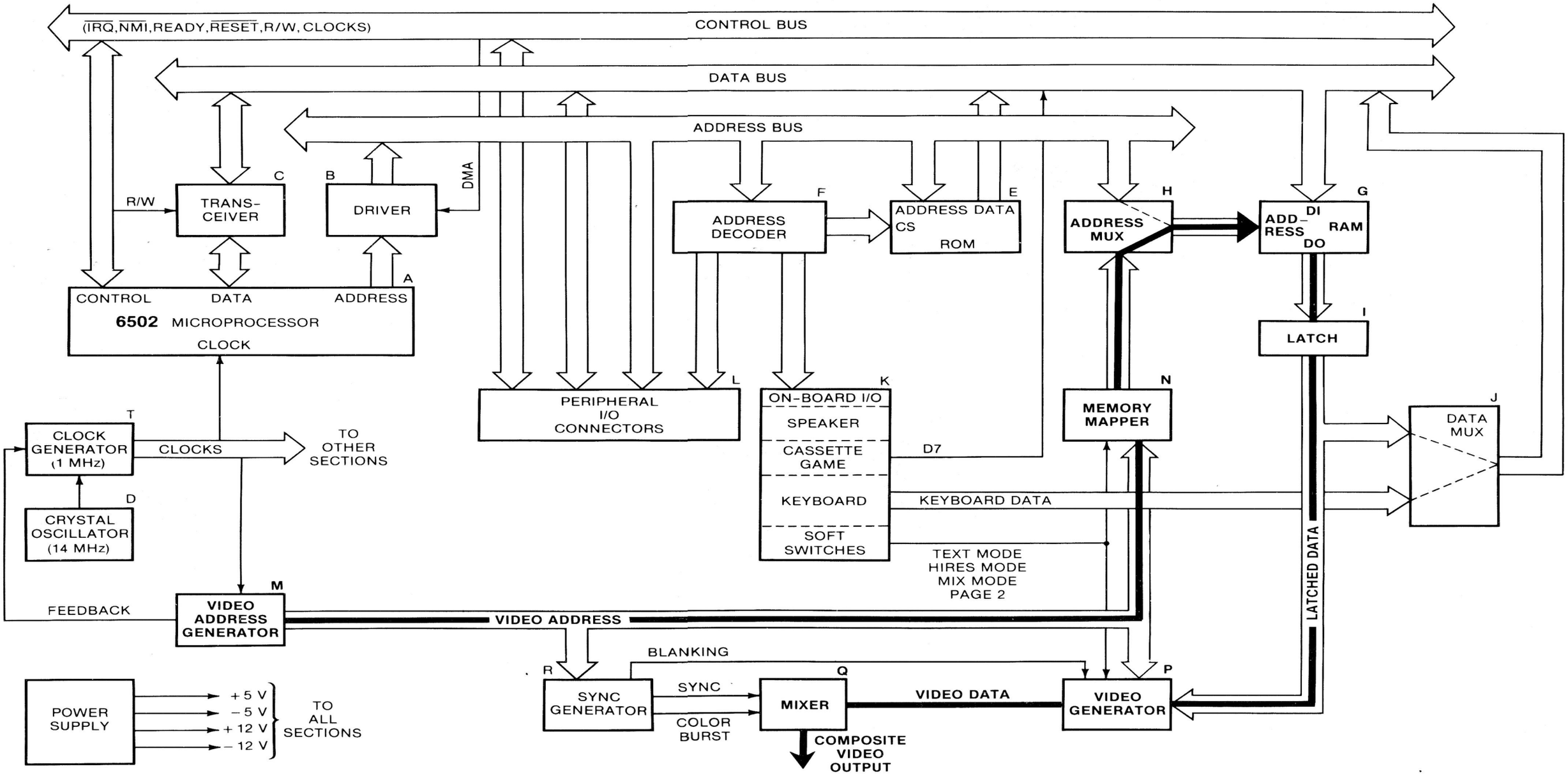


Fig. 2-6. Video-cycle path.

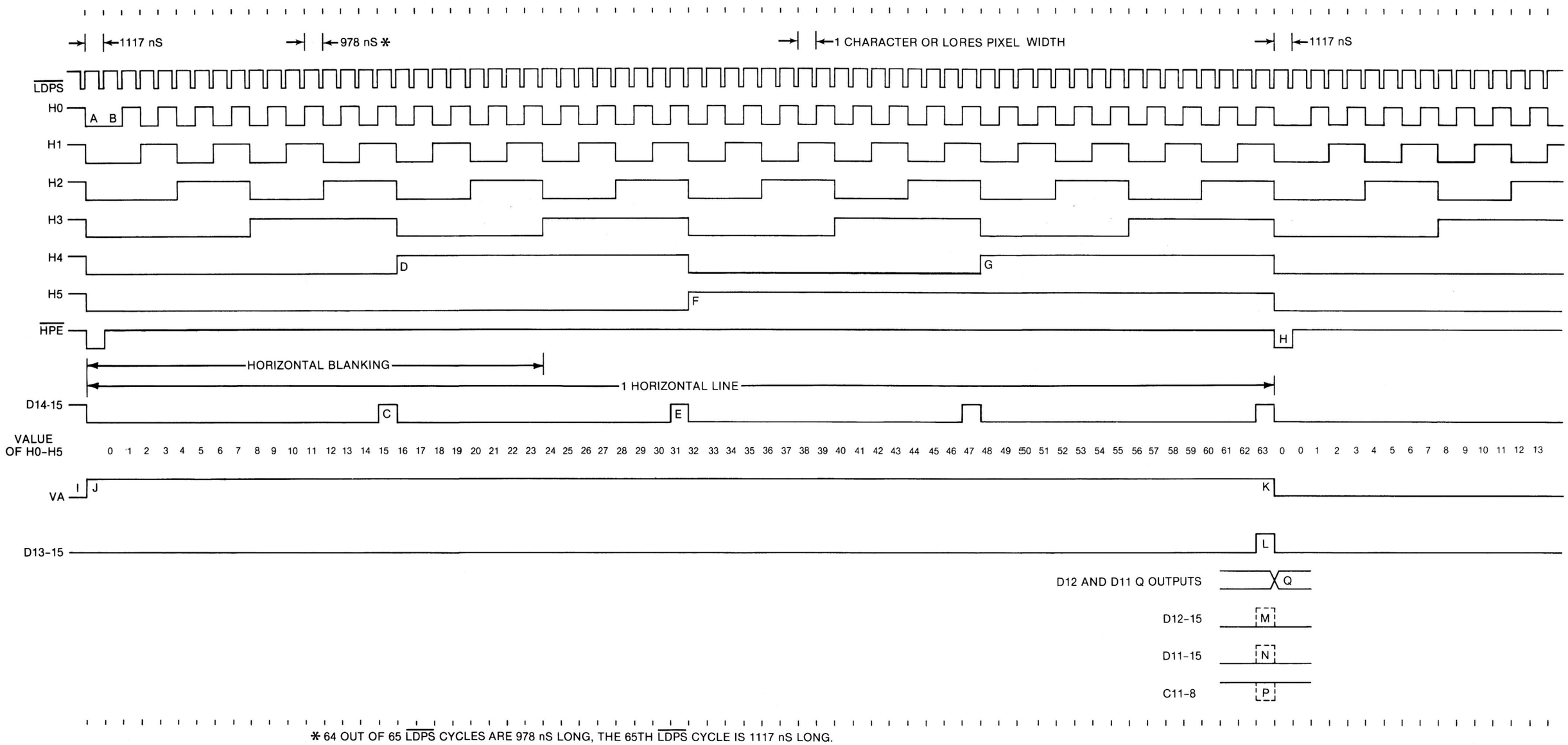


Fig. 3-4. Horizontal timing.

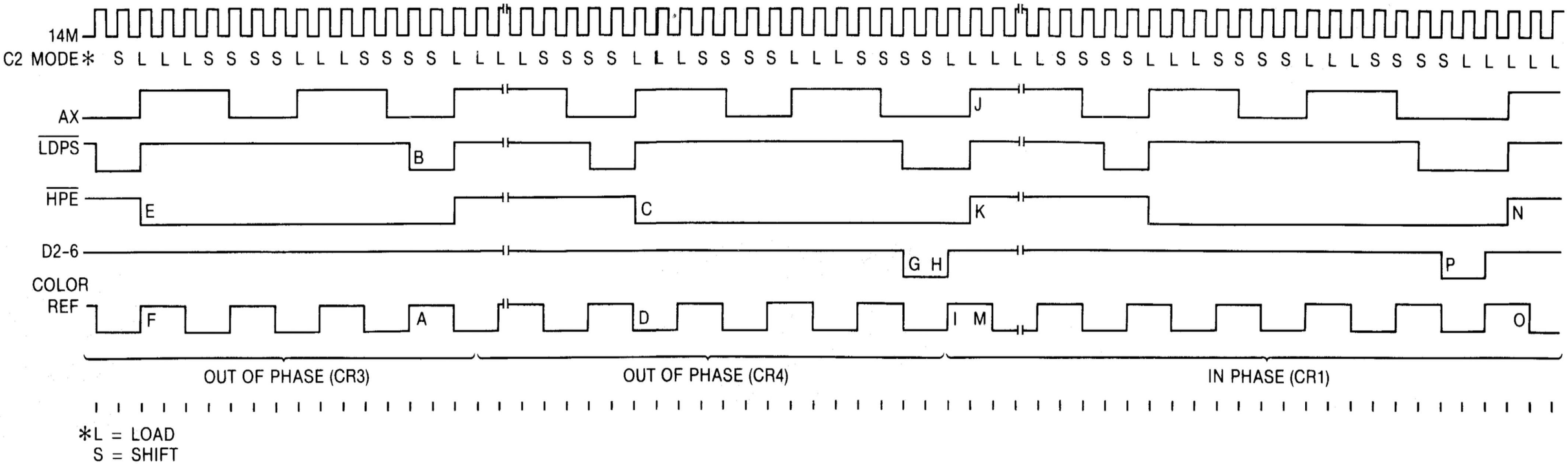


Fig. 3-7. Clock synchronization—CR3 and CR4.

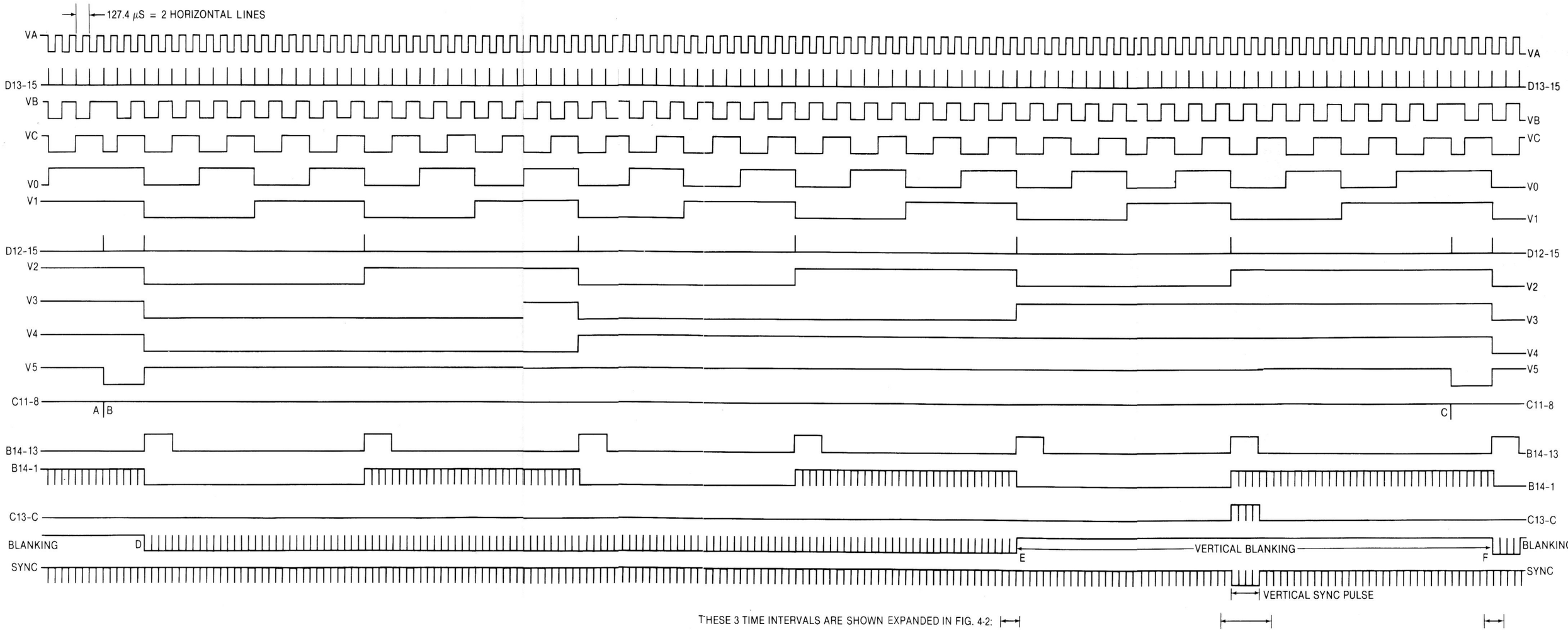


Fig. 4-1. Vertical timing

Fig. 4-1. Vertical timing

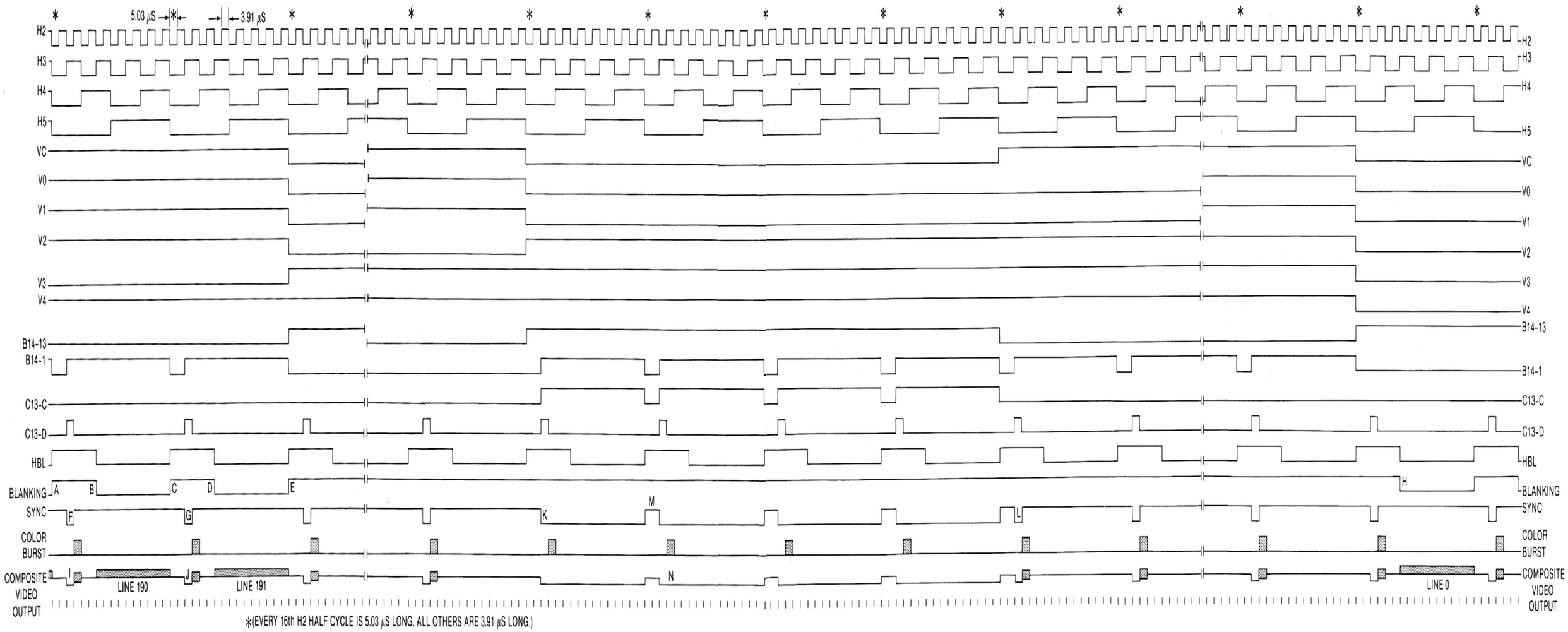


Fig. 4-2. Vertical retrace interval.

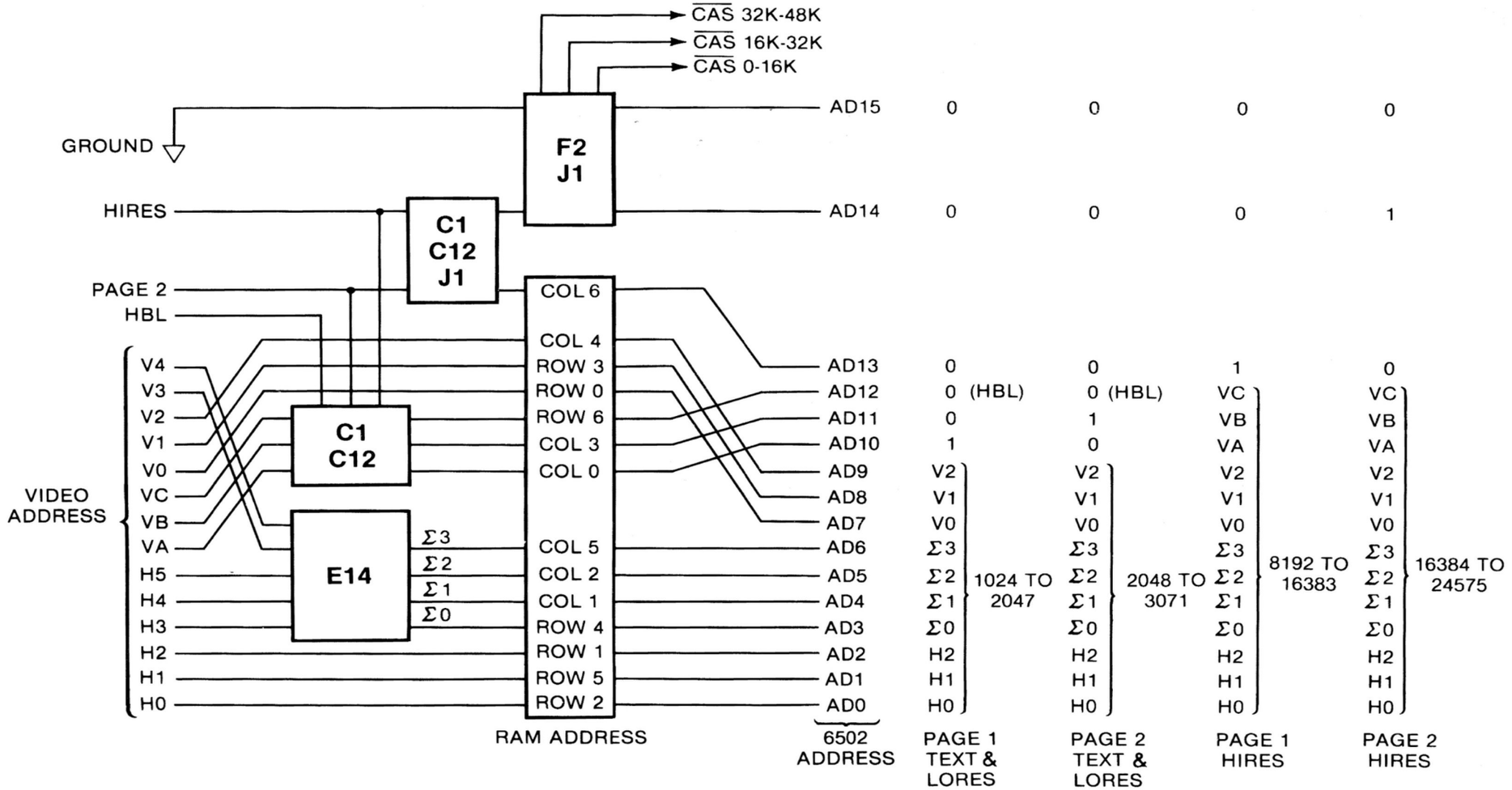
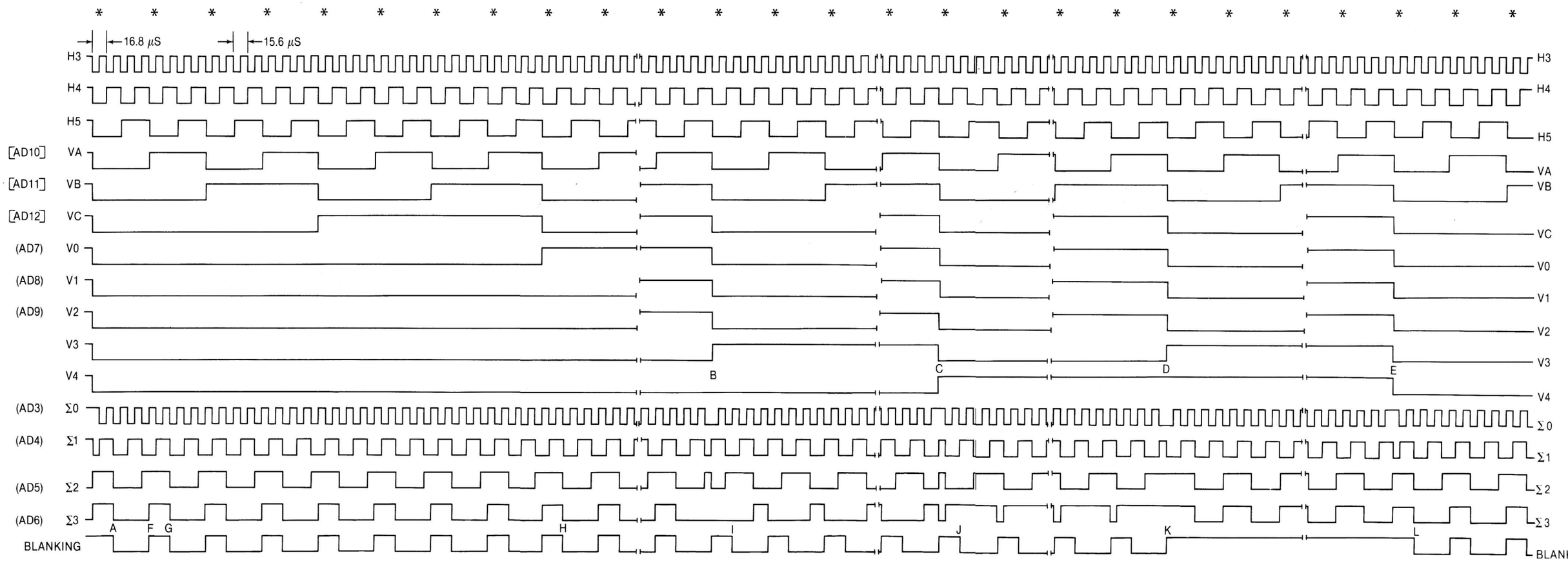


Fig. 5-7. Video mapping.



*EVERY FOURTH H3...PERIOD IS 16.8 μS LONG. ALL OTHERS ARE 15.6 μS LONG.

Fig. 5-8. Video address signals.

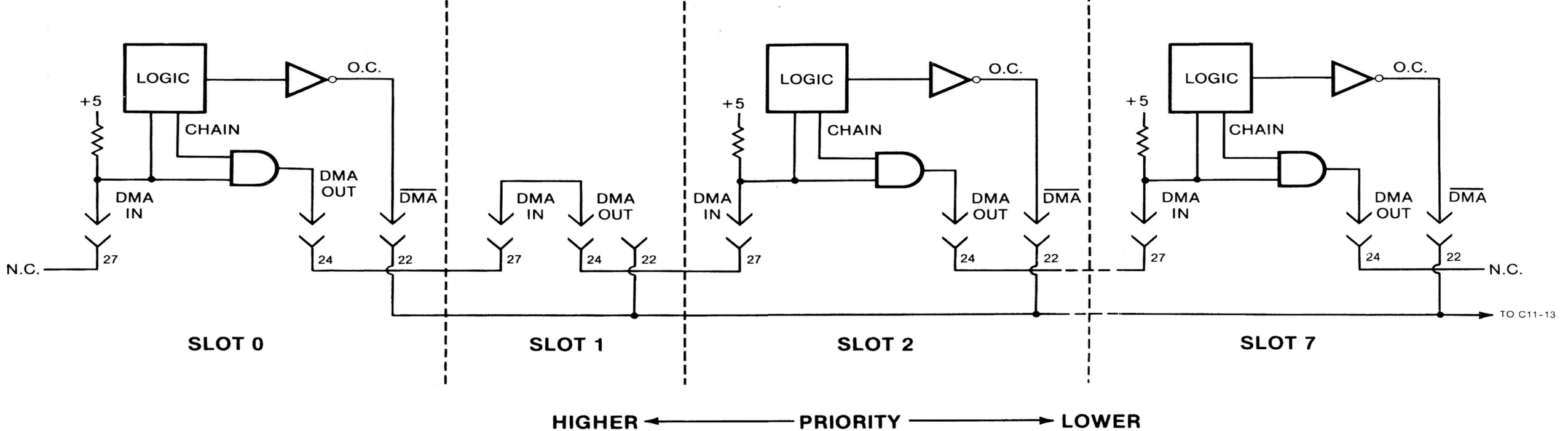


Fig. 6-11. DMA daisy chain.

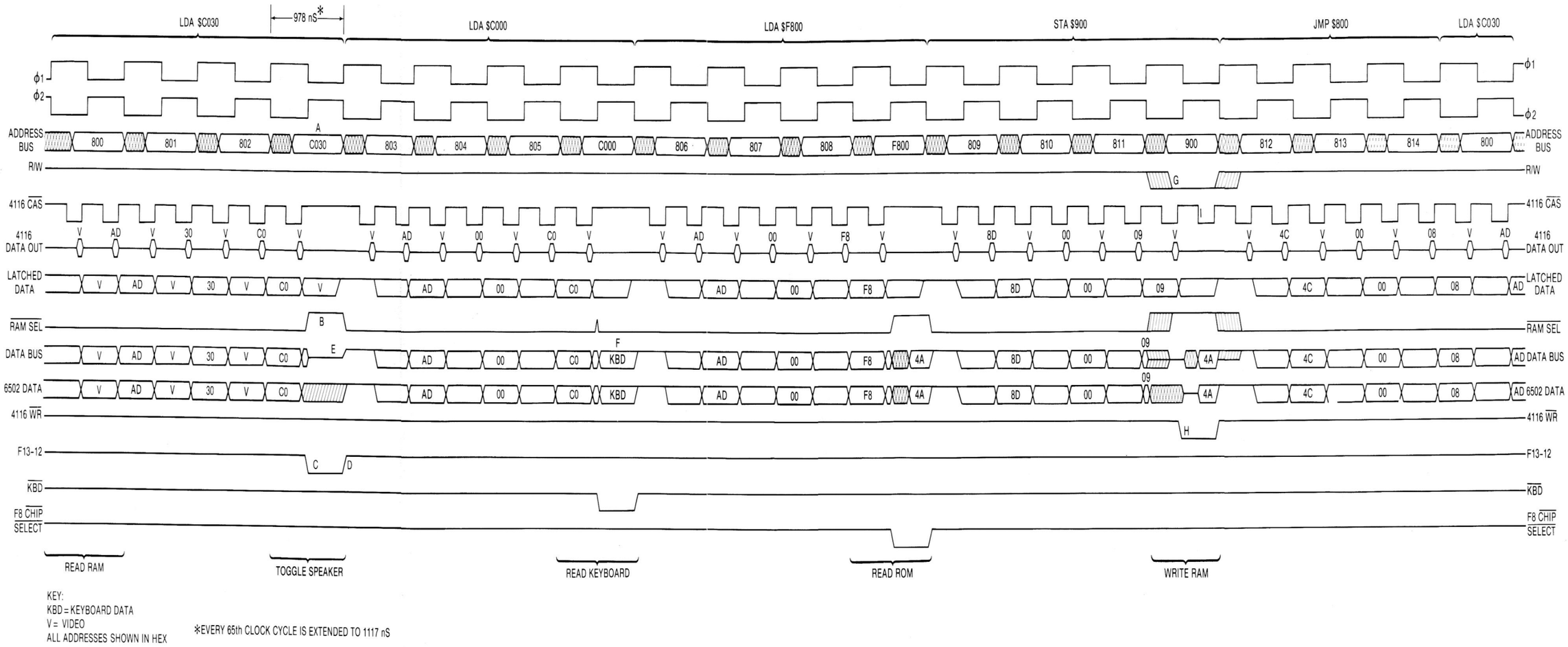


Fig. 6-16. 6502 scope loop.

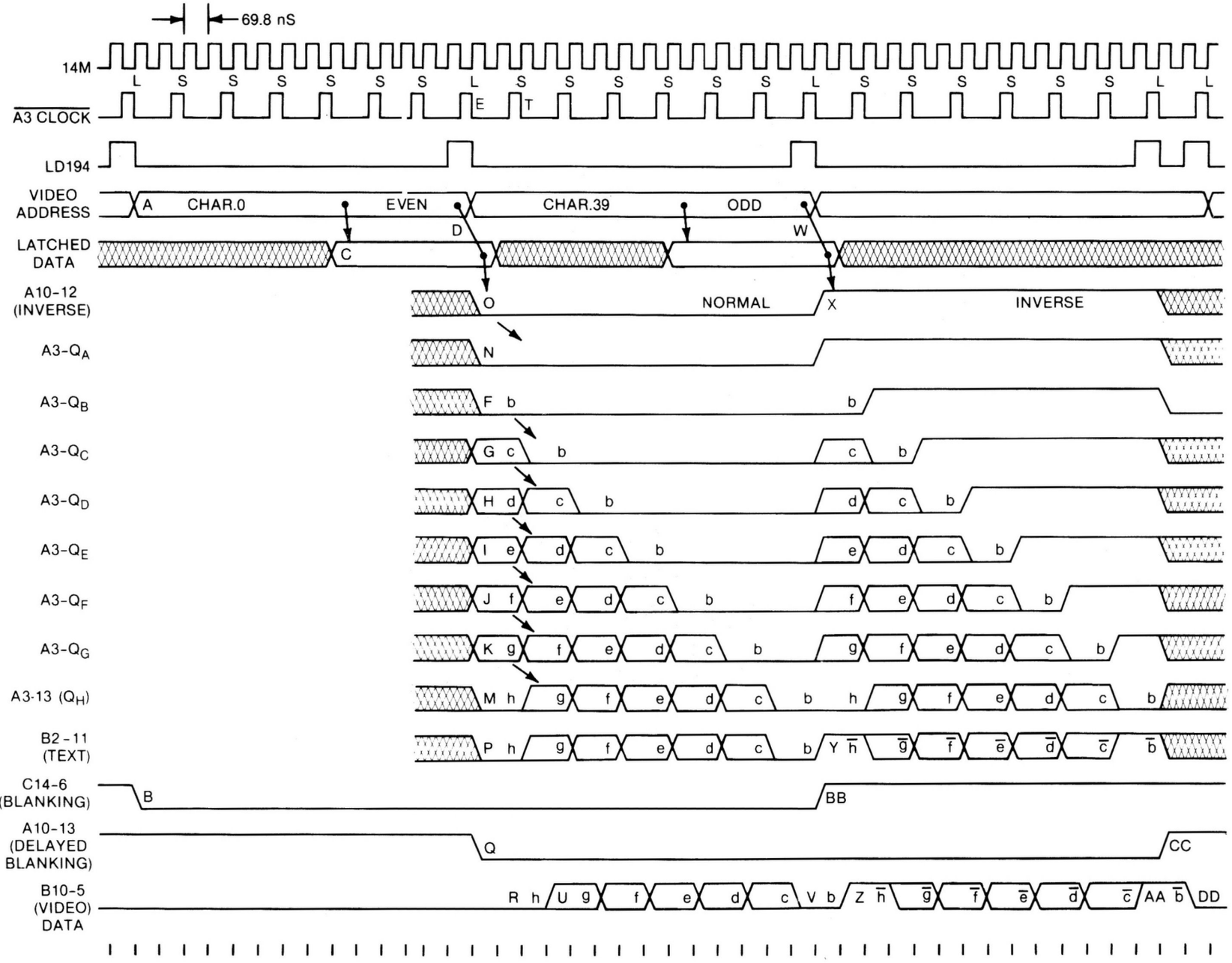


Fig. 8-26. Text mode timing diagram

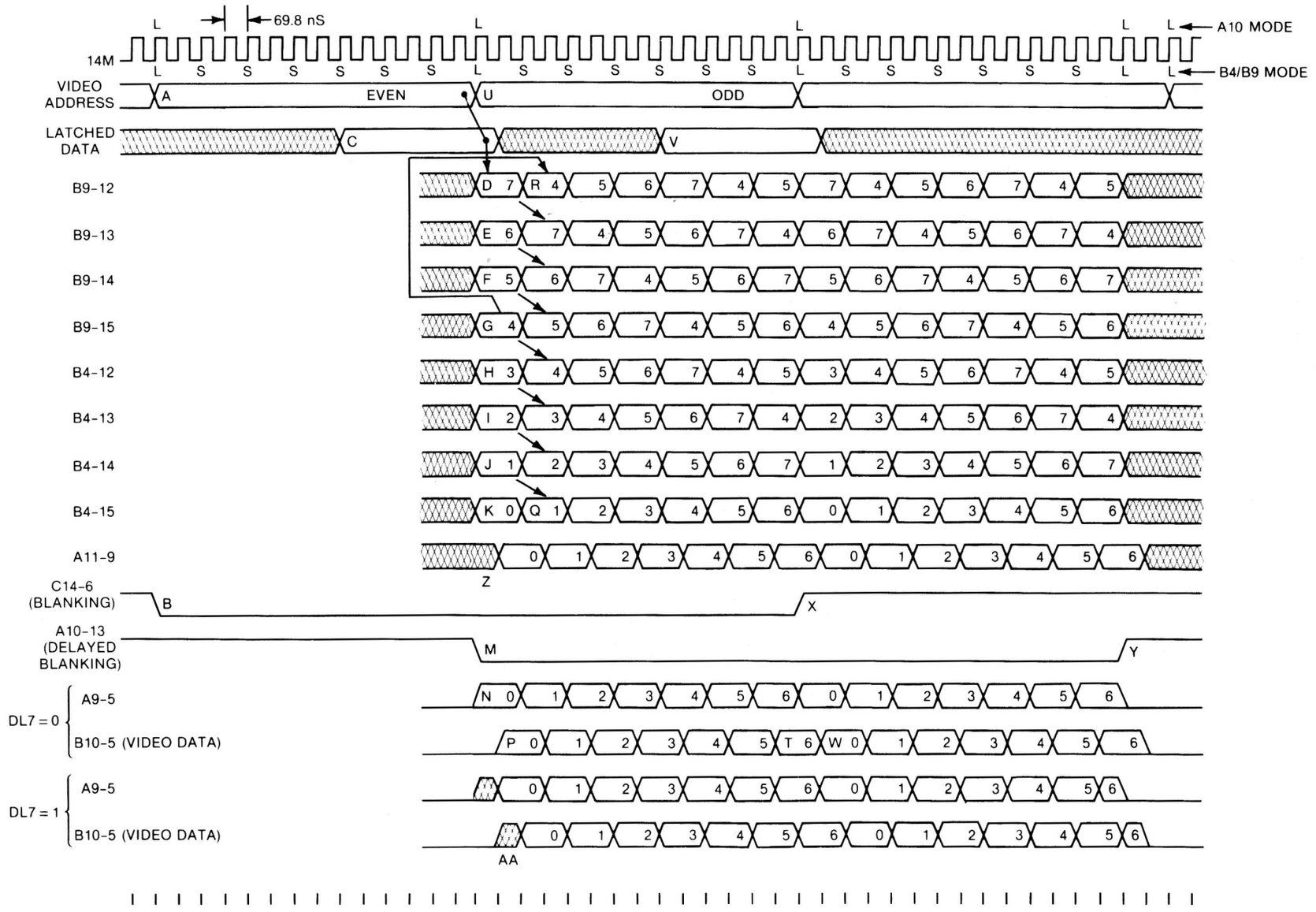


Fig. 8-28. HIRES-mode timing diagram.

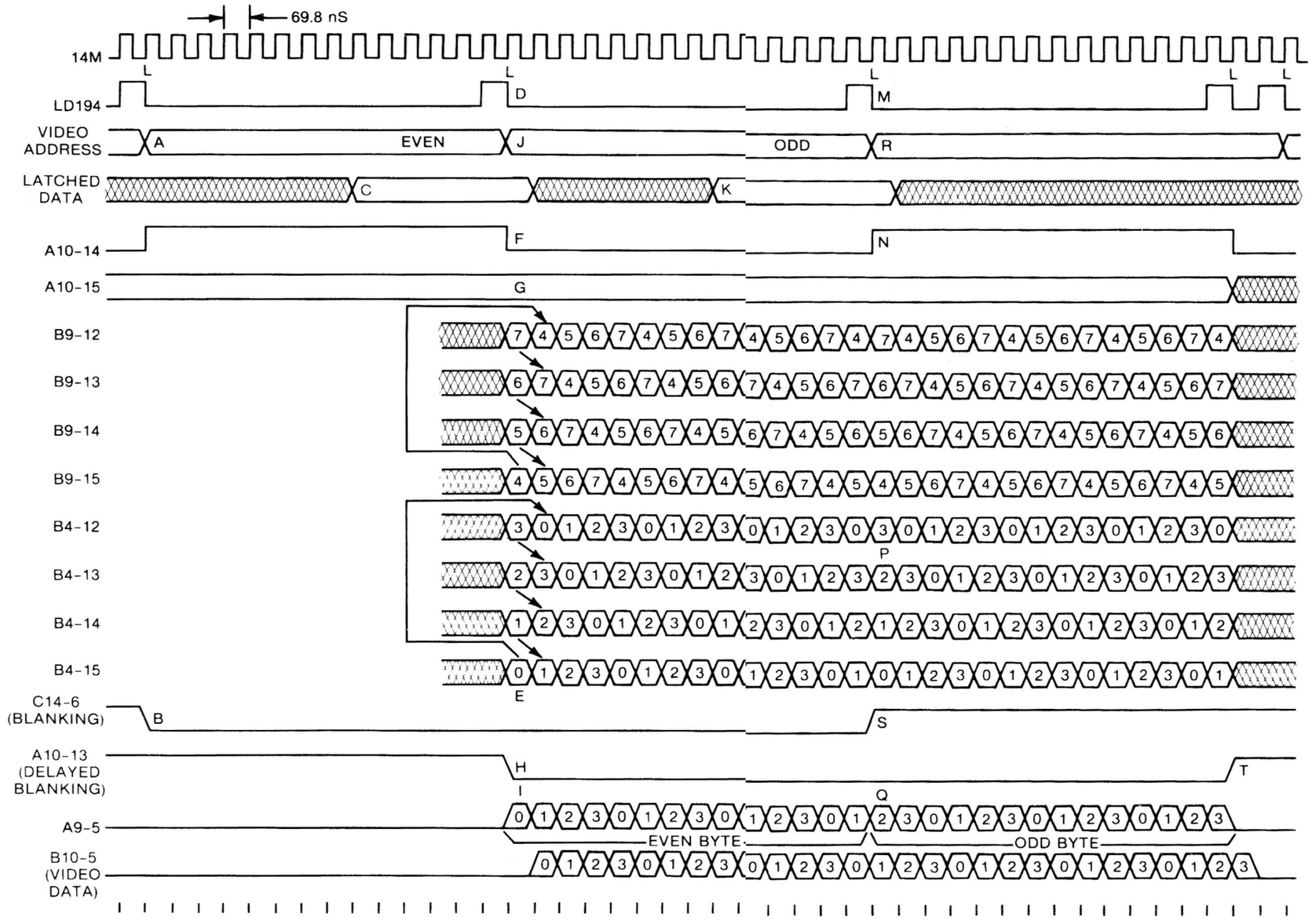


Fig. 8-31. LORES-mode timing diagram.

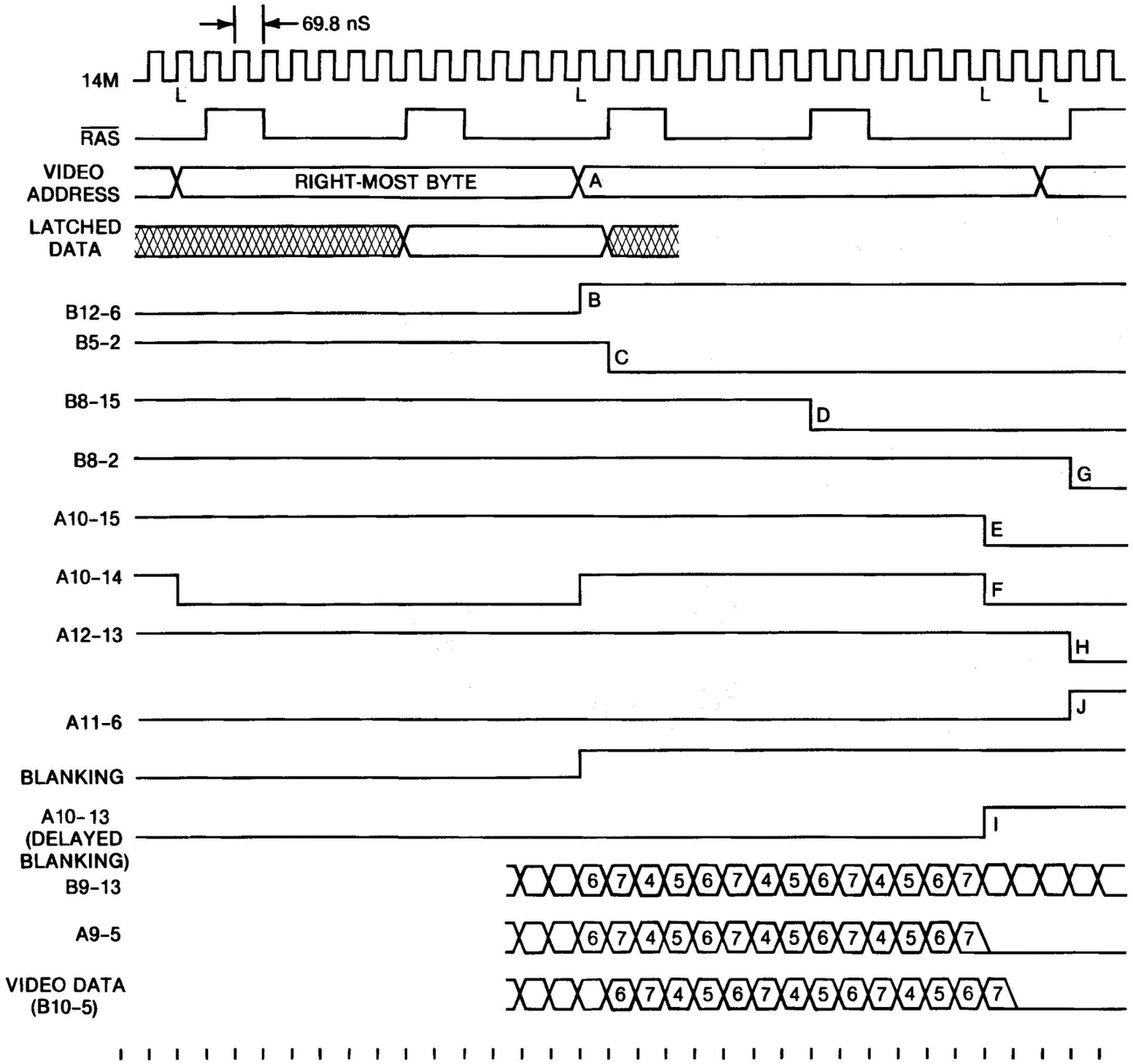


Fig. 8-34. Mixed LORES and text.

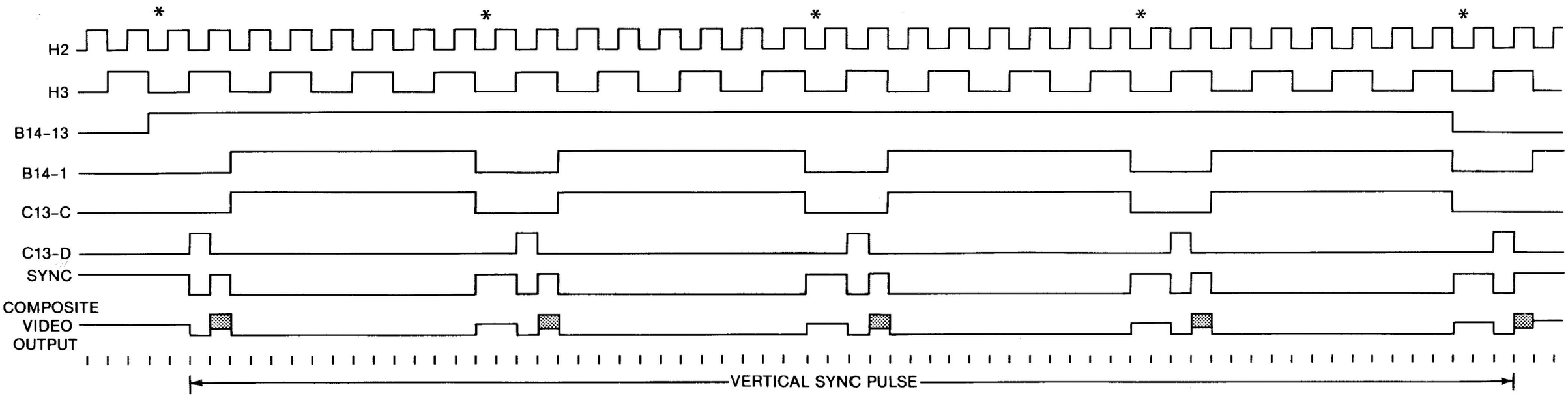


Fig. B-2. Vertical sync—Rev. 7.

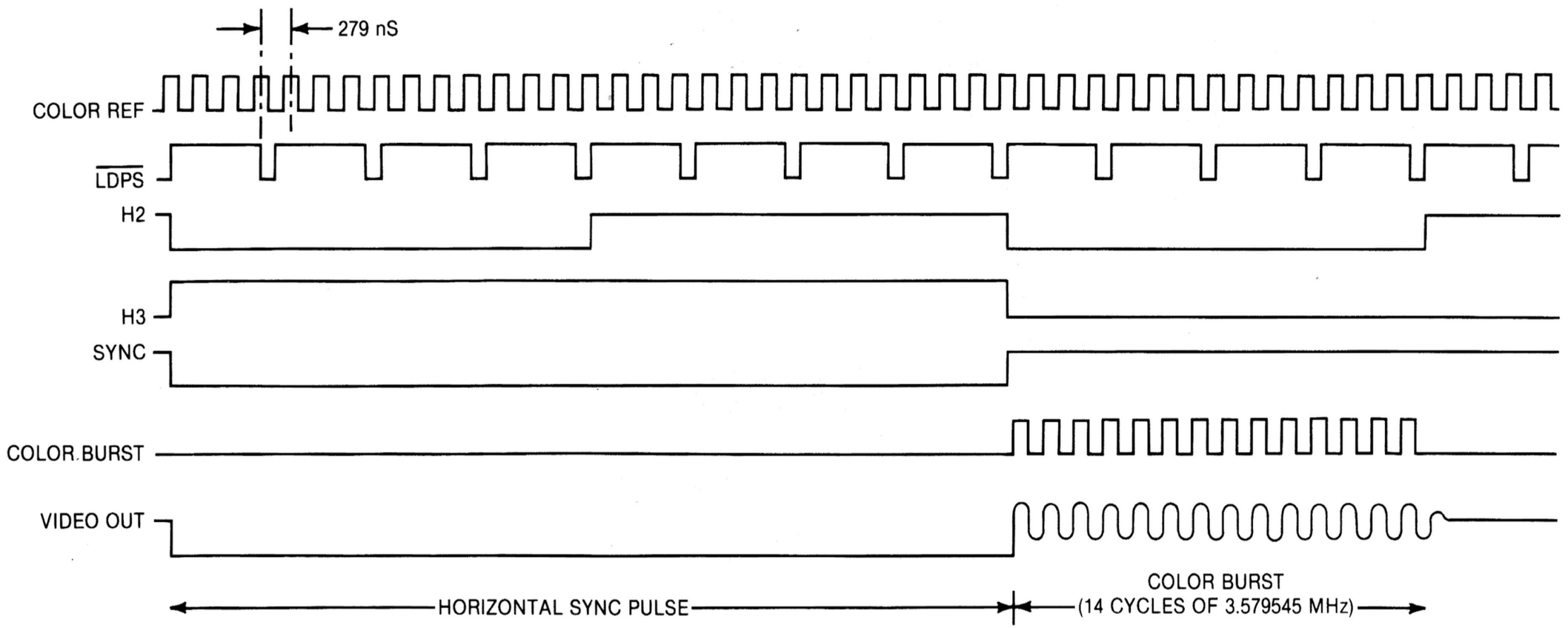


Fig. B-4. Color burst—Rev. 0 and 1.

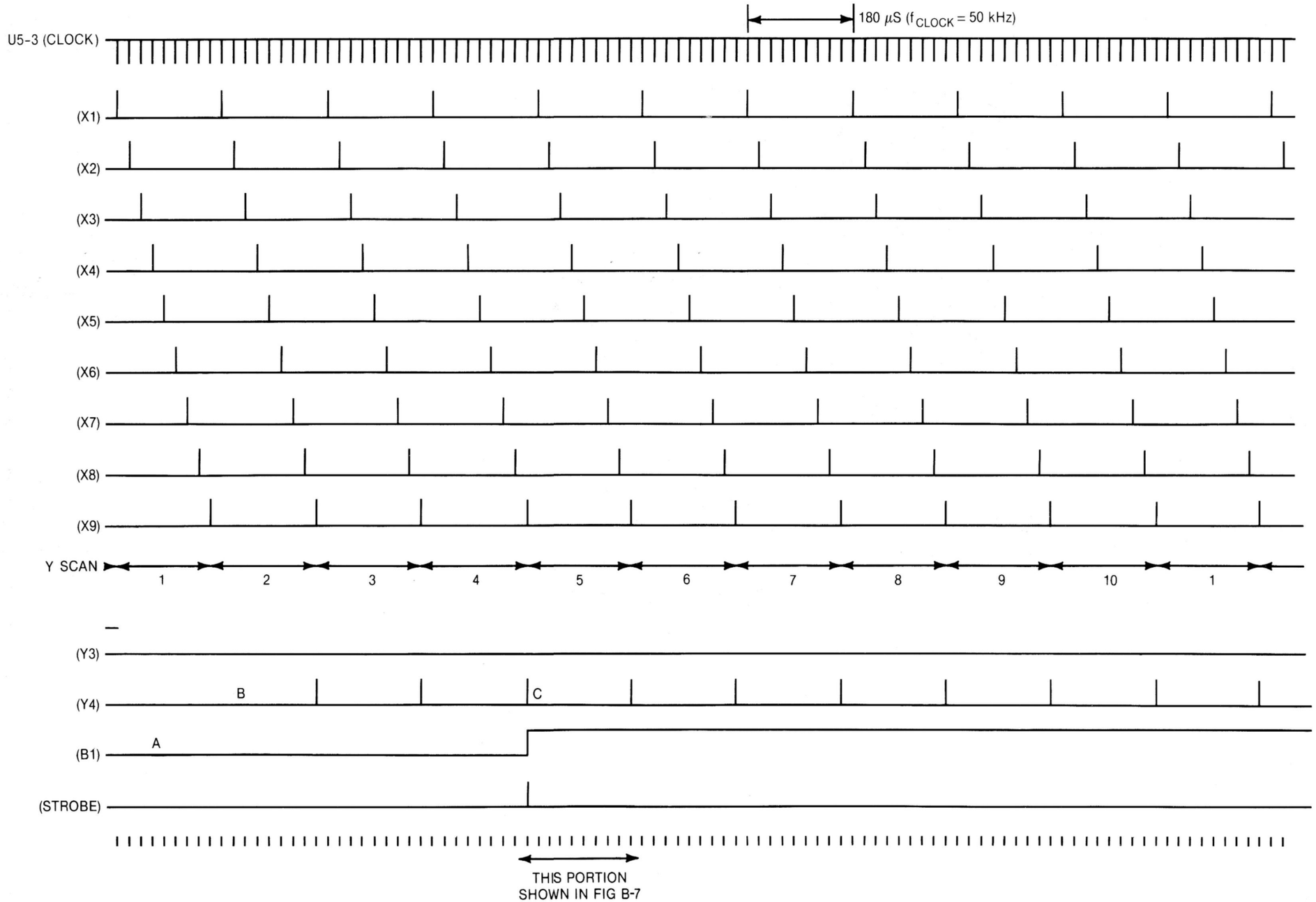


Fig. B-8. Y scan—single-piece keyboard.

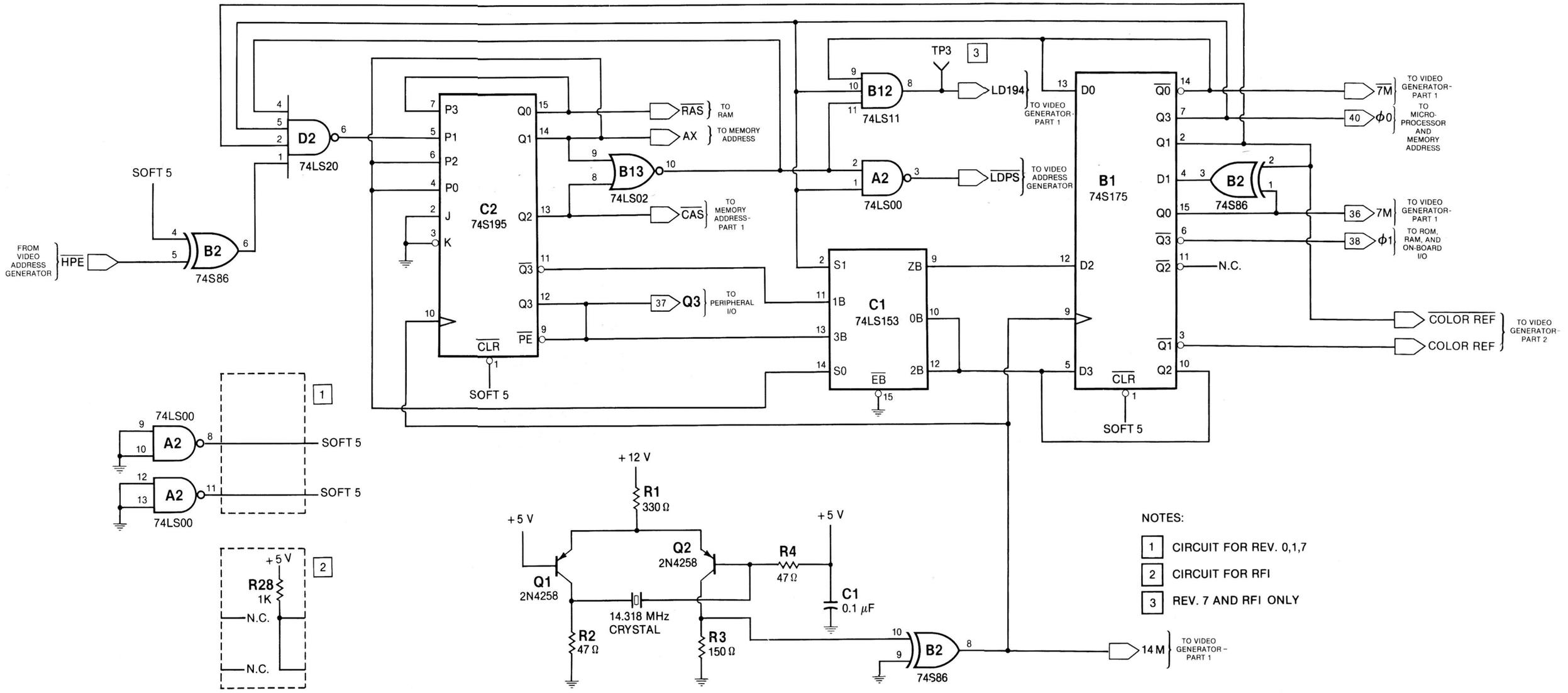
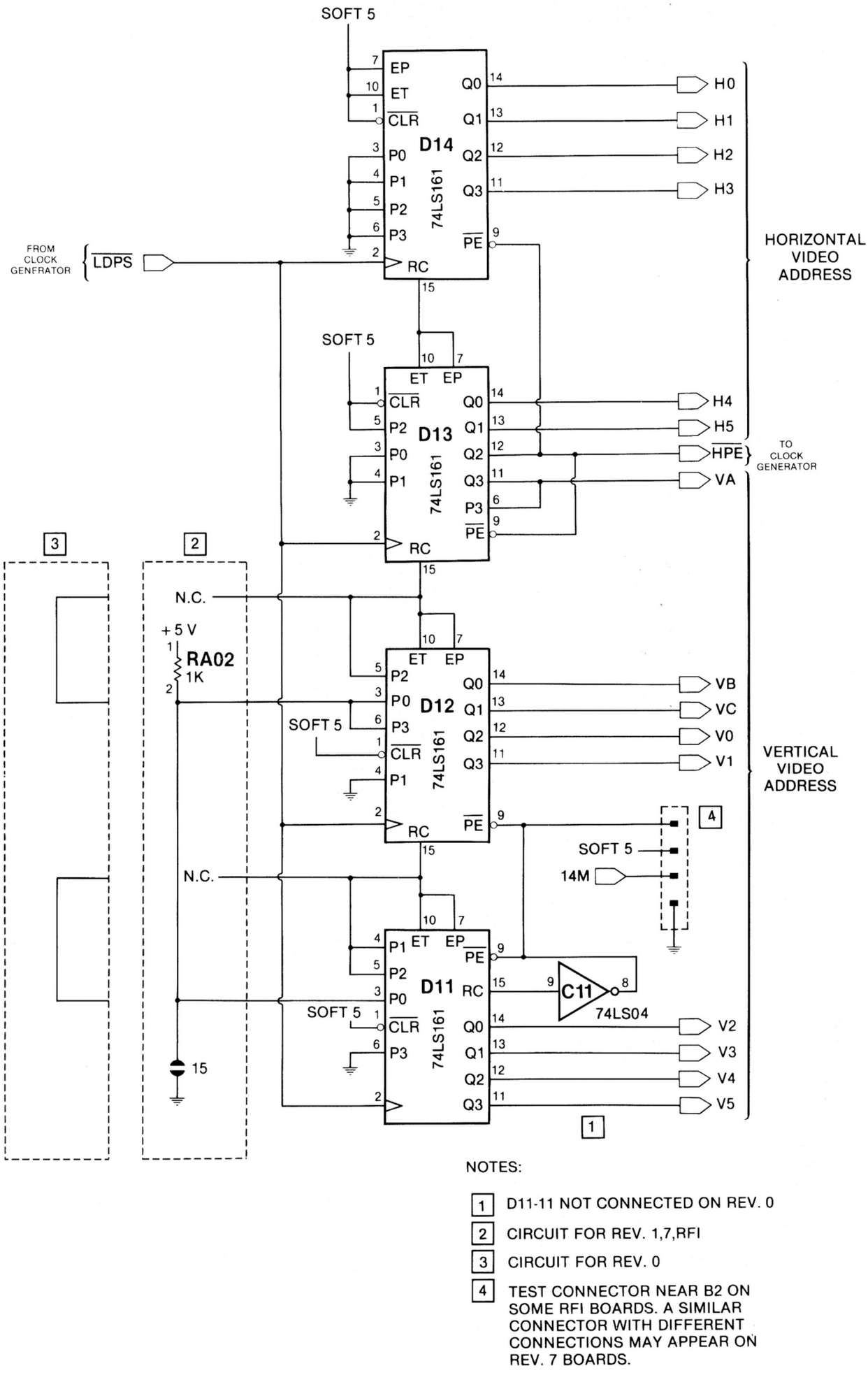


Fig. C-2. Clock generator (all revisions).



NOTES:

1. D11-11 NOT CONNECTED ON REV. 0
2. CIRCUIT FOR REV. 1,7,RFI
3. CIRCUIT FOR REV. 0
4. TEST CONNECTOR NEAR B2 ON SOME RFI BOARDS. A SIMILAR CONNECTOR WITH DIFFERENT CONNECTIONS MAY APPEAR ON REV. 7 BOARDS.

Fig. C-3. Video address generator (all revisions).

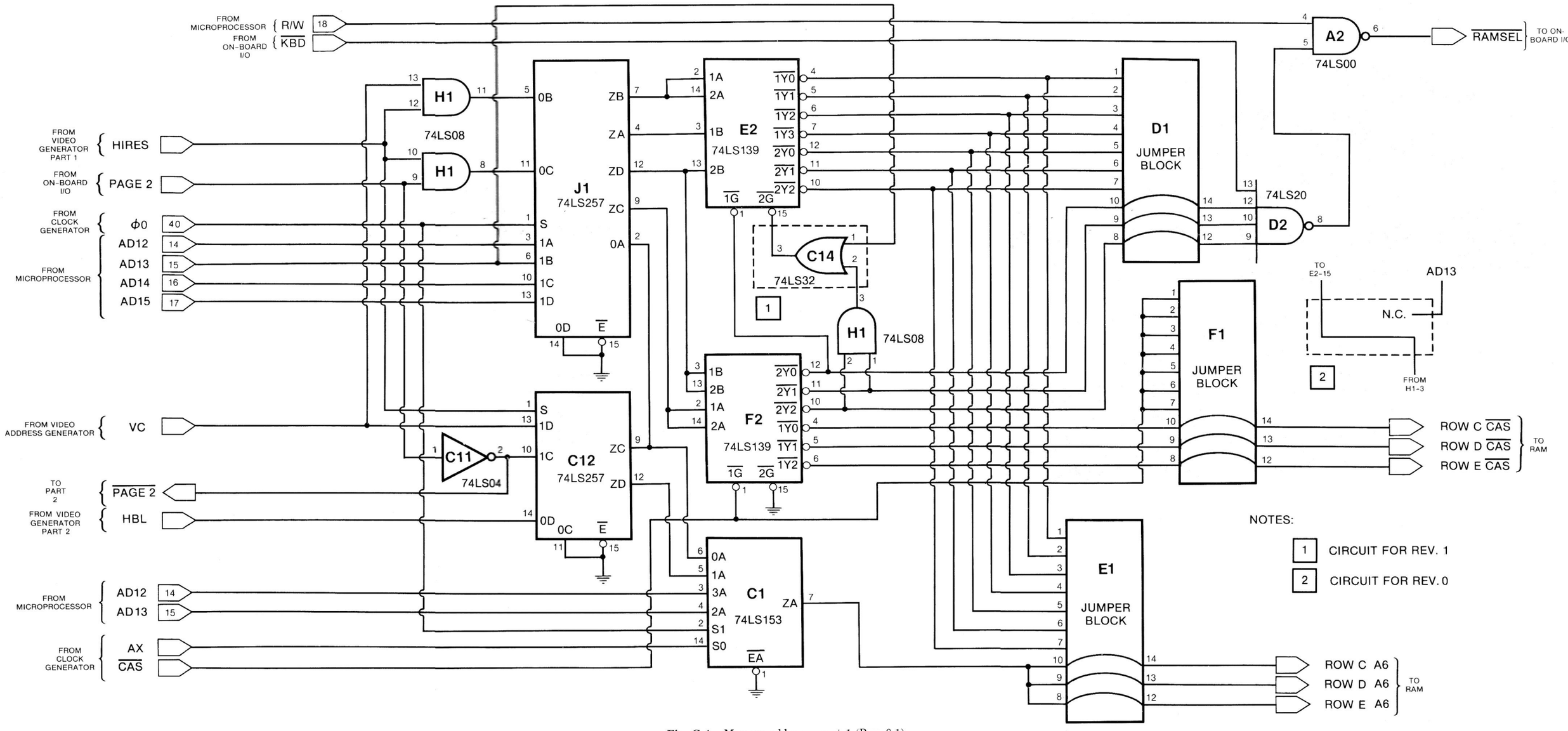


Fig. C-4. Memory address—part 1 (Rev. 0,1).

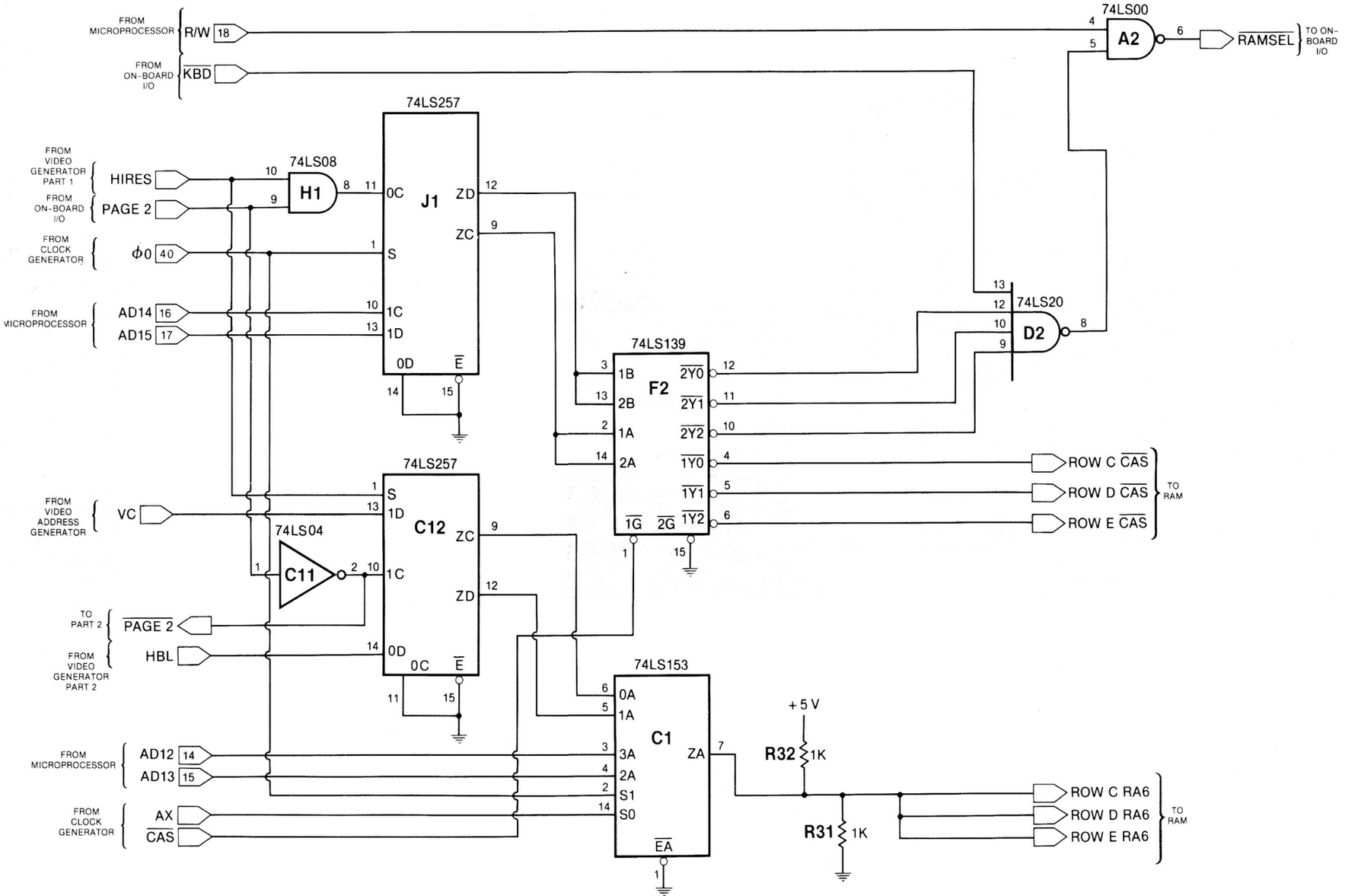


Fig. C--5. Memory address—part 1 (Rev. 7, RFI).

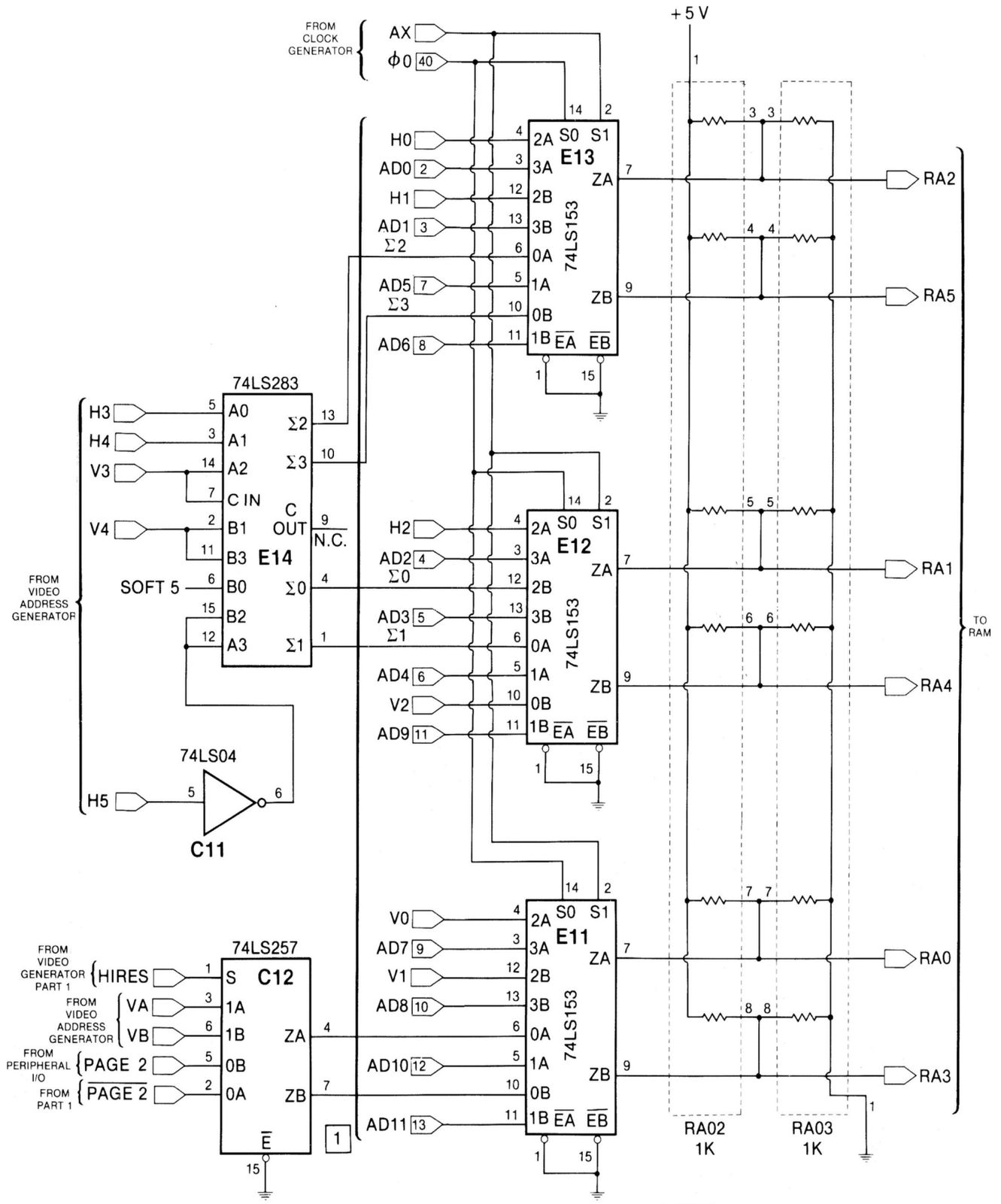


Fig. C-6. Memory address—part 2 (all revisions).

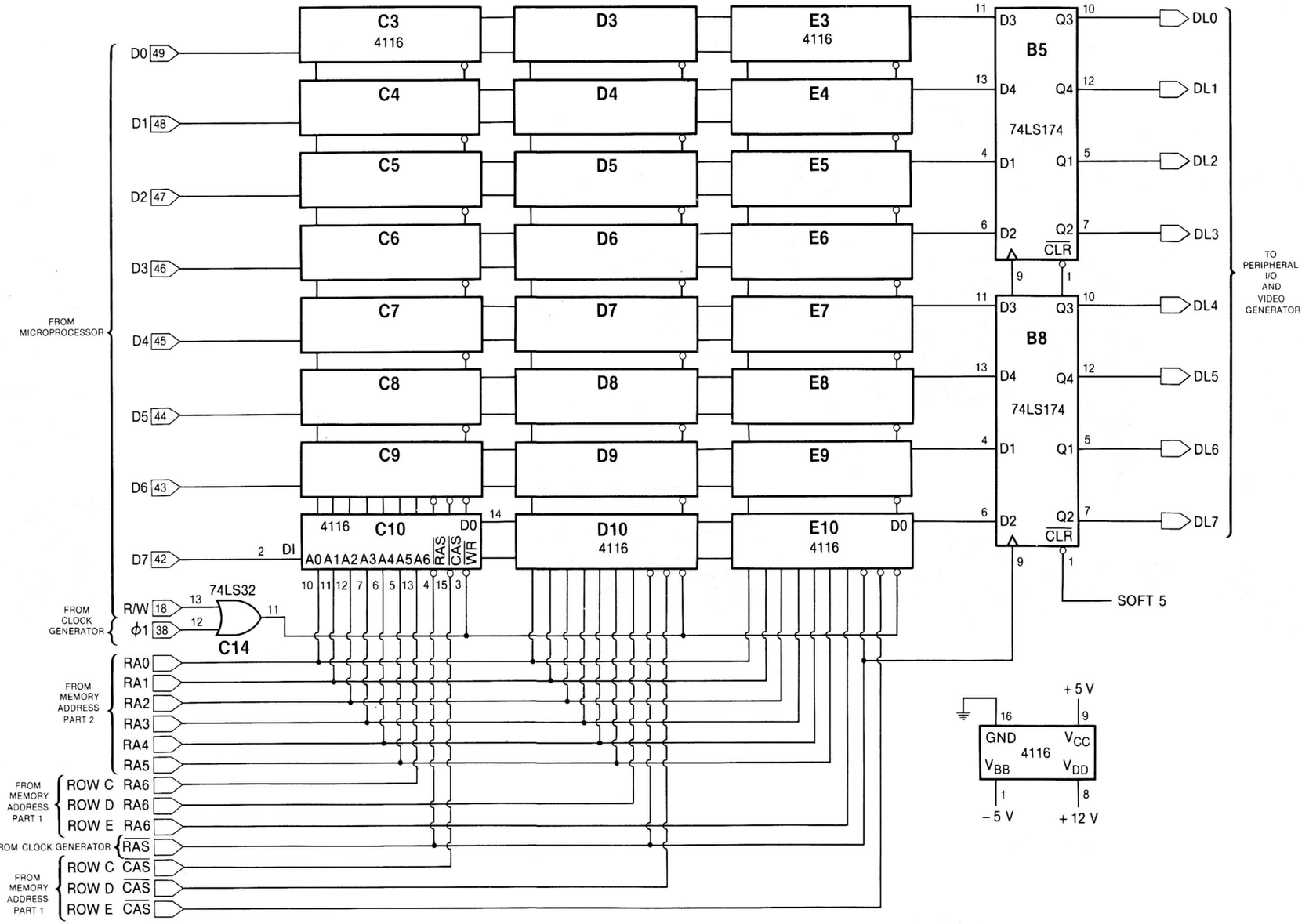


Fig. C-7. RAM (all revisions).

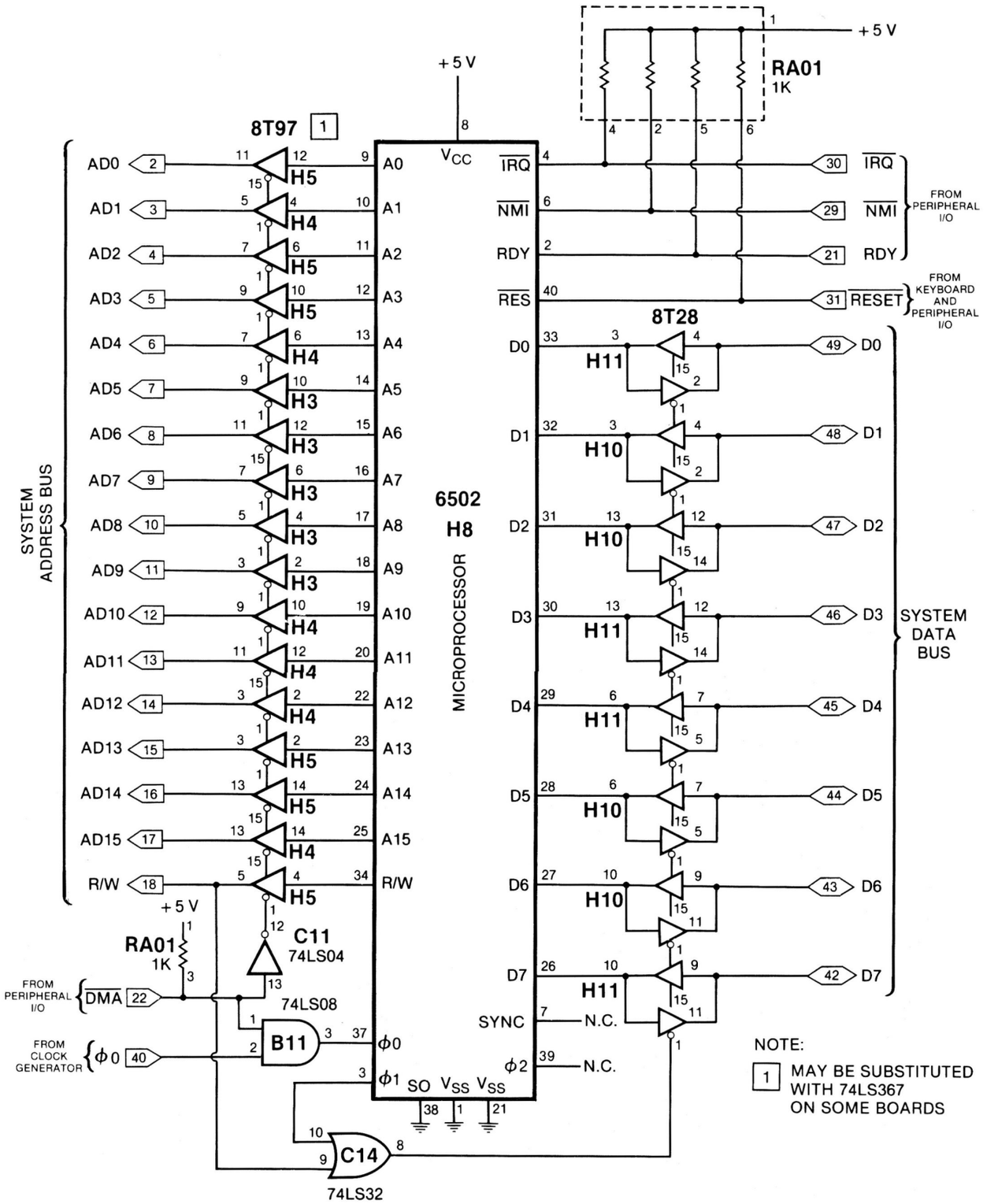


Fig. C-8. Microprocessor (Rev. 0.1.7).

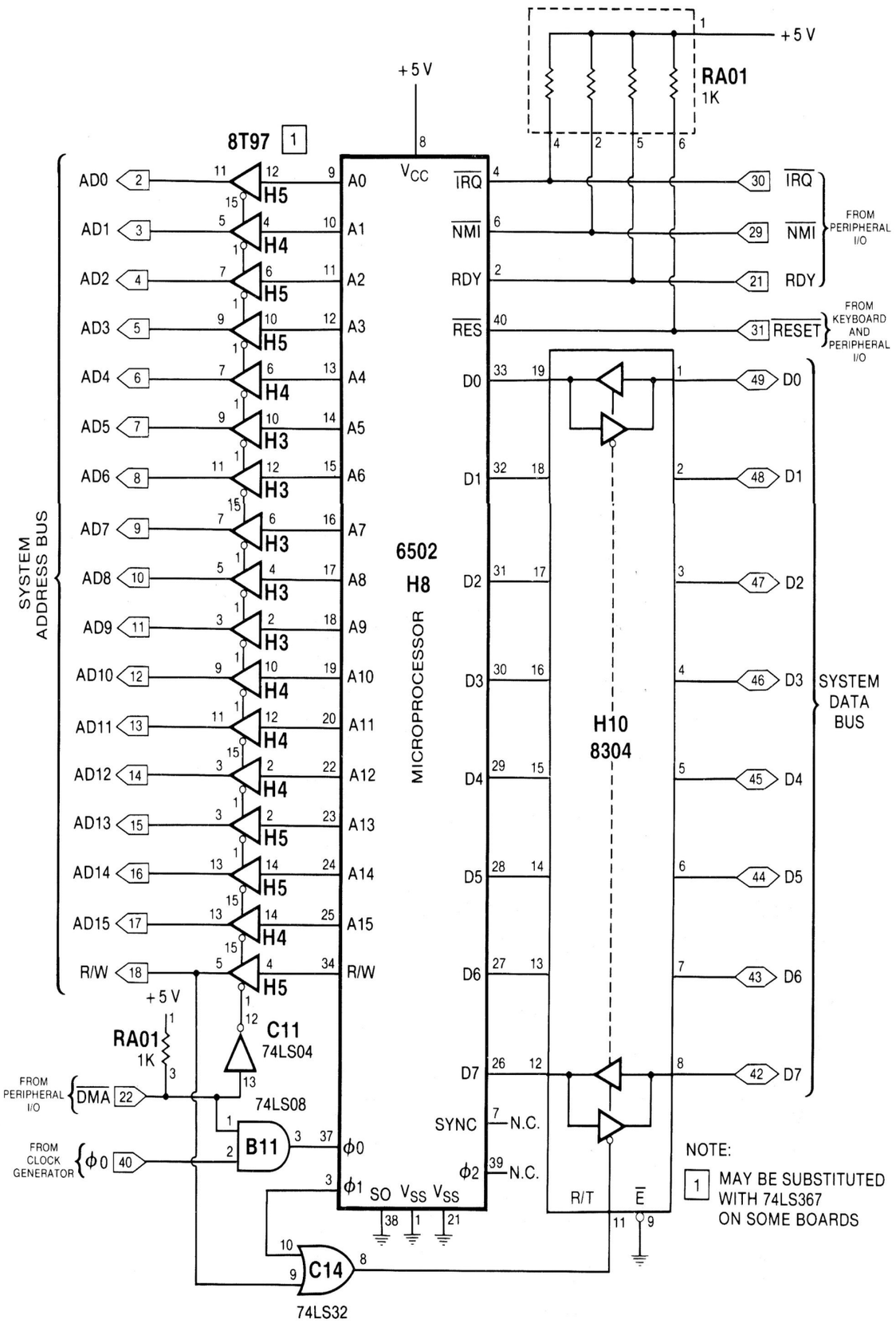


Fig. C-9. Microprocessor (RFI).

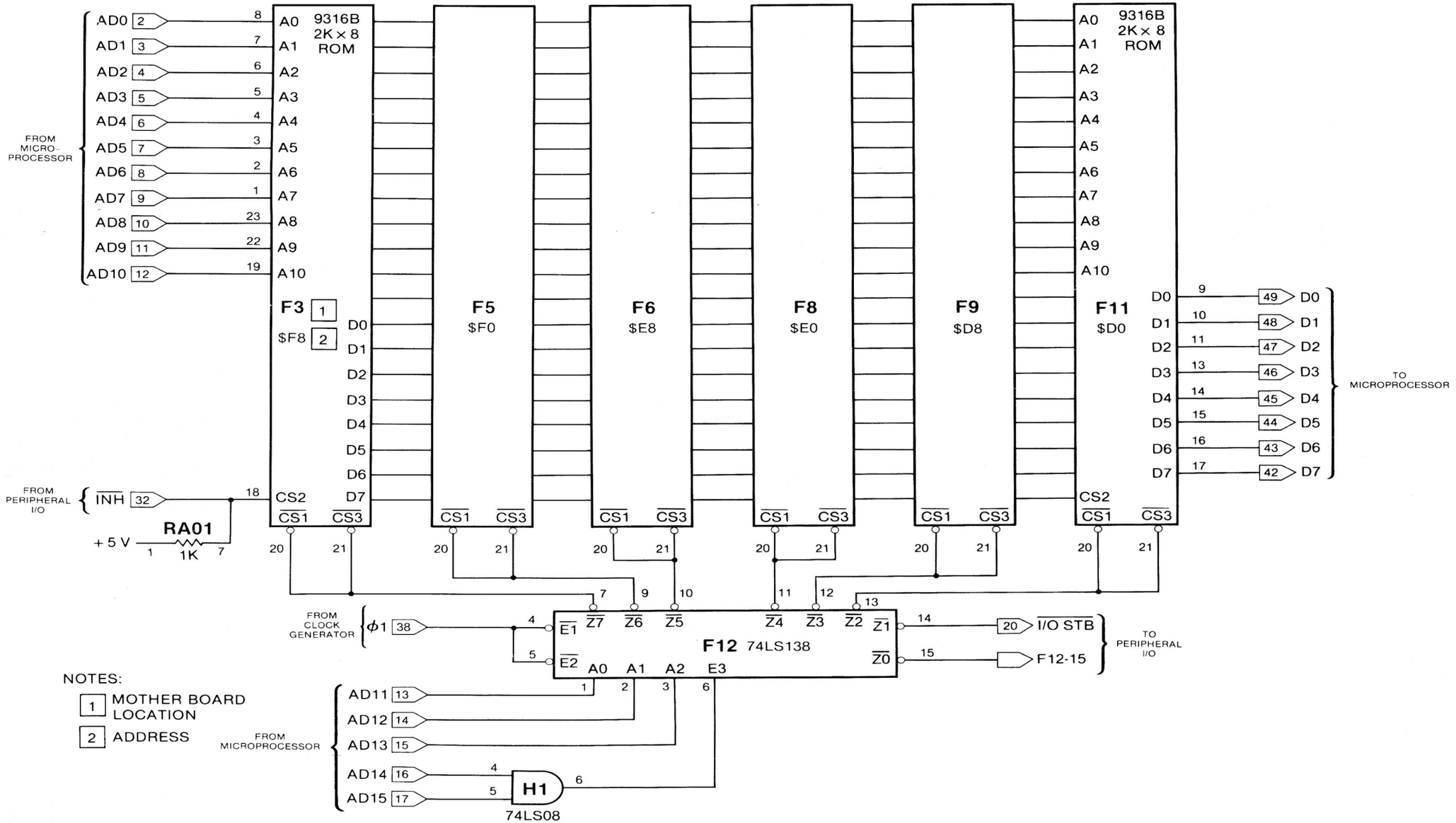


Fig. C-10. ROM (all revisions).

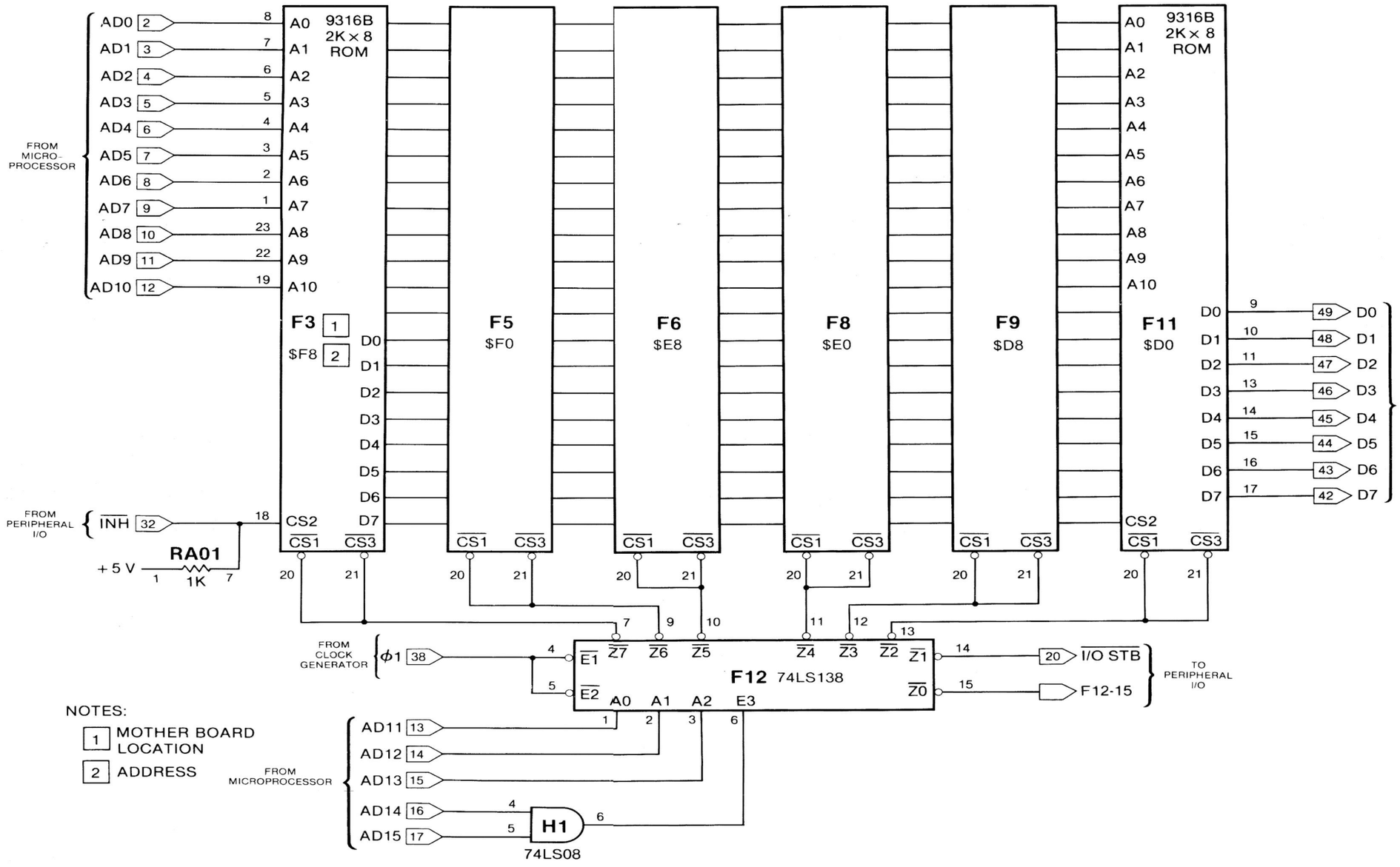
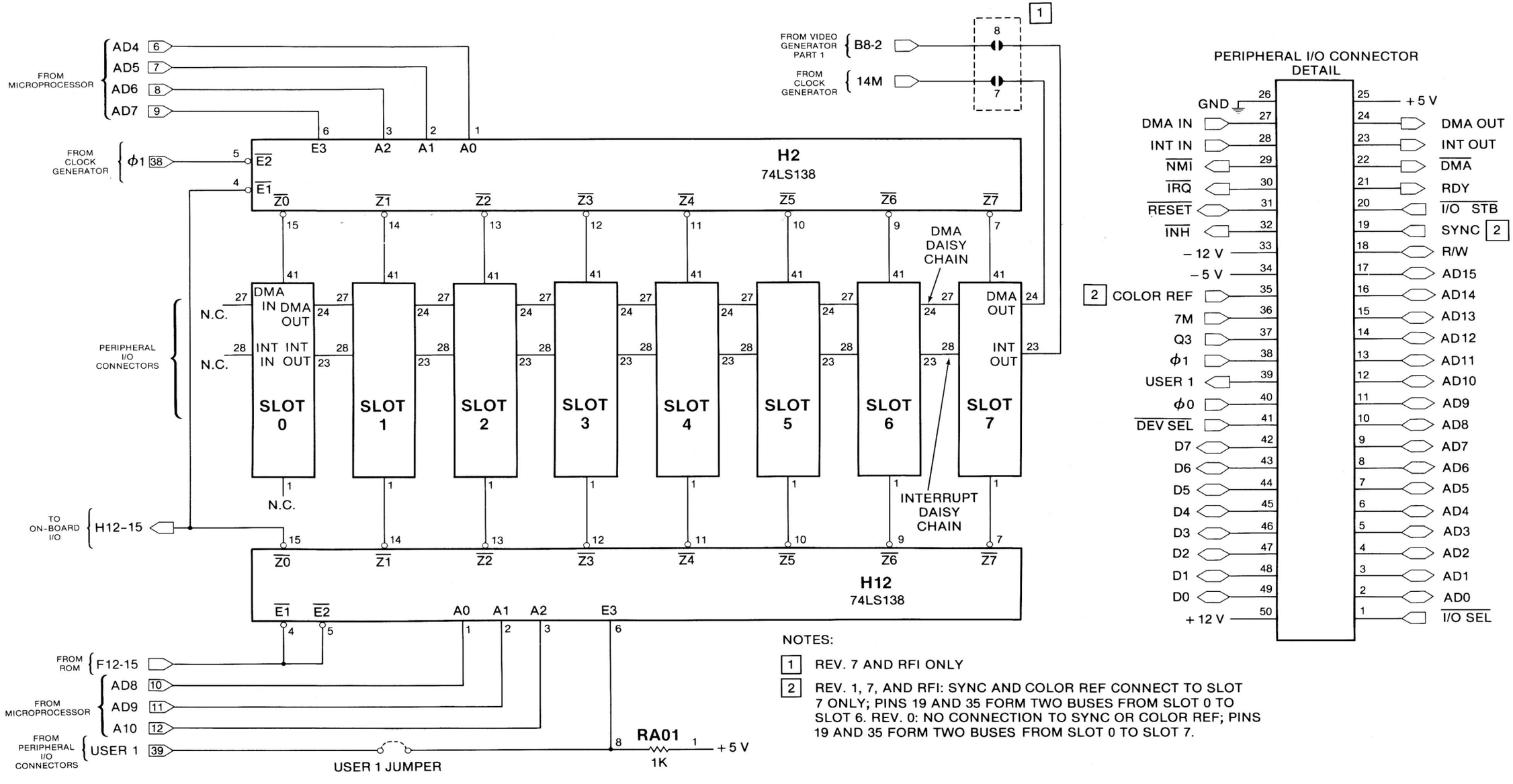


Fig. C-10. ROM (all revisions).



PERIPHERAL I/O CONNECTOR DETAIL

NOTES:

[1] REV. 7 AND RFI ONLY

[2] REV. 1, 7, AND RFI: SYNC AND COLOR REF CONNECT TO SLOT 7 ONLY; PINS 19 AND 35 FORM TWO BUSES FROM SLOT 0 TO SLOT 6. REV. 0: NO CONNECTION TO SYNC OR COLOR REF; PINS 19 AND 35 FORM TWO BUSES FROM SLOT 0 TO SLOT 7.

Fig. C-11. Peripheral I/O (all revisions).

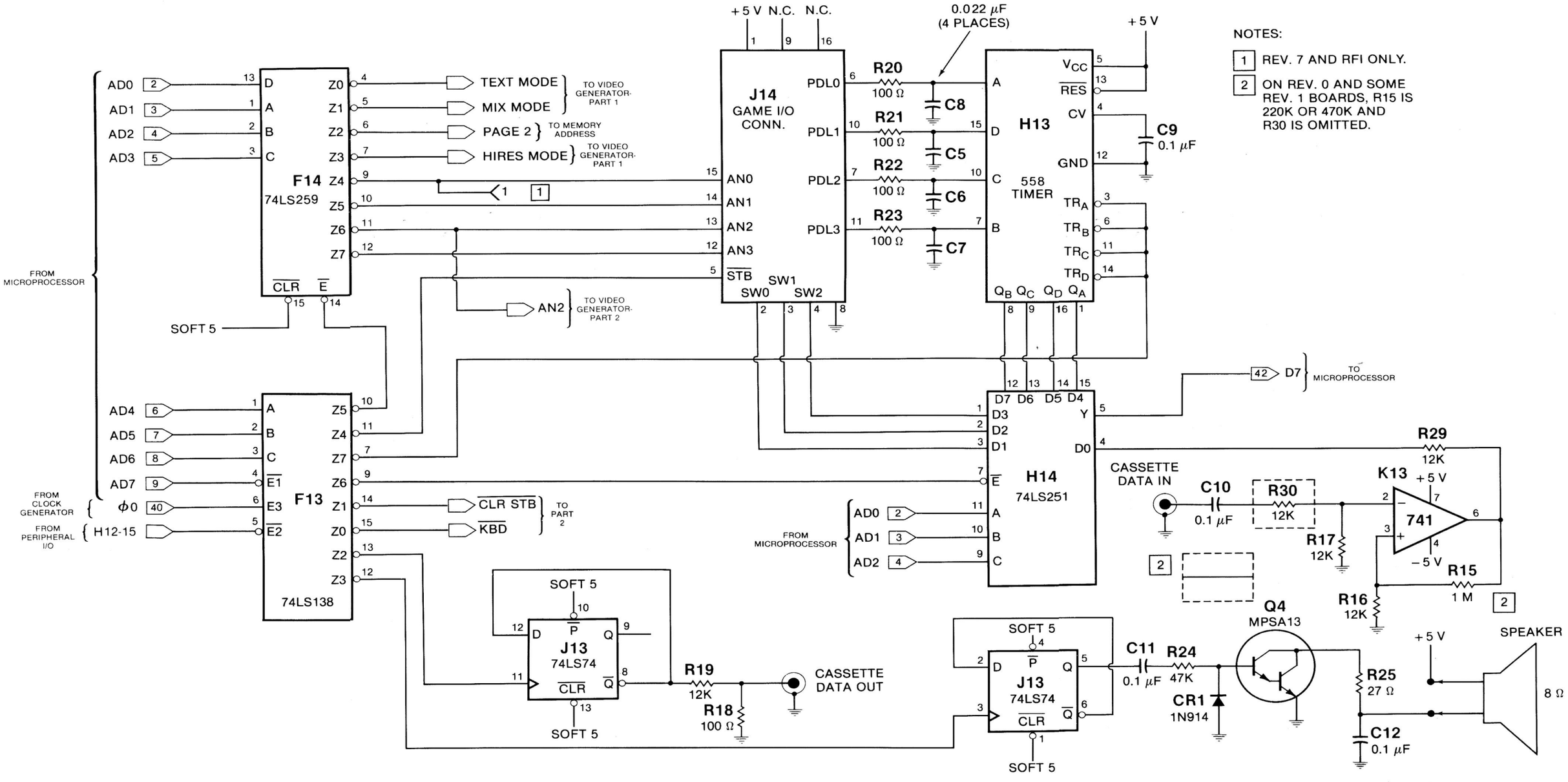


Fig. C-12. On-board I/O—part 1 (all revisions).

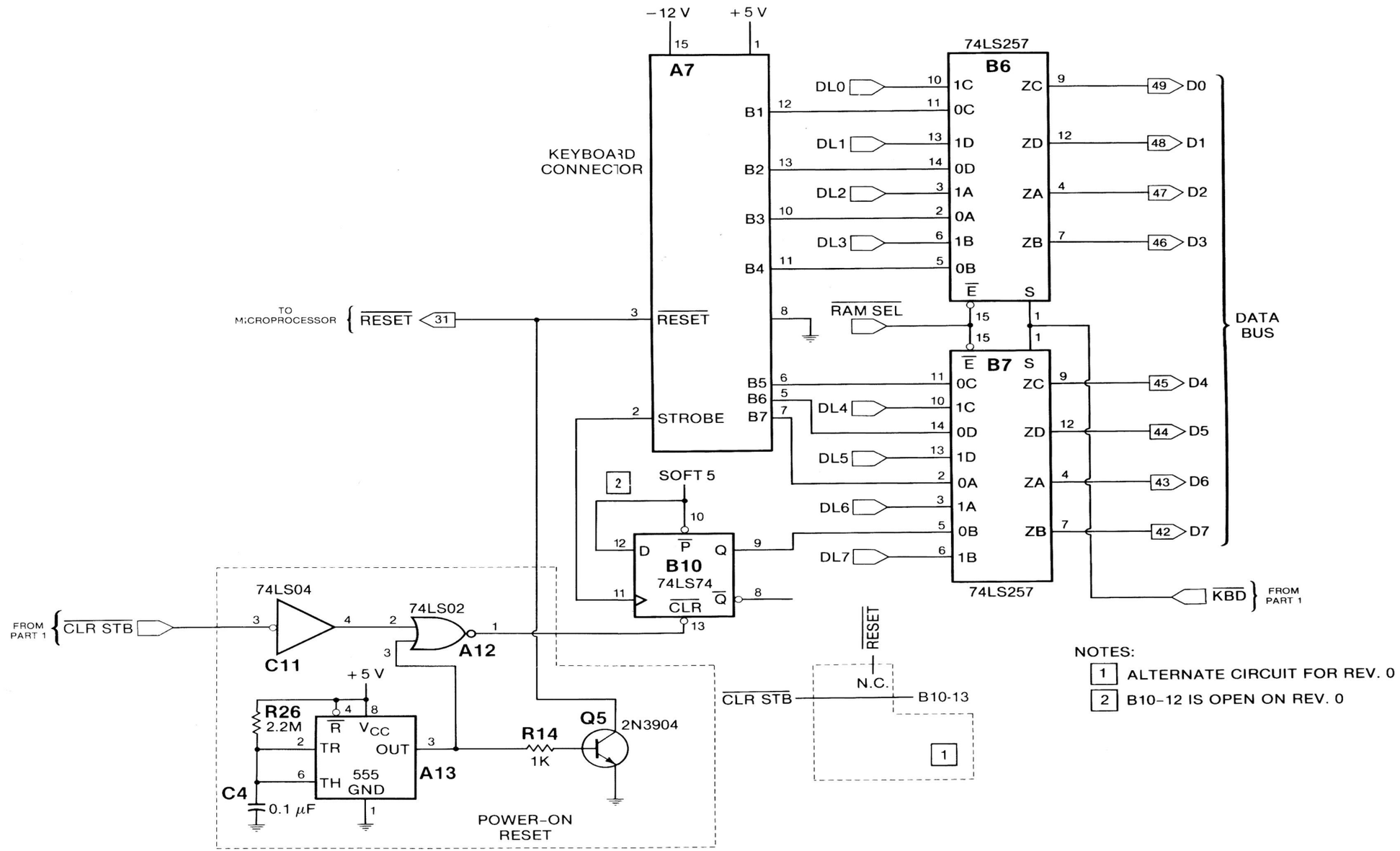


Fig. C-13. On-board I/O—part 2 (all revisions).

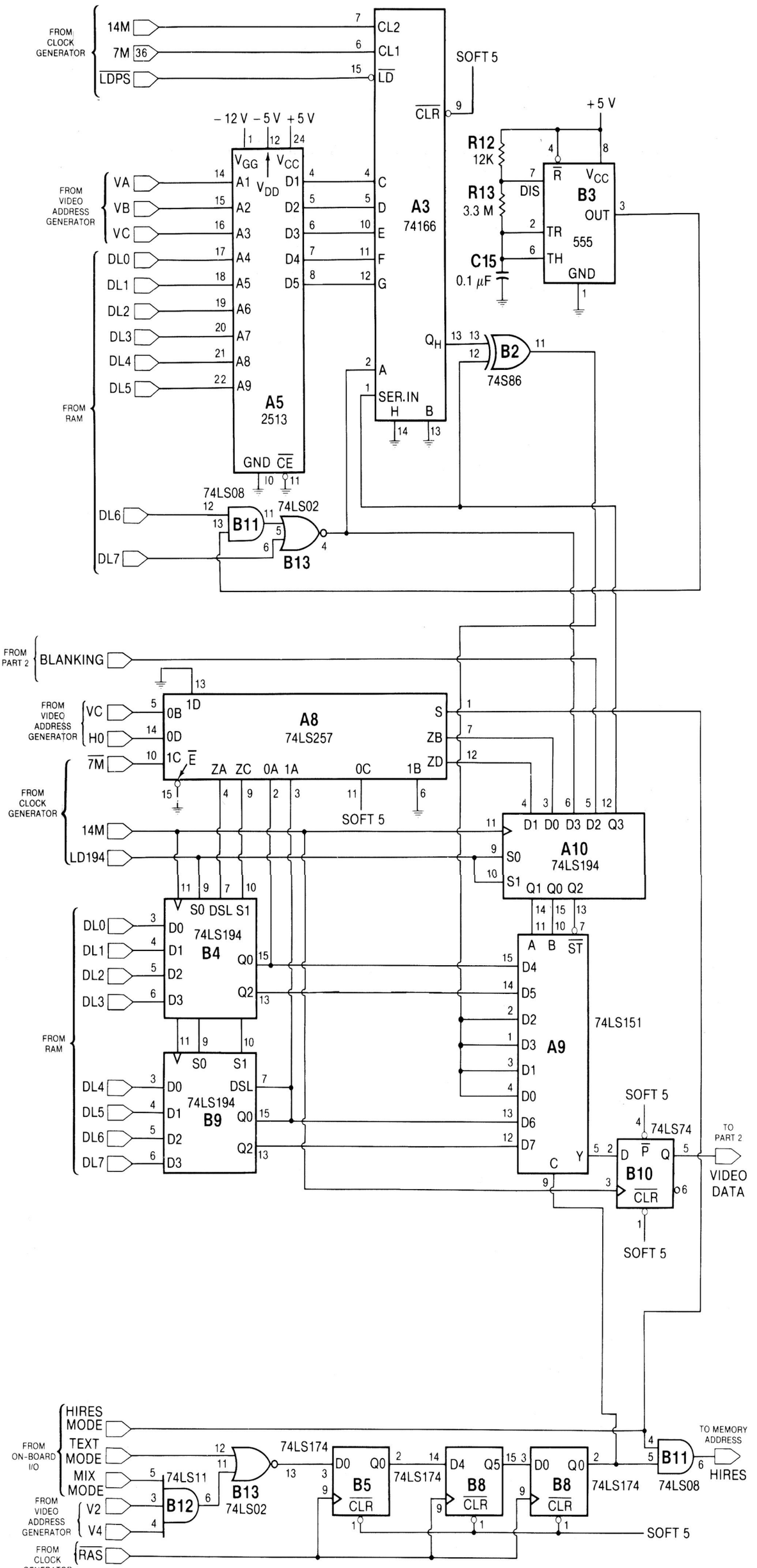
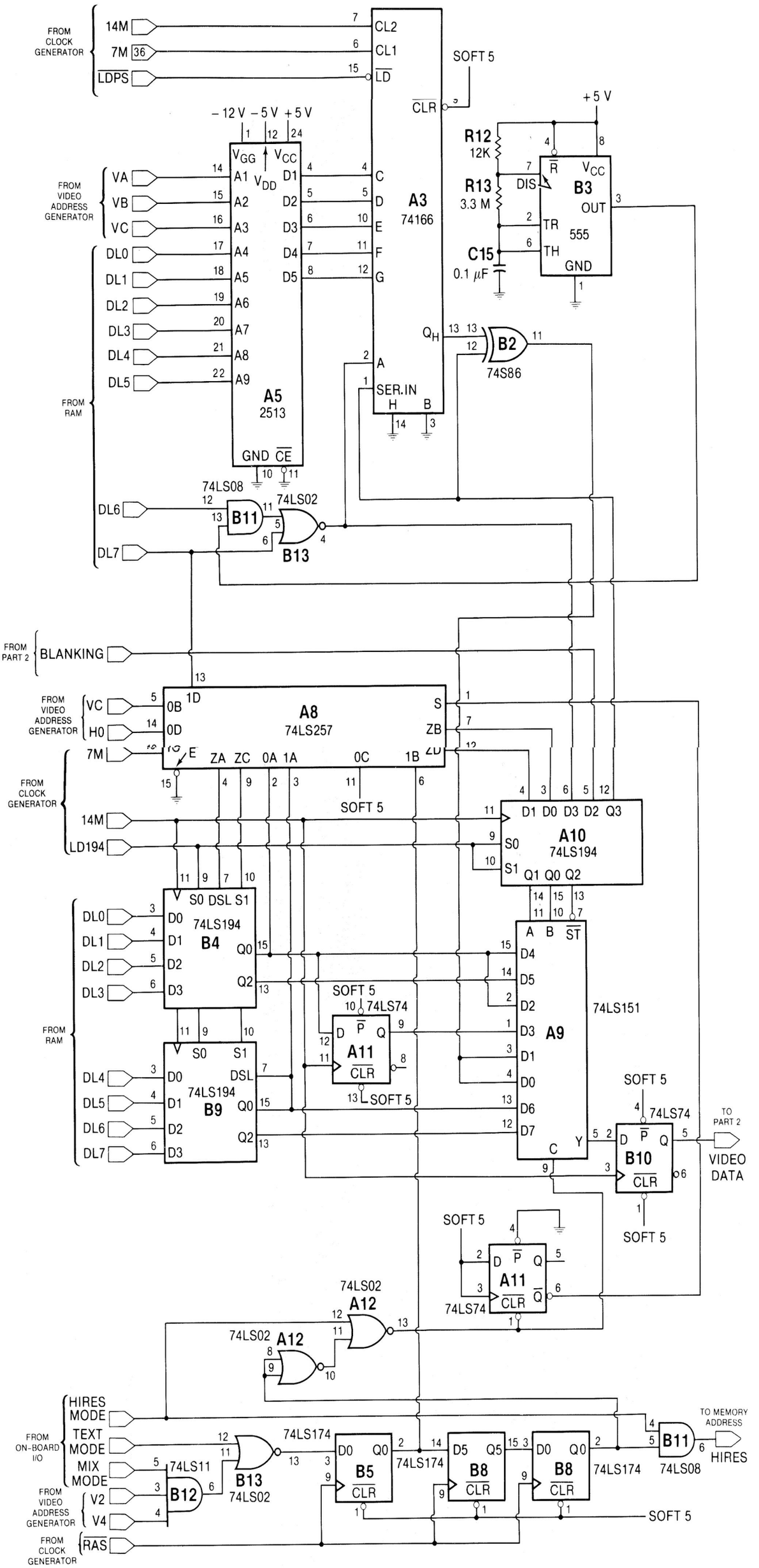


Fig. C-14. Video generator—part 1 (Rev. 0).



o generator—part 1 (Rev. 1).

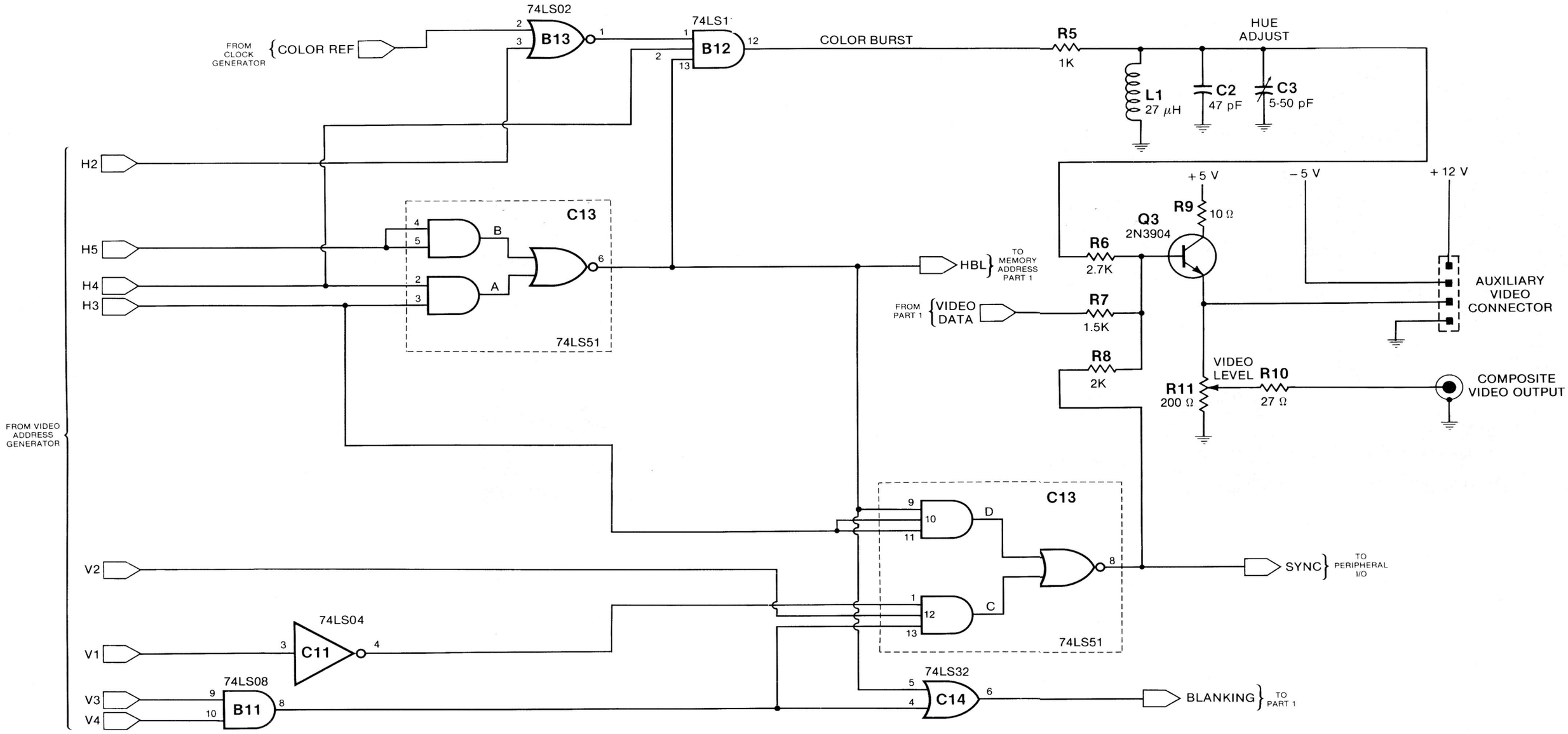


Fig. C-17. Video generator—part 2 (Rev. 0).

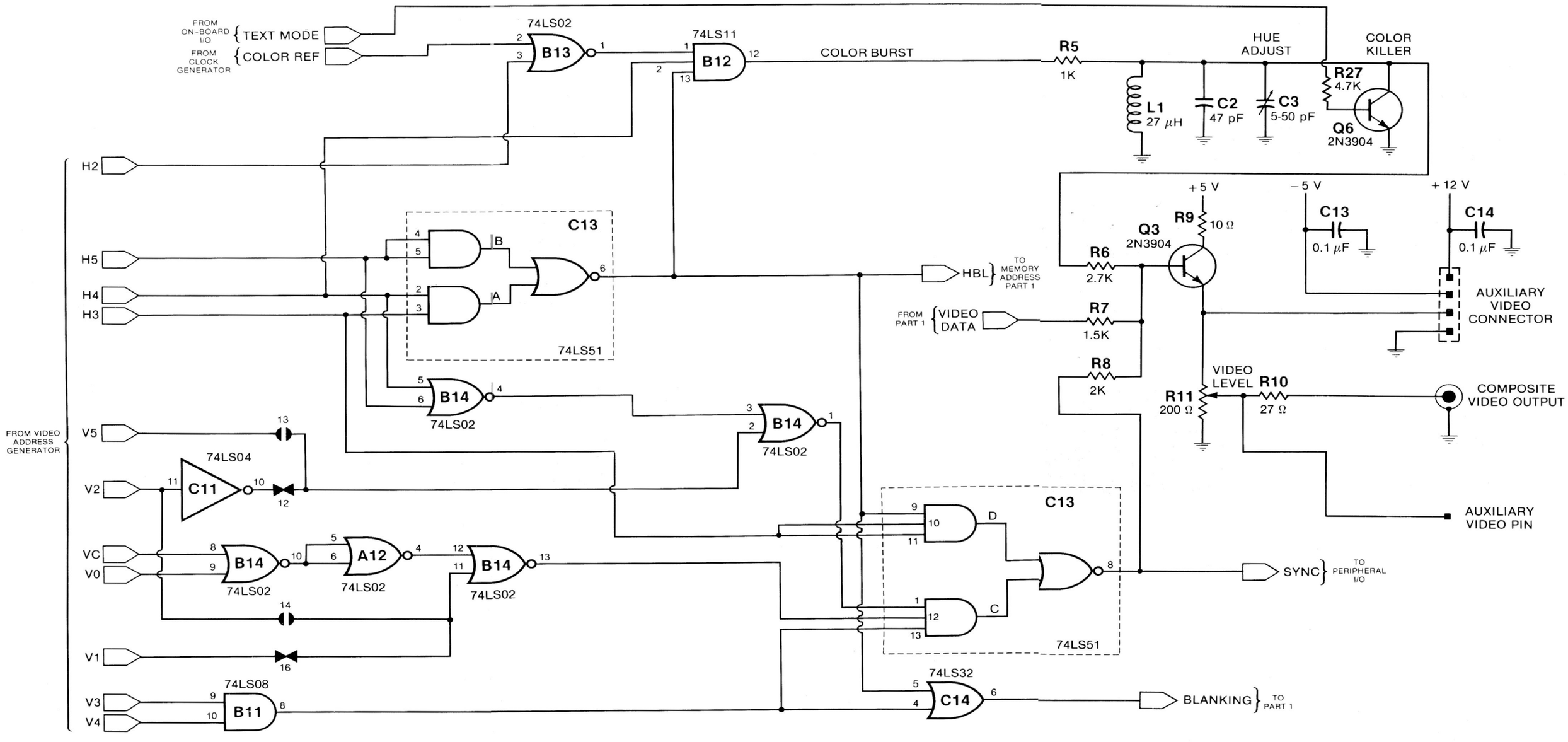


Fig. C-18. Video generator—part 2 (Rev. 1).

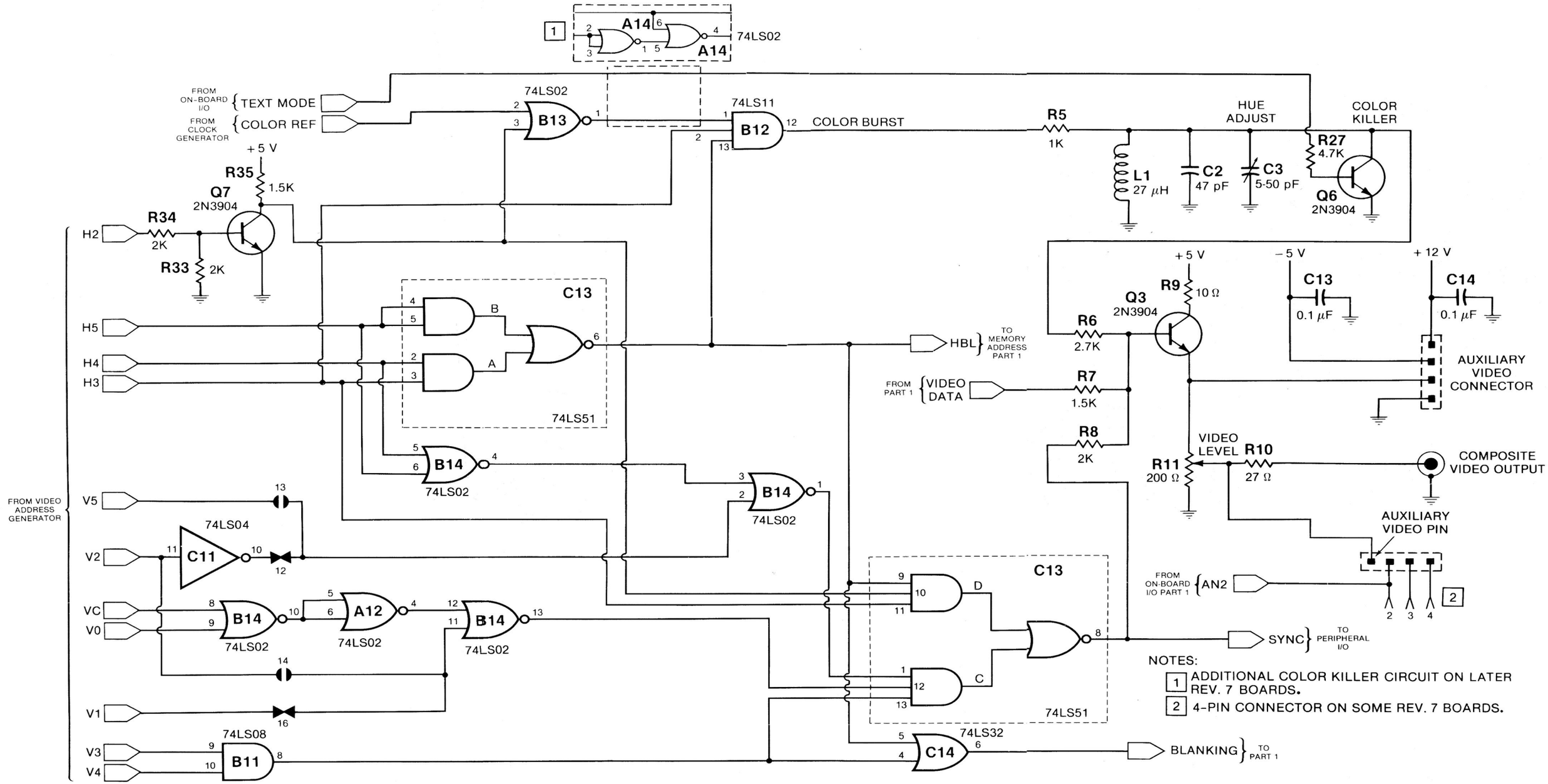


Fig. C-19. Video generator—part 2 (Rev. 7).

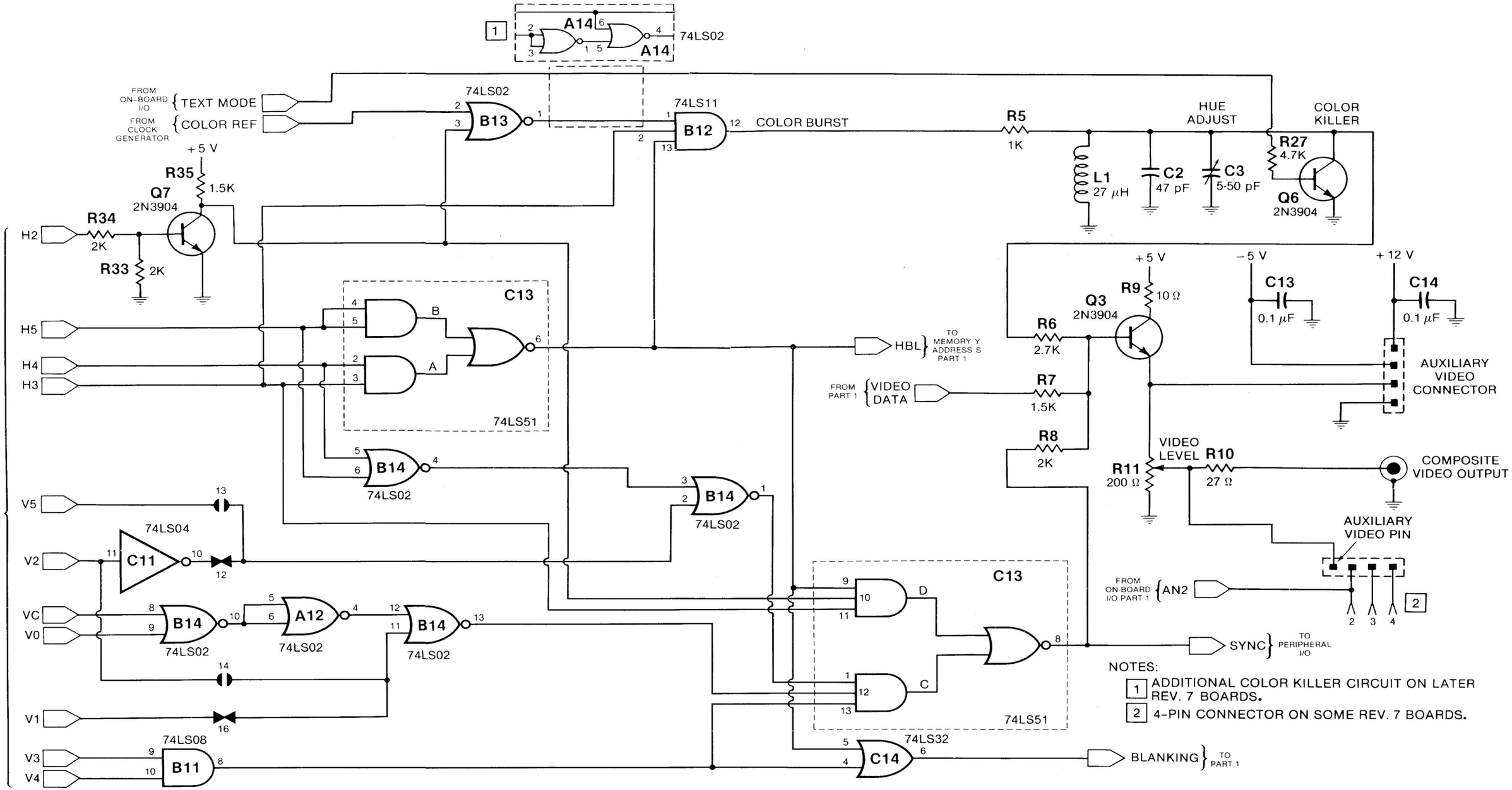


Fig. C-19. Video generator—part 2 (Rev. 7).

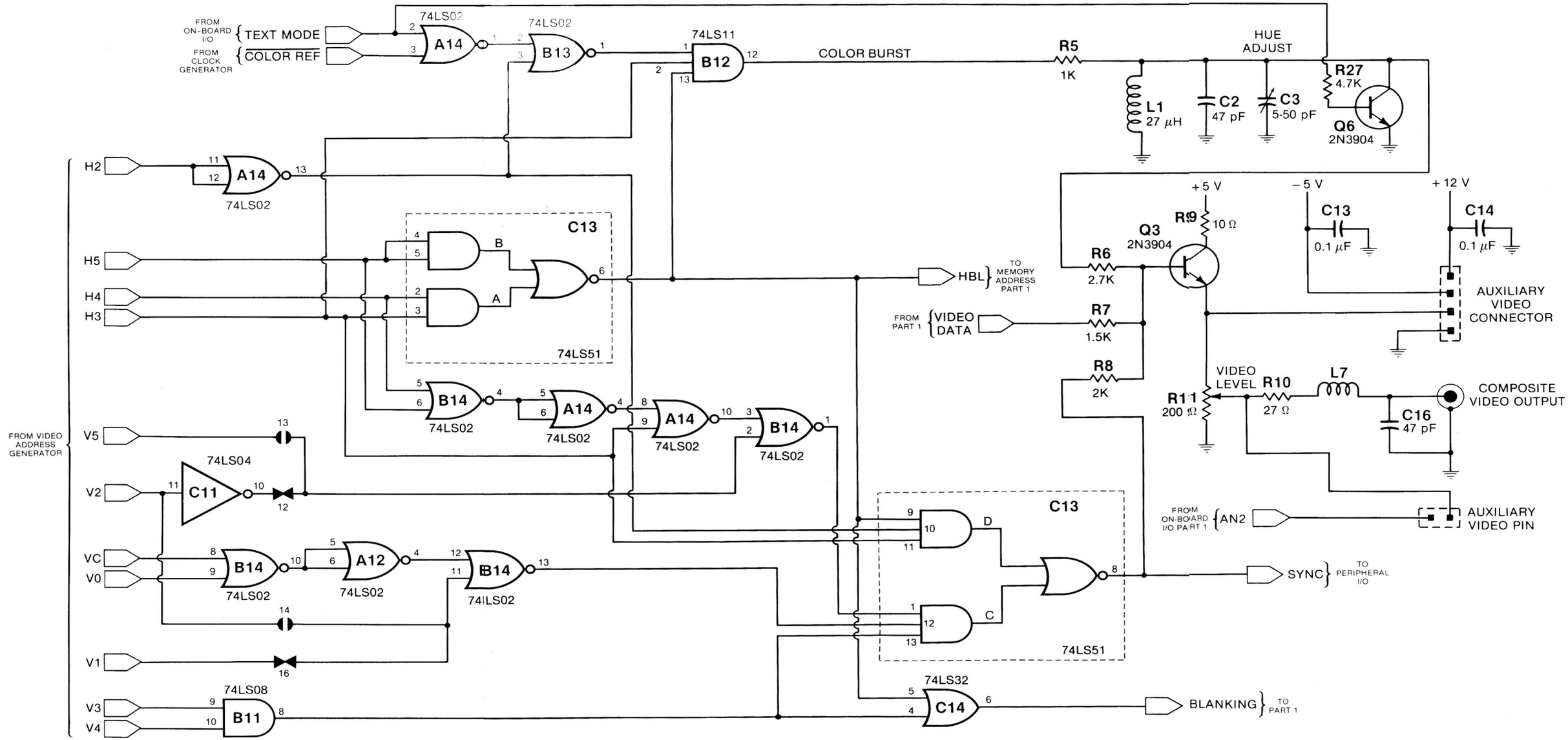
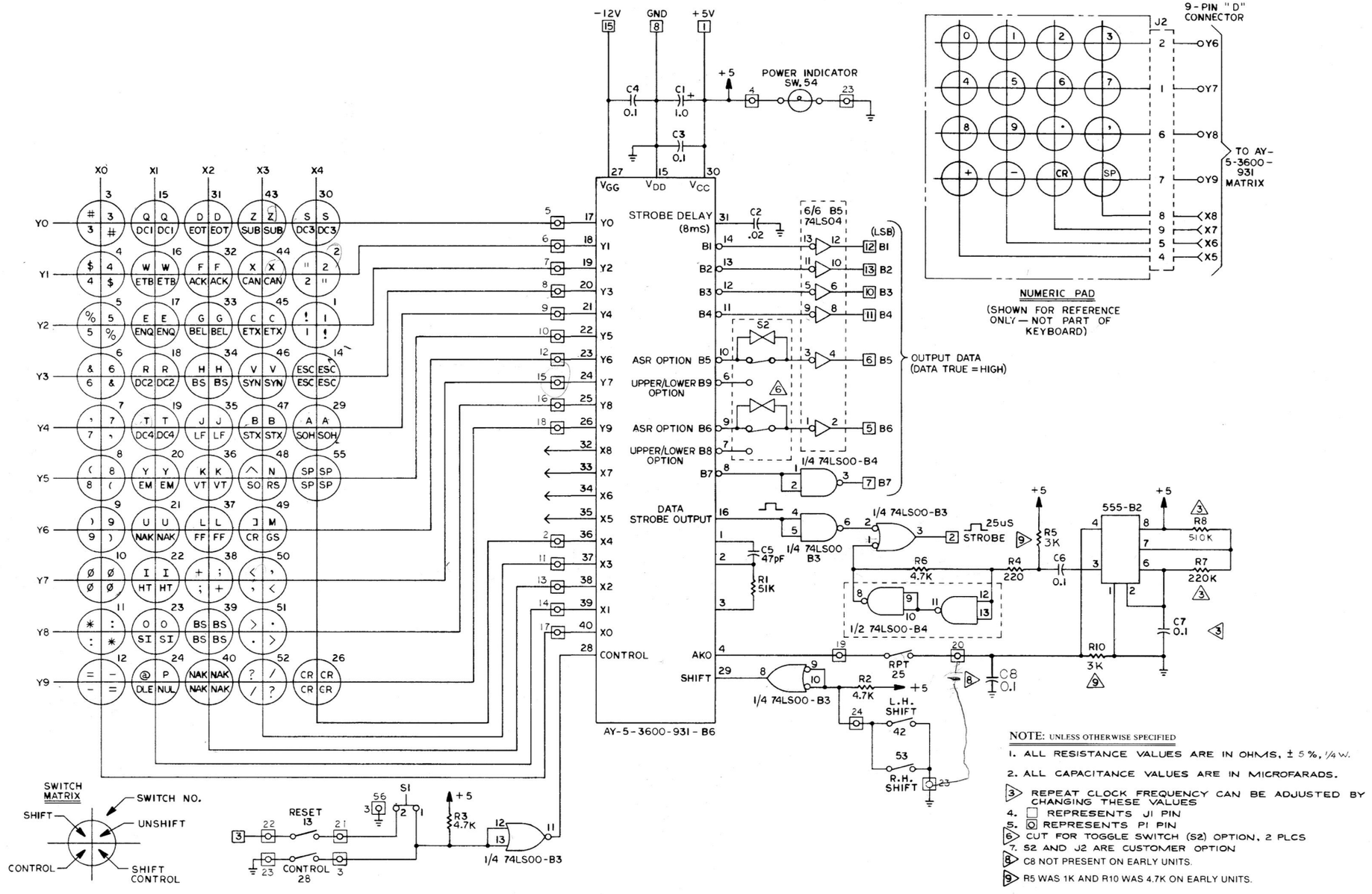


Fig. C-20. Video generator—part 2 (RFI).



NOTE: UNLESS OTHERWISE SPECIFIED

1. ALL RESISTANCE VALUES ARE IN OHMS, ± 5 %, 1/4 W.
2. ALL CAPACITANCE VALUES ARE IN MICROFARADS.
3. REPEAT CLOCK FREQUENCY CAN BE ADJUSTED BY CHANGING THESE VALUES
4. □ REPRESENTS J1 PIN
5. ○ REPRESENTS P1 PIN
6. CUT FOR TOGGLE SWITCH (S2) OPTION, 2 PLCS
7. S2 AND J2 ARE CUSTOMER OPTION
8. C8 NOT PRESENT ON EARLY UNITS.
9. R5 WAS 1K AND R10 WAS 4.7K ON EARLY UNITS.

Fig. C-22. Two-piece keyboard.

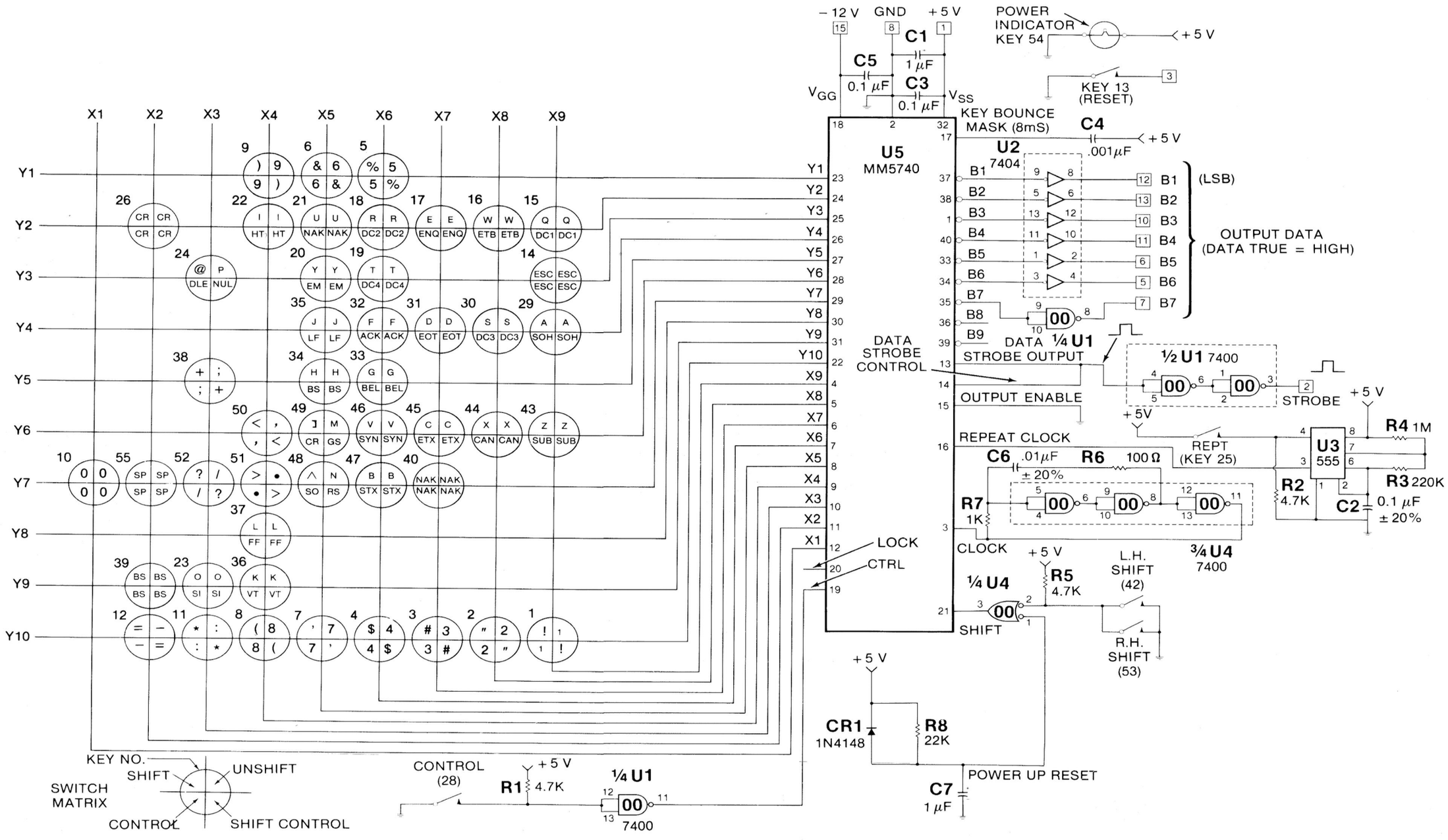


Fig. C-21. Single-piece keyboard.

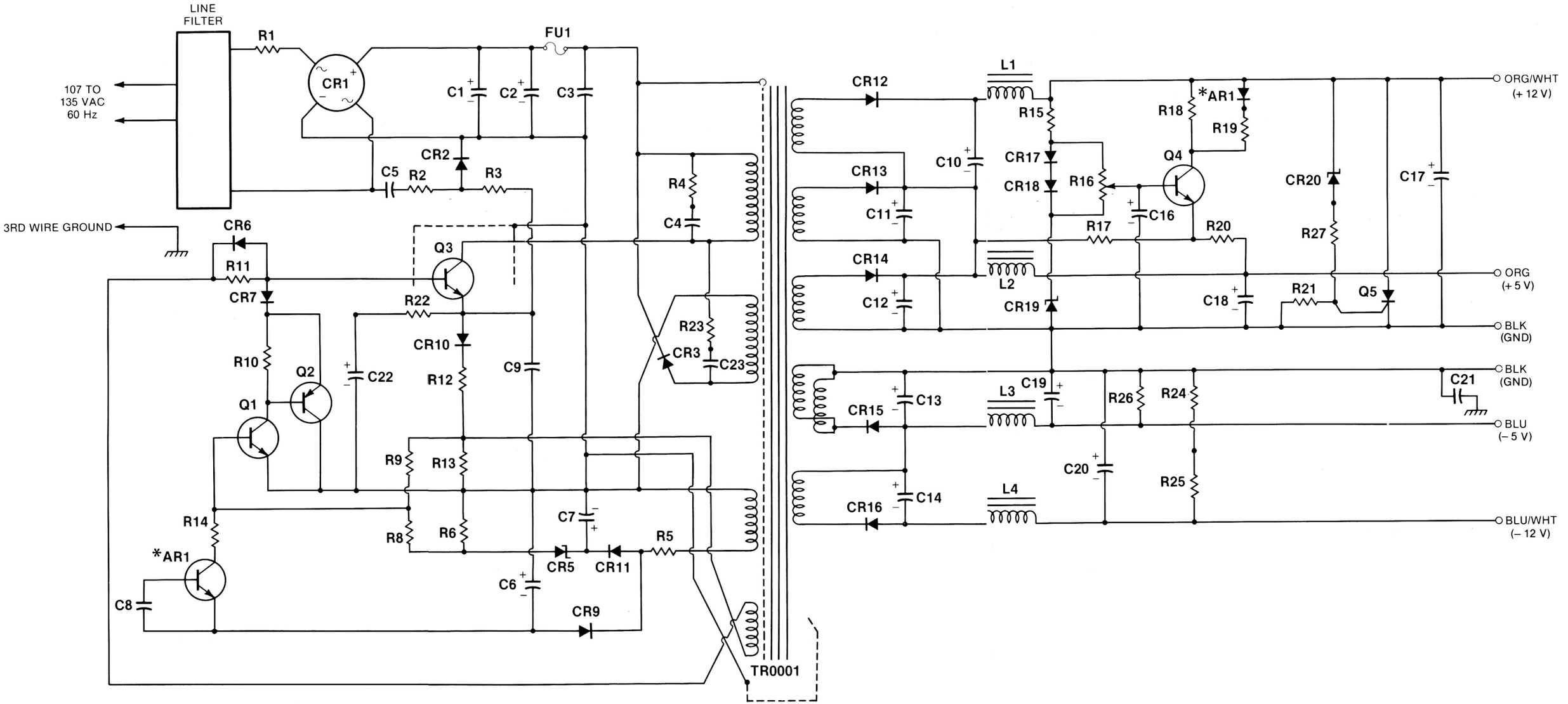
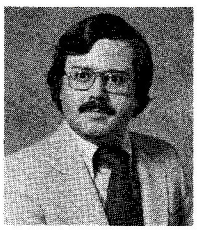


Fig. C-23. Power supply.

THE APPLE II® CIRCUIT DESCRIPTION

- Covers all Apple II® motherboard and keyboard revisions!
- Helps you learn about microcomputer hardware in general and Apple II® hard ware in detail!
- Provides you with accurate schematics and verified waveforms to rely on for servicing and repair!
- Explains the advanced concepts of daisy chains, interrupts, direct memory access, and the ready line!
- Gives you many valuable hints for successful interfacing!
- Contains tutorials on video signals, memory ICs, and the 6502 microprocessor, as well as full explanations of advanced concepts!
- Each chapter contains an overview for the beginner and a detailed section for the more adventurous!
- Ideal for students, technicians, hobbyists, engineers, and others who need Apple II® technical information!



Winston D. Gayler is a design engineer and consultant specializing in microcomputers and digital telephony. He received his Bachelor of Science and Master of Engineering degrees from Cornell University. Now a resident of Silicon Valley, Mr. Gayler is a member of the Audio Engineering Society, the Institute of Electrical and Electronic Engineers, and is also a California Registered Professional Engineer. He has presented technical papers to the IEEE. His work in digital telephony has been published in IEEE and other conference proceedings.

HOWARD W. SAMS & CO., INC.
4300 West 62nd Street, Indianapolis, Indiana 46268 USA

$22.95/21959

ISBN: 0-672-21959-X