LASER™ 128

Technical Reference Manual

VTECH

LASER™ 128

Technical Reference Manual

VTECH

TABLE OF CONTENT

CHAPTER 1

INTRODUCTION

1.1 ASSEMBLY OF THE COMPUTER 5

1.2 THE TOP PANEL 6
1.2.1 KEYBOARD 6
1.2.2 SWITCHES AND INDICATORS 6

1.3 THE EXPANSION SLOT 7

1.4 THE DISK DRIVE 7

1.5 THE SOUND OUTPUT 8

1.6 THE BACK PANEL 8

1.7 THE POWER SUPPLY 9

CHAPTER 2

THE COMPUTER SYSTEM - A FIRST LOOK

2.1 OVERALL SYSTEM 10

CHAPTER 3

THE MICROPROCESSOR

3.1 THE 65C02 MPU 13

3.2 65C02 TIMING 14

CHAPTER 4

MEMORY

4.1 MAIN MEMORY MAP 15

4.2 ROM and I/O FIRMWARE 16
4.2.1 THE COMPUTER BASIC INTERPRETER 17
4.2.2 THE COMPUTER MONITOR 17
4.2.3 I/O FIRMWARE 17
4.2.3.1 I/O FIRMWARE MAP 17
4.2.3.2 I/O FIRMWARE CONTROL 18
4.2.4 SCHEMATIC OF ROM CONTROL 19

4.3 RAM 19
4.3.1 RAM MEMORY USAGE 20
4.3.2 HIGH BANK RAM MEMORY 23
4.3.2.1 HIGH BANK RAM SWITCHES 23
4.3.3 LOW BANK RAM MEMORY 29
4.3.3.1 ZERO PAGE RAM 29
4.3.3.2 $0200 - $BFFF RAM 29
4.3.3.3 DISPLAY BUFFER MEMORY 35
4.3.4 RAM TIMING 42
4.3.5 SCHEMATIC OF RAM CONTROL 43

4.4 HARDWARE PAGE 44

4.5 MEMORY EXPANSION 47

CHAPTER 5

THE VIDEO DISPLAY

5.1 TEXT MODES 50

5.2 GRAPHICS MODES 52
5.2.1 LOW RESOLUTION GRAPHICS 53
5.2.2 MEDIUM RESOLUTION GRAPHICS 53
5.2.3 HIGH RESOLUTION GRAPHICS 54
5.2.4 DOUBLE HIGH RESOLUTION GRAPHICS 56

5.3 MIX TEXT and GRAPHICS MODES 58

5.4 VIDEO DISPLAY GENERATOR (VDG) HARDWARE 58
5.4.1 VIDEO COUNTER 59

5.4.2 BLANKING SYNC and BURST GATE GENERATOR 60
5.4.3 VIDEO ADDRESS MAPPER and RAM ADDRESS MULTIPLEXER 63
5.4.4 VIDEO DATA GENERATOR 65
5.4.5 COLOR ENCODER 69
5.4.6 CHROMA GENERATOR 69
5.4.7 VIDEO SUMMER 71

CHAPTER 6

THE INPUT / OUTPUT PORT

6.1 THE KEYBOARD 72
6.1.1 ACCESSING THE KEYBOARD 76
6.1.2 SPECIAL FUNCTION KEYS and SWITCHES ..77

6.2 SOUND 78

6.3 GAME PORT 78
6.3.1 SWITCH and ANALOG (PADDLE) INPUT 80
6.3.2 THE MOUSE 82

6.4 PARALLEL PRINTER PORT 85

6.5 SERIAL I/O 87

6.6 THE FLOPPY DISK DRIVE 89
6.6.1 FLOPPY DISK DRIVE CONTROLLER 90
6.6.2 HARDWARE LOCATION FUNCTION 91

6.7 EXPANSION SLOT 94

CHAPTER 7

THE COMPUTER FIRMWARE

7.1 SYSTEM CODE 97
7.1.1 POWER-UP AND CONTROL-RESET 97
7.1.2 INTERRUPT AND BRK HANDLER 99
7.1.3 MISCELLANEOUS ROUTINES 101
7.1.4 THE SYSTEM MONITOR 107

7.2 INPUT/OUTPUT ROUTINES 109

7.2.1 PORT 0 : 40-COLUMN DISPLAY ROUTINE ... 111
7.2.2 PORT 3 : 80-COLUMN DISPLAY ROUTINE ... 113
7.2.3 PORT 1 : PRINTER ROUTINES ... 115
7.2.4 PORT 2 : SERIAL COMMUNICATIONS ROUTINES ... 118

CHAPTER 8

SERVICING OF THE COMPUTER

8.1 SYSTEM OVERVIEW ... 121

8.2 CPU ... 122

8.3 ROM ... 123

8.4 RAM ... 124

8.5 ADDRESS DECODING AND BANK SWITCHING ... 126

8.6 TIMING SIGNALS GENERATION ... 127
8.6.1 CIRCUIT OF THE PHASE-LOCKED LOOP (PLL) ... 127
8.6.2 NTSC MODELS ... 128
8.6.3 PAL MODELS ... 129
8.6.4 FREQUENCY FINE TUNING ... 129

8.7 RESET CIRCUIT ... 130
8.7.1 THE CPU RESET CIRCUIT ... 130
8.7.2 THE GATE ARRAY RESET CIRCUIT ... 130

8.8 KEYBOARD ... 131

8.9 SPEAKER ... 133

8.10 THE VIDEO CIRCUIT ... 133

8.11 PARALLEL PRINTER ... 135

8.12 SERIAL I/O PORTS ... 135

8.13 MOUSE ... 136

8.14 HAND CONTROL/JOYSTICK ... 138

8.15 DISK DRIVE 138

8.16 THE POWER SUPPLY 140
8.16.1 AC/DC ADAPTOR 141
8.16.2 THE REGULATOR BOX 141

8.17 EXPANSION SLOTS 143

8.18 SERVICE FLOW CHART 147

APPENDIX

A) 65C02 PROGRAMMING SPECIFICATIONS A1

B) CUSTOM IC PIN ASSIGNMENT B1

C) 6551 DATA SHEET C1

D) KEYBOARD LAYOUT D1

E) SCHEMATICS E1

F) PARTS LIST OF THE COMPUTER F1

G) PCB AND COMPONENT LAYOUT G1

CHAPTER 1

INTRODUCTION

This chapter gives the user an overall view of the computer.

1.1 ASSEMBLY OF THE COMPUTER

Fig 1.1 shows the exploded view of the computer.

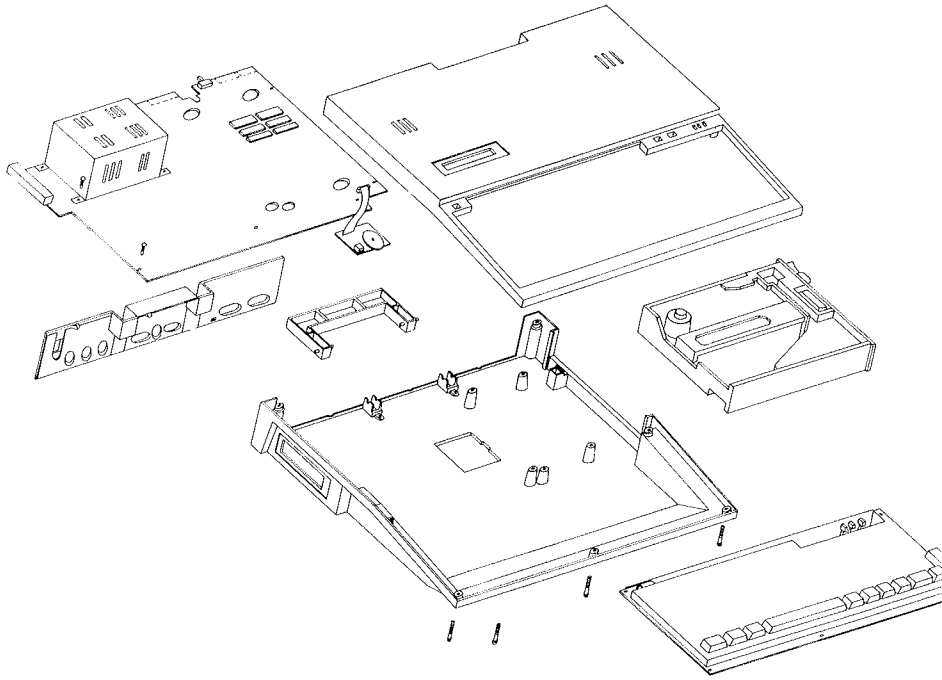


Fig 1.1 Exploded view of the computer

1.2 THE TOP PANEL

1.2.1 Keyboard

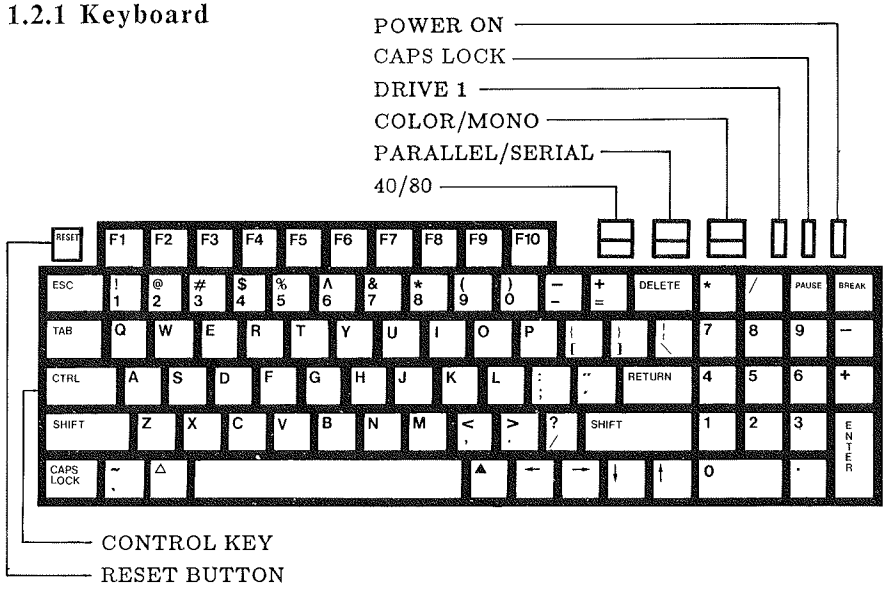


Fig 1.2 USA standard keyboard layout

Fig 1.2 shows the keyboard (USA standard or Sholes keyboard) of the computer.

It has a standard typewriter layout. In addition, on the right side there is a calculator type key-pad. On the top, there are 10 function keys.

The keyboard layouts of different versions of the computer for different countries will differ. Details are found in APPENDIX D.

1.2.2 THE SWITCHES and INDICATORS

On the upper right corner of the keyboard panel, there are three slide switches and three indicators. (Fig 1.2).
Some software will detect the 40 / 80 switchs when setting screen text in either 40 columns or 80 columns. Next to the 40 / 80 switch is a switch to select parallel or serial printer. The last one is the MONO / COLOR switch to select a monochrome or color display.

When the DRIVE 1 lamp is on, the built-in dive is in use. The CAPS LOCK indicator shows the current setting status of the CAPS LOCK key. The last indicator will be on when power is turned on.

1.3 THE EXPANSION SLOT

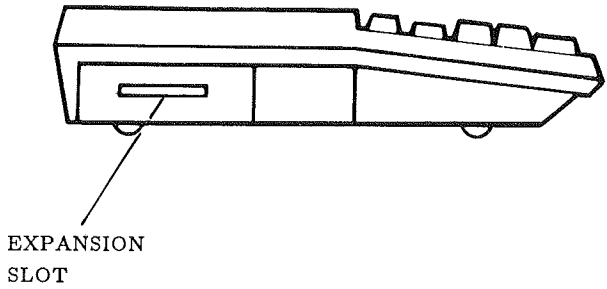


Fig 1.3 Expansion slot

There is one expansion slot on the left side of the computer. APPLE®II expansion peripheral card can be inserted to this slot.

⚠It is very important that never try to insert or remove a peripheral card to or from the expansion slot when the computer is turned on. Otherwise unpredictable damage may be made to the computer and the peripheral.

1.4 THE DISK DRIVE

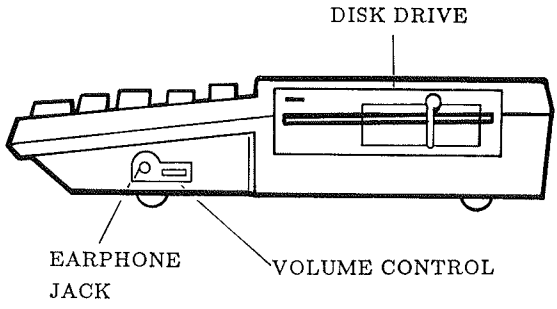


Fig 1.4 Disk drive, volume control and earphone jack

One 5 1/4" disk drive is built-in on the right side of the computer.

1.5 THE SOUND OUTPUT

There is a speaker inside the cabinet of the computer. The loudness of the speaker can be adjusted by a volume control on the right side of the computer. An earphone jack is next to the volume control. Plugging an earphone plug into this jack disconnects the internal speaker.

1.6 THE BACK PANEL

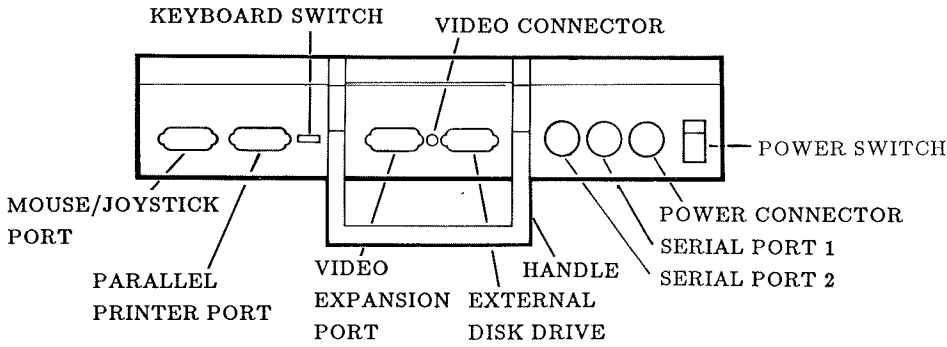


Fig 1.6 Back panel connectors and switches

The back panel has 8 connectors and 2 switches. From right to left, they are:

- a Power switch
- a 7-pin DIN connector for power input
- a 5-pin DIN connector for serial printer (serial port 1)
- a 5-pin DIN connector for modem (serial port 2)
- a 19-pin D-type connector for second drive
- an RCA-type jack for a composite video monitor
- a 15-pin D-type connector for video expansion
- a switch to select standard or alternate keyboard layout
- a 15-pin D-type connector for parallel printer
- a 9-pin D-type connector for hand controllers, mouse, or joystick or some other pointing device.

Besides, there is an opening on the bottom for you to adjust the speed of the internal disk drive.

* In some versions of the computer, there is a 50 / 60 Hz switch on the bottom for you to select appropiate video circuitry. If you are using the APPLE® IIC LCD panel, you must switch to 60 Hz.

Right in the middle of the bottom, a door can be found. If you open this door, you will see one or two 28 pin IC sockets. If there is only one ROM, it should be a 256K bytes ROM. However, in some units two 128K bytes EPROMs are used. They are equivalent to one 256K bytes ROM. They contain the monitor, BASIC interpreter and I/O firmwares.

1.7 THE POWER SUPPLY

The input requirement of the power supply is 17V DC and 1.8A.

An AC to DC adaptor comes with the unit. You may easily plug the AC connector to the AC supply and connect the DC plug to the main unit. Then turn on the power switch and the computer will be in operation.

The NTSC version AC / DC adaptor will be of AC input 120V. Those PAL version AC / DC adaptor will have an AC input of 240V.

When the computer is in operation, pull down the handle such that the computer tilts. This helps to dissipate heat and helps typing.

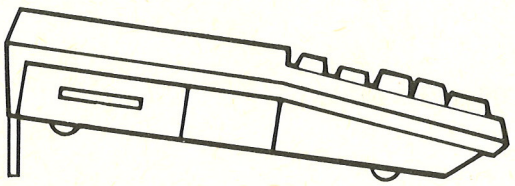


Fig 1.7 When in operation pull down the handle for heat dissipation

Furthermore, NEVER put a monitor or television set on the top cabinet of the computer.

CHAPTER 2

THE COMPUTER SYSTEM - A FIRST LOOK

This chapter introduces to the user the overall system and its subsystems. Further discussions of individual subsystems can be found in later chapters.

2.1 OVERALL SYSTEM

Fig 2.1 gives a block diagram of the computer system.

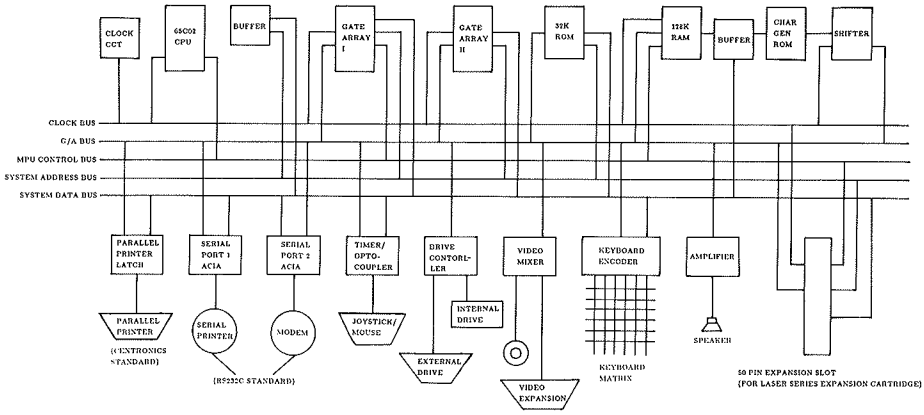


Fig 2.1 Block diagram of the computer system

The system can be divided into 5 subsystems:

. The MPU subsystem

  The computer uses 65C02 as its MPU.

. The memory subsystem

  . ROM

  The ROM houses

  . 12K BASIC interpreter and monitor

  . 8K I/O firmware

. RAM

Two banks of 64K RAM comprise the 128K RAM system.

. The video subsystem

This part generates all the text modes and graphics modes timing signals

. 40 - column text mode
. 80 - column text mode
. Low resolution graphics
. Medium resolution grpahics
. High resolution graphics
. Double high resolution graphics

One video expansion port is available besides the composite video output. Optional TV systems standards are available: NTSC, PAL or PERITEL

. The I/O subsystem

It includes

. keyboard
. speaker
. joystick / mouse port
. parallel printer port
. Two serial ports (Port 1 for Serial printer and Port 2 for Modem)
. Built-in disk drive
. Second drive port
. 50 - pin expansion slot

. The power converter subsystem

The input of this power converter is 17V(+8V)/-3V) DC and 1.8A

. The outputs are

+5V 1.5A

+12V 1A

-12V 0.2A

The power supply utilizes efficient switch-mode power supply.

CHAPTER 3

THE MICROPROCESSOR

The chapter is an introduction to the microprocessor - the brain of the computer.

3.1 The 65C02 MPU

The 65C02 is an enhanced version of the 6502. It is a CMOS device and thus requires less power. Besides it has new instructions and addressing modes. Details of the 65C02 instructions can be found in APPENDIX A.

The schematic of the MPU subsystem is illustrated in Fig 3.1.

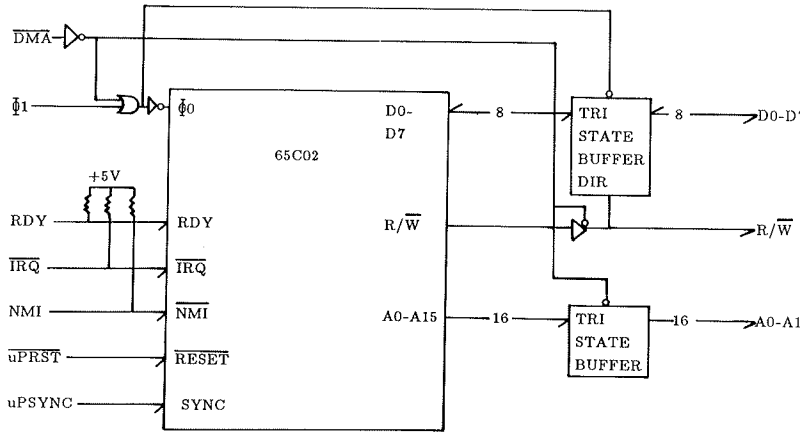


Fig 3.1 Block diagram of the MPU subsystem

In the MPU subsystem, the data bus and the address bus of the 65C02 are buffered by tri-state buffers.

The $\overline{\text{DMA}}$ signal can be used to halt the CPU temporarily. By pulling low the $\overline{\text{DMA}}$, the CPU clock is pulled low and it stops its current execution until this pin is high again. An external device can then use the address bus and data bus. However, the CPU has only dynamic storage. This means that the CPU clock must not be

pulled low for too long. Otherwise, the data in the CPU's registers would be lost.

3.2 65C02 TIMING

A 1 MHz clock ($\bar{\Phi}0$) is tied to the 65C02. The data bus timing is shown in Fig 3.2.

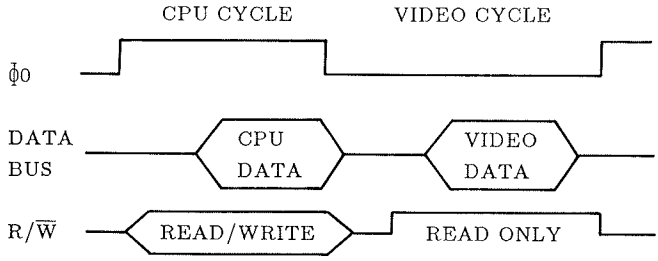


Fig 3.2 Data bus and R/W signal Timing Diagram

When Φ0 is high, it is called the CPU cycle because the CPU will control the address bus and data bus. When Φ0 is low, the video circuitry will generate video address and use the data bus. So it is called the video cycle. The clock signal Φ0 is derived from the F14M clock source.

CHAPTER 4

MEMORY

The ROM and RAM memory system will be investigated in this chapter.

4.1 MAIN MEMORY MAP

The map of Fig 4.1 shows the partitions of memory. The memory can be divided into four areas:

a) ROM (Read Only Memory)
b) RAM (Random Access Memory)
c) I/O (Input / Output)
d) Hardware page

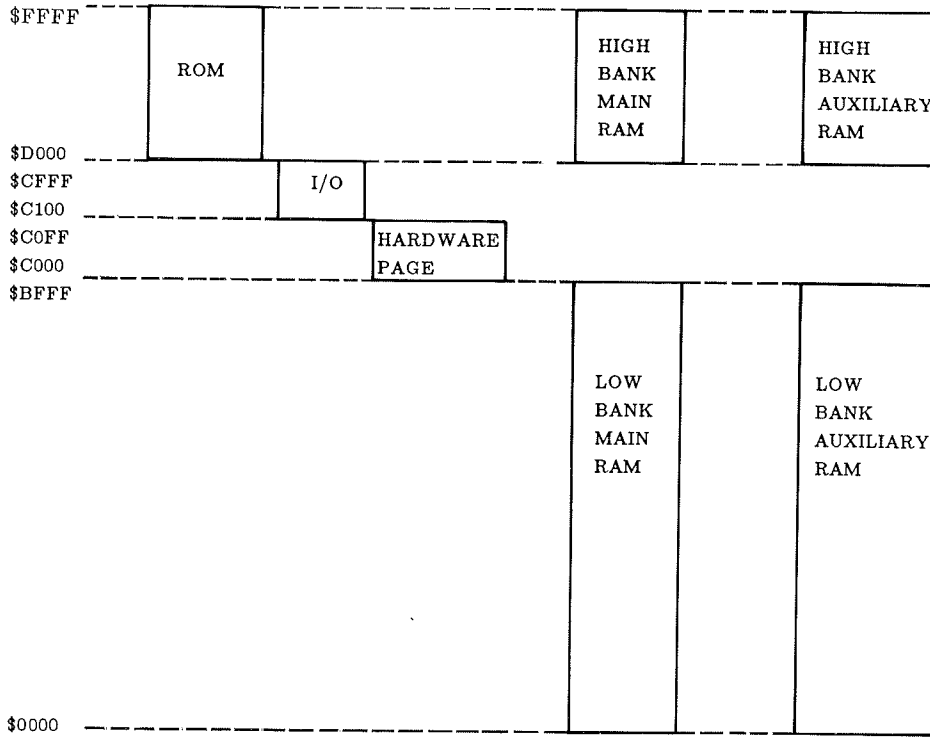


Fig 4.1 Memory map

The 65C02 has 16 address lines. It can access only 64K bytes of memory. In order to access more than 64K bytes of memory, the bank-switch technique is employed. There are bank registers to configure the physical memory to the 64K bytes of memory.

4.2 ROM AND I/O FIRMWARE

The ROM consists of three parts:

a) BASIC interpreter

b) The computer monitor

c) The computer I/O firmware

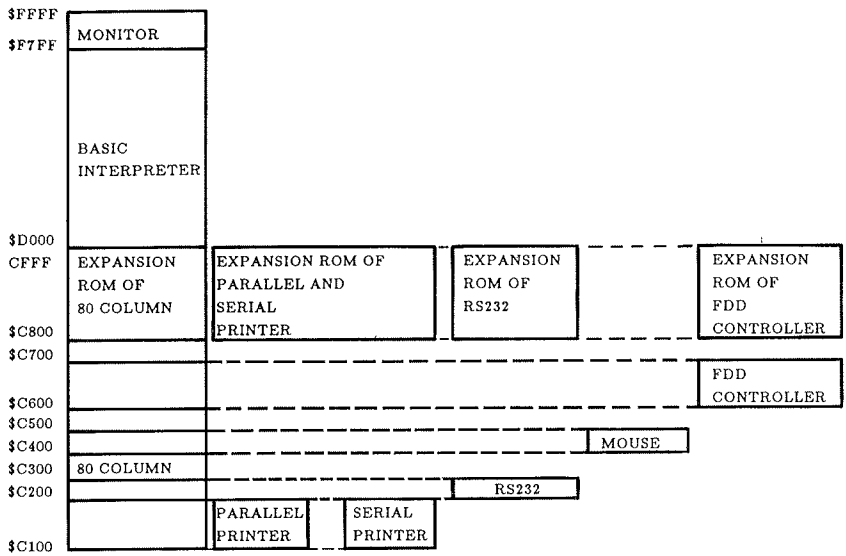


Fig. 4.2 The ROM memory mapping

The INTIOROM is a soft-switch which controls the I/O ROM banks configuration. Soft-switches are widely used in the computer. The logical state of the soft-switch is set by software. A summary of all soft-switches is listed in TABLE 4.5.

4.2.1 The computer BASIC interpreter

The address range is from $D000 to $F7FF. Details of the BASIC commands are in the main unit manual.

4.2.2 The computer monitor

The monitor occupies $F800 to $FFFF. The basic I/O subroutines are in the monitor. You can enter the monitor by typing CALL-151 and RETURN.

4.2.3 I/O firmwares

4.2.3.1 I/O firmwares map

The I/O firmwares' partitions are illustrated in Fig 4.3.

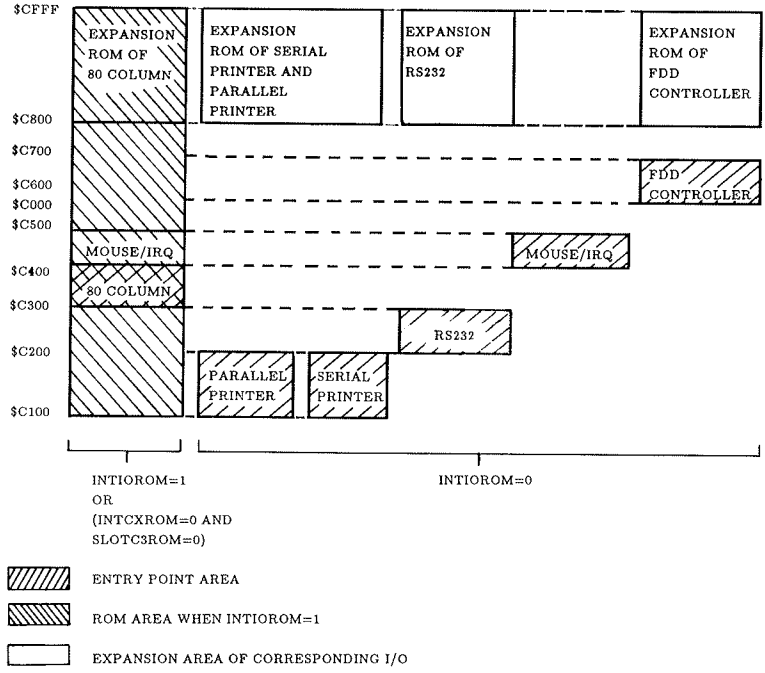


Fig 4.3 I/O firmware areas

The only I/O firmware when INTIOROM is active is 80 column text mode. When INTIOROM remains inactive. i.e. INTIOROM = 0, the possible I/O activities are:

. parallel printer
. serial printer
. RS232
. 80 column text mode
. mouse
. FDD (Floppy Disk Drive) controller

At any one time, only one kind of printer can be used (either parallel or serial). This is selected by a switch on the front panel of the computer.

4.2.3.2 I/O firmware control

a) INTIOROM =1

If INTIOROM equals one, all the internal ROM area ($C100 to $CFFF) can be accessed freely.

b) INTIOROM =0

If 80ROM is reset, then the internal $C3XX (for 80 column) and its expansion ROM cannot be accessed. Should 80ROM be active, the 80 column firmware can be accessed.

The expansion ROM can be accessed after reading from or writing to any location within $C300 - $C3FF. The expansion ROM is turned off by accessing $CFFF.

For all I/O firmware, their expansion ROM can be turned off by accessing $CFFF.

If reading from or writing to that I/O firmware entry point region, their expansion ROM will be automatically turned on.

. The parallel printer and serial printer share the same expansion ROM ($C800 - $CFFF) so only one type of printer can be active at any one time.

. RS232 firmware occupies the memory address range $C200-$C2FF.

. The mouse firmware only occupies $C400 - $C4FF. Accessing any location in this range cannot affect any $C800 - $CFFF expansion ROM.

. The floppy disk drive controller firmware occupies from $C600-$C6FF.

4.2.4 Schematic of ROM

The ROM control logic divides into two parts, namely:

a) High order address translator
b) ROM output enable

Since different banks of physical ROM memory share the same bank of logical memory, the ROM addresses have to be translated so as to support multi-bank ROM memory. The functional block diagram is shown in Fig 4.4.

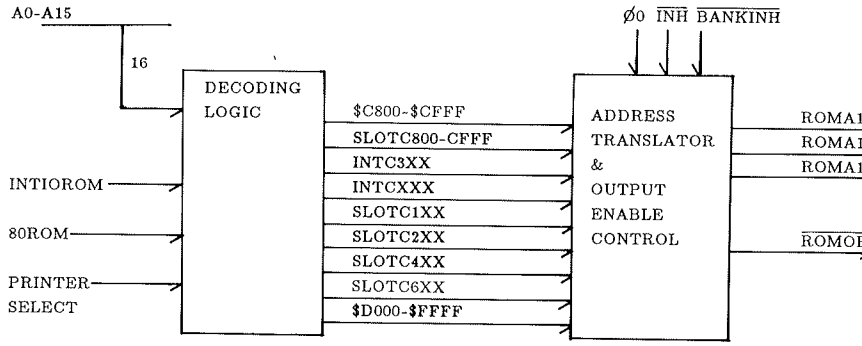


Fig 4.4 Block diagram of ROM address translation and ROM enable

4.3 RAM

The RAM subsystem can be divided into:

a) Low bank main RAM ($0000 - $BFFF)
b) High bank main RAM ($D000 - $FFFF)
c) Low bank auxiliary RAM ($0000 - $BFFF)
d) High bank auxiliary RAM ($D000 -$FFFF)

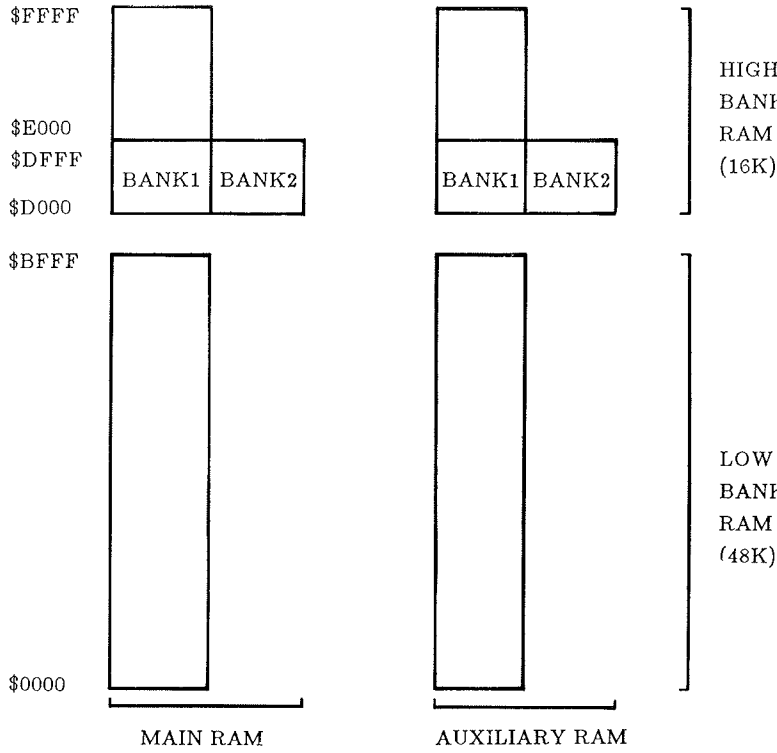


Fig 4.5 RAM map

The operation of these memory banks will be discussed in details in following sections.

4.3.1 RAM memory usage

Some RAM memory is dedicated for special purpose due to the CPU and system requirement. However most of the RAM memory are free areas where the user's program can be input. By knowing more of the RAM functional partition, the user can manage the RAM more efficiently.

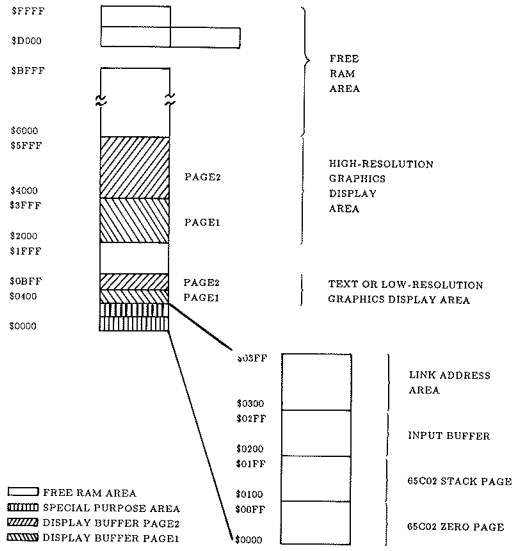


Fig 4.6 RAM memory usage map

.Page zero

Page zero RAM is used extensively by the system. The user should use the page zero area carefully if he really needs to do so.

. Page one

The 65C02 uses page 1 as its stack. When a subroutine call or an interrupt occurs the microprocessor will push the return addresses onto the stack. Upon return from a subroutine or an interrupt the address will be pulled from stack. Many programs also use the stack for temporary storage of variables or parameters. The 65C02 operates the stack on a first-in, last-out basis.

. Page two

This is the keyboard input buffer used by the Monitor and the BASIC interpreter.

. Page three

The DOS and Monitor use it to store link addresses or vectors. Since only the upper part of page 3 is used by the computer, most of the page 3 area may be used freely.

ADDRESS	FUNCTION
03F0 03F1	BRK request vector used by Monitor
03F2 03F3	Reset vector
03F4	Power-up signature byte
03F5 03F6 03F7	Jump instruction to the subroutine that handles BASIC "&" commands
03F8 03F9 03FA	Jump instruction to the subroutine that handles user (CTRL - Y) commands
03FB 03FC 03FD	Jump instruction to the subroutine that handles non-maskable interrupt
03FE 03FF	Interrupt Request vector

Table 4.1 Page three usage: vectors used by Monitor

. Video buffer pages

The text mode and low resolution graphics mode share the same display buffer. Video buffer page one is from $0400 - $07FF and page two is from $0800 - $0BFF. High resolution graphics page one uses $2000 - $3FFF and the video buffer page two is from $4000 - $5FFF. If the display buffer is not used for display, it can be used by the user as program area.

. Free RAM area ($6000 - $BFFF)

The user's program can be freely placed in this RAM area.

4.3.2 High-bank RAM memory ($D000 - $FFFF)

This RAM memory refers to address $D000 - $FFFF. An additional 4K RAM is placed next to $D000 - $DFFF as BANK 2.

This high-bank RAM can be freely selected as read only, write only, read and write or inhibited totally. It should be noted that this high-bank RAM occupies the same address range as ROM. Moreover, there are separate high-bank RAM areas for main and auxiliary RAM memory.

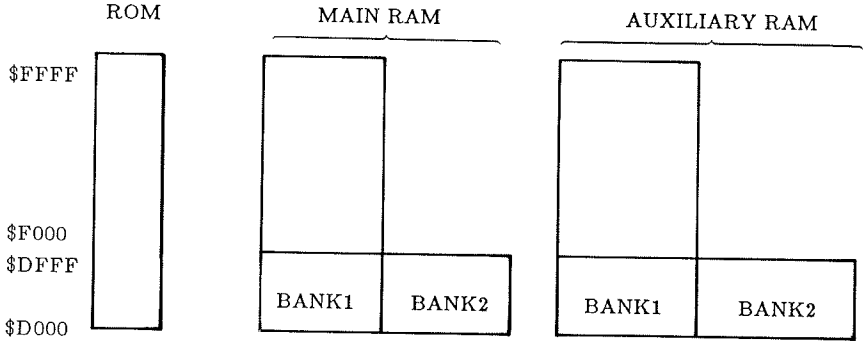


Fig 4.7 High bank RAM memory mapping

4.3.2.1 High - Bank RAM Switches

To control the operation of the high-bank RAM, the switches in TABLE 4.2 can be used. When power is turned on or CTRL-RESET is pressed, the high-bank RAM will always be set to write enabled and ROM set to read enabled.

OPERATION	ADDRESS	FUNCTION
R	C080 (or C084)	Read high-bank RAM only; bank 2
R twice	C081 (or C085)	Read ROM, write high-bank RAM; bank 2
R	C082 (or C086)	Read ROM only
R twice	C083 (or C087)	Read and write high-bank RAM;bank2

R	C088 (or C08C)	Read high-bank RAM only; bank 1
R twice	C089 (or C08D)	Read ROM,write high-bank RAM;bank1
R	C08A (or C08E)	Read ROM only
R twice	C08B (or C08F)	Read and write high-bank RAM;Bank1
W	C008 (AUXZP)	Off AUXZP; use main high-bank,page 0 & page 1
W	C009 (AUXZP)	On AUXZP; use auxiliary high bank,page 0 & page1
R7	C016 (Read AUXZP)	If AUXZP=1 then auxiliary bank RAM is in use If AUXZP=0 then main bank RAM is in use

Table 4.2 RAM control hardware locations for Zero Page, stack page and high bank (N.B. R=Read; W=Write; R7=value of Bit 7 of the I/O location)

How the switches affect the high-bank memory are illustrated in Fig 4.8 to Fig 4.11.

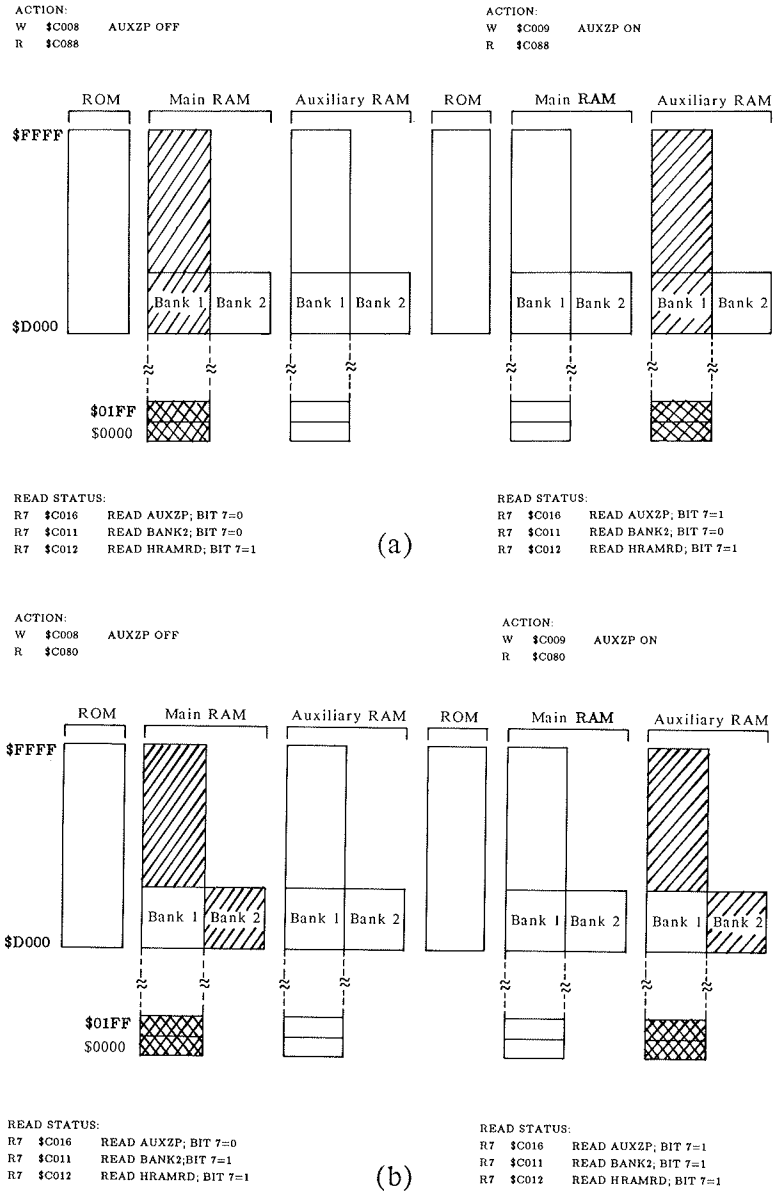
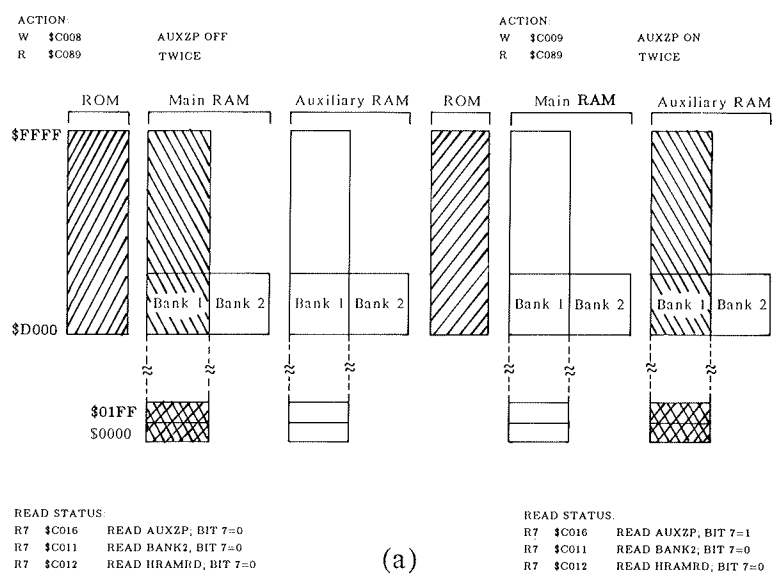


Fig 4.8 High bank and zero page memory.

(a) Read high-bank RAM bank1 memory

(b) Read high-bank RAM bank2 memory



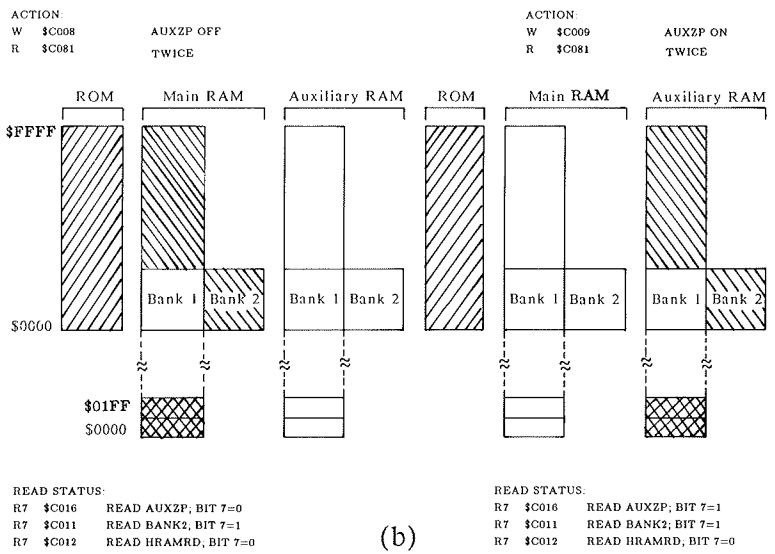


Fig 4.9 High bank and zero page memory.

(a) Read ROM; write high-bank RAM bank1 memory

(b) Read ROM; write high-bank RAM bank2 memory

ACTION:

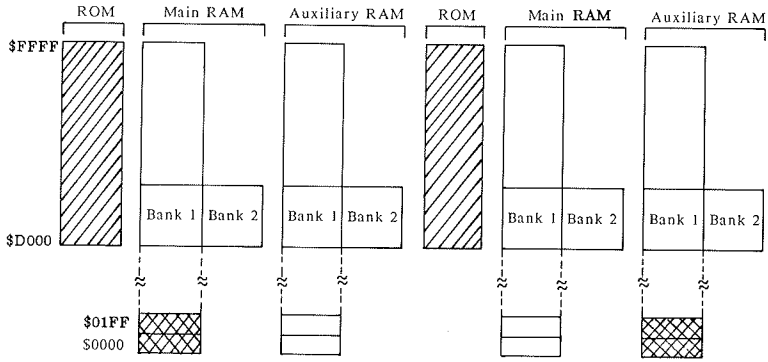
W	$C009	AUXZP OFF
R	$C082	READ ROM, USE BANK 2
R	$C08A	READ ROM, USE BANK 1

OR

ACTION:

W	$C009	AUXZP ON
R	$C082	READ ROM, USE BANK 2
R	$C08A	READ ROM, USE BANK 1

OR



READ STATUS:

R7	$C016	READ AUXZP; BIT 7=0
R7	$C011	READ BANK2; BIT 7=1 OR 0
R7	$C012	READ HRAMRD; BIT 7=0

READ STATUS:

R7	$C016	READ AUXZP; BIT 7=1
R7	$C011	READ BANK2; BIT 7=1 OR 0
R7	$C012	READ HRAMRD; BIT 7=0

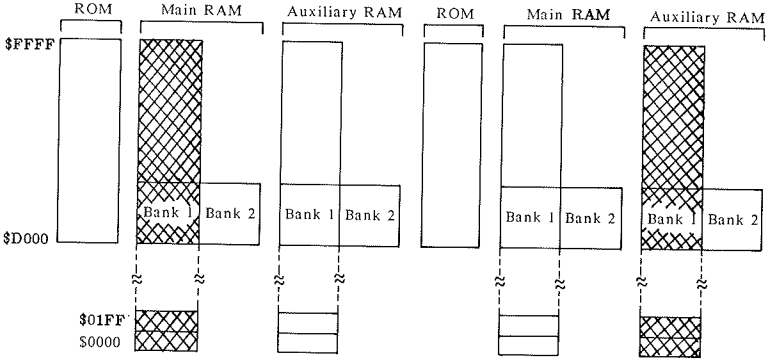
Fig. 4.10 High bank and zero page memory Read ROM only

ACTION:

W	$C008	AUXZP OFF
R	$C08B	TWICE

ACTION:

W	$C009	AUXZP ON
R	$C08B	TWICE



READ STATUS:

R7	$C016	READ AUXZP; BIT 7=0
R7	$C011	READ BANK2; BIT 7=0
R7	$C012	READ HRAMRD; BIT 7=1

(a)

READ STATUS:

R7	$C016	READ AUXZP; BIT 7=1
R7	$C011	READ BANK2; BIT 7=0
R7	$C012	READ HRAMRD; BIT 7=1

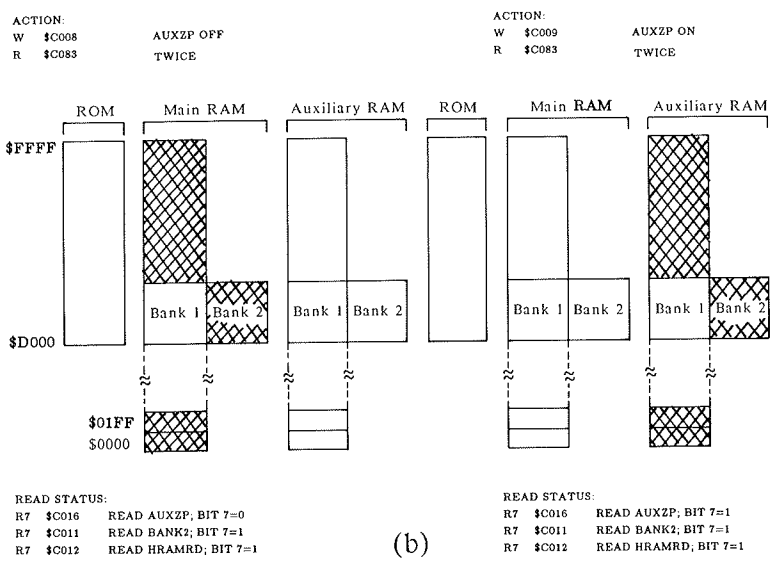


Fig 4.11 High bank memory

(a) Read and write high-bank RAM bank1 memory

(b) Read and write high-bank RAM bank2 memory

Read

Write

4.3.3 Low-bank RAM memory ($0000 - $BFFF)

The low-bank RAM can be divided into 2 parts according to bank control switches:

. Zero page ($0000 - $00FF) and the Stack ($0100 - $01FF) RAM
. RAM addresses from $0200 - $BFFF

4.3.3.1 Zero page and the Stack RAM

This part of RAM can be collectively controlled by the AUXZP switch similar to the high-bank RAM range. When AUXZP soft-switch is turned on, the auxiliary RAM page zero and Stack ($0000 - $01FF) is read and write accessible. If AUXZP is off, the main RAM ($0000 - $01FF) is read and write accessible. The function of the soft-switches can be found in Fig. 4.8 to Fig. 4.11.

⚠ The user is advised to take care in using the zero page and the stack because the 65C02 and system software use this area for storage of important parameters.

4.3.3.2 $0200 - $BFFF RAM memory

The switches which affect the read and write operation on this memory bank are shown in TABLE 4.3. However, the display buffer memory of either graphics or text mode is also within the range of $0200 to $BFFF. The display buffer memory switches have higher priority over memory control switches in TABLE 4.3 , i.e.,when the display buffer memory switches are effective the ARAMRD and ARAMWR switches lose control over the display buffer memory. The display buffer memory will be discussed fully in next section.

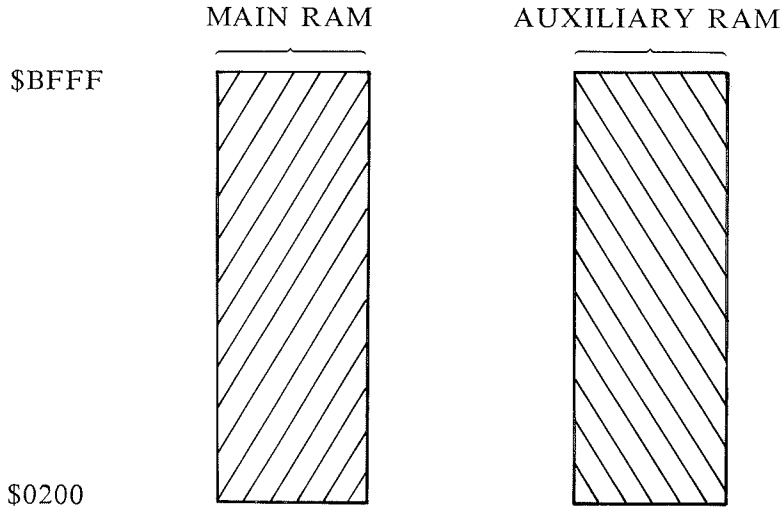
ADDRESS	OPERATION	FUNCTION
$C002	W	off ARAMRD; read main RAM ($0200-$BFFF)
$C003	W	on ARAMRD; read auxiliary RAM ($0200-$BFFF)
$C004	W	off ARAMWR; write main RAM ($0200-$BFFF)

$C005	W	on ARAMWR; write auxiliary RAM ($0200 -$BFFF)
$C013	R7	read ARAMRD status: If bit 7=1; read auxiliary RAM If bit 7=0; read main RAM
$C014	R7	read ARAMWR status: If bit 7=1; write auxiliary RAM If bit 7=0; write main RAM

Table 4.3 $0200 - $BFFF RAM bank switches

The ARAMRD and ARAMWR switches affect the READ / WRITE status of the $0200-$BFFF bank of RAM. These two switches operate independently. E.g. we can set main-RAM-read and auxiliary-RAM-write.

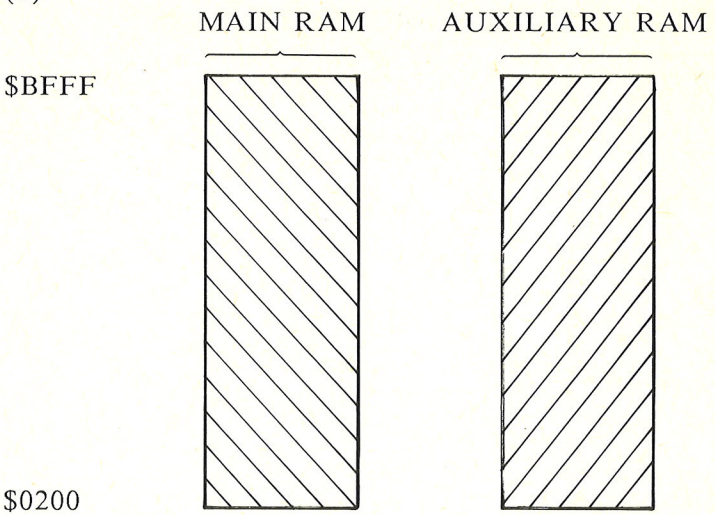
(a)



OPERATION:
W $C000 ; off DOUBLE
W $C002 ; Read main RAM
W $C005 ; write auxiliary RAM

Read Status:
R7 $C018 ; Read DOUBLE bit 7=0
R7 $C013 ; Read ARAMRD bit 7=0
R7 $C014 ; Read ARAMWR bit 7=1

(b)



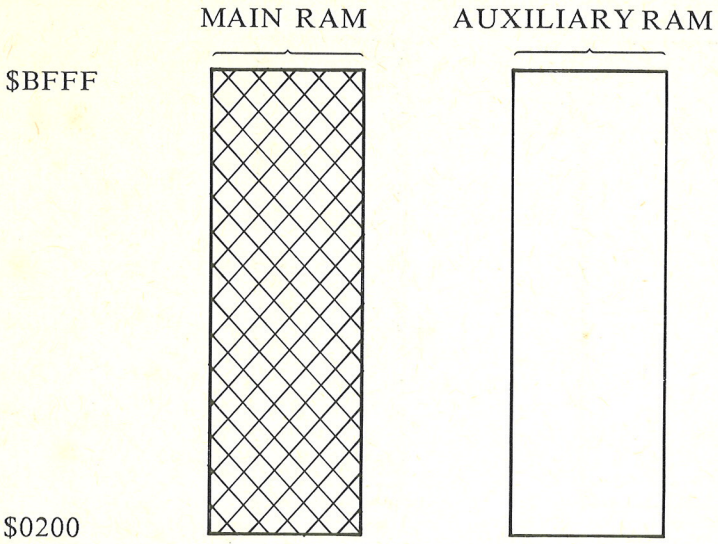
OPERATION:

```
W       $C000 ; DOUBLE off
W       $C003 ; read auxiliary RAM
W       $C004 ; write main RAM
```

Read Status:

```
R7      $C018 ; Read DOUBLE bit 7=0
R7      $C013 ; Read ARAMRD bit 7=1
R7      $C014 ; Read ARAMWR bit 7=0
```

(c)



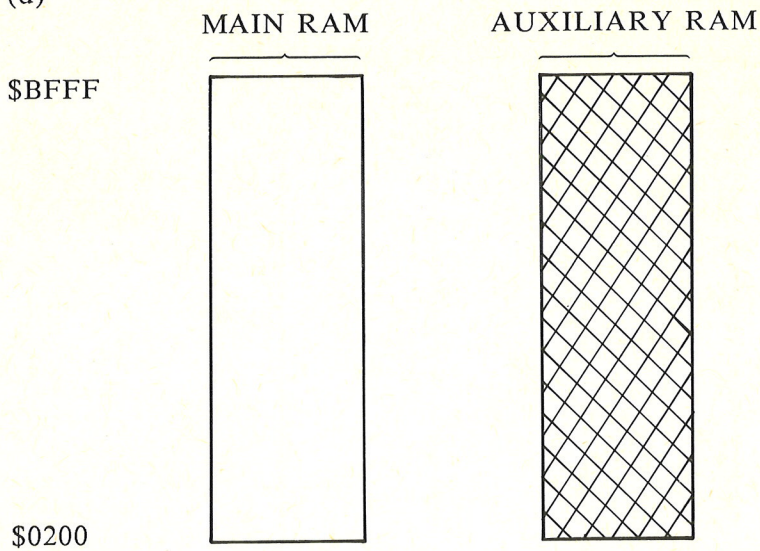
OPERATION:

```
W       $C000 ; DOUBLE off
W       $C002 ; read main RAM
W       $C004 ; write main RAM
```

Read Status:

```
R7      $C018 ; Read DOUBLE bit 7=0
R7      $C013 ; Read ARAMRD bit 7=0
R7      $C014 ; Read ARAMWR bit 7=0
```

(d)



OPERATION:

```
W       $C000 ; DOUBLE off
W       $C003 ; Read auxiliary RAM
W       $C005 ; Write auxiliary RAM
```

Read Status:

```
R7      $C018 ; Read DOUBLE bit 7=0
R7      $C013 ; Read ARAMRD bit 7=1
R7      $C014 ; Read ARAMWR bit 7=1
```

Fig 4.12 $0200-$BFFF Bank RAM

a) Read main RAM, write auxiliary RAM

b) Read auxiliary RAM, write main RAM

c) Read and write main RAM

d) Read and write auxiliary RAM

READ RAM WRITE RAM

4.3.3.3 Display buffer memory

Fig 4.13 illustrates the memory range covered by display buffer memory.

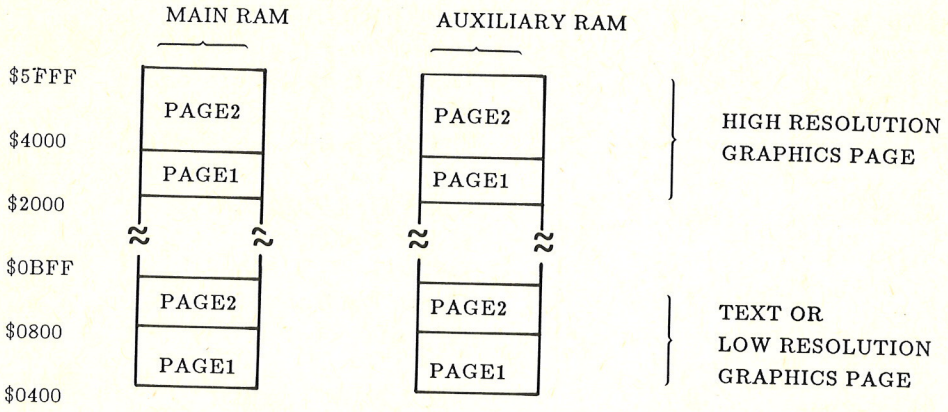


Fig 4.13 Display buffer memory mapping

Since the display buffer memory overlaps with part of the low bank RAM, when the display buffer memory soft-switches are active, ARAMRD and ARAMWR will have no effect on these memory areas.

The switches affecting the display buffer memory are shown in TABLE 4.4.

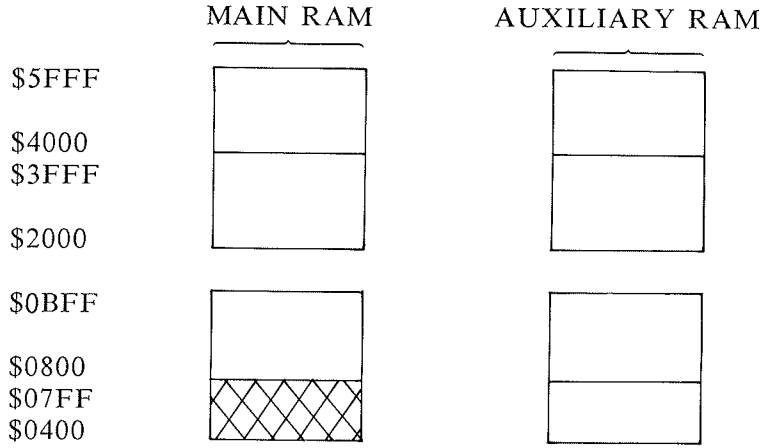
ADDRESS	OPERATION	FUNCTION
$C000	W	off DOUBLE ; ARAMRD and ARAMWR control the display memory
$C001	W	on DOUBLE ; PAGE2 and HGR control the display memory
$C054	R/W	off DP 2; Refer to Fig 4.14
$C055	R/W	on DP2; refer to Fig 4.14
$C056	R/W	off HGR: turns off high resolution graphics; refer to Fig 4.14

$C057	R/W	on HGR; turns on high resolution graphics; refer to Fig 4.14
$C018	R7	Read DOUBLE status If bit 7 = 1 ; DOUBLE on If bit 7 = 0 ; DOUBLE off
$C01C	R7	Read DP2 status: If bit 7 = 1 ; DP2 on If bit 7 = 0 ; DP2 off
$C01D	R7	Read HGR status: If bit 7 = 1 ; HGR on If bit 7 = 0 ; HGR off

Table 4.4 Display buffer memory switches

In Fig 4.14 the operational relation between DOUBLE, DP2 and HGR is shown. It should be noted that when DOUBLE is off, ARAMRD and ARAMWR control the reading or writing of the RAM. The DP2 switch only affects the display. The relation between DP2 and display memory is shown in Fig 4.15.

(a)



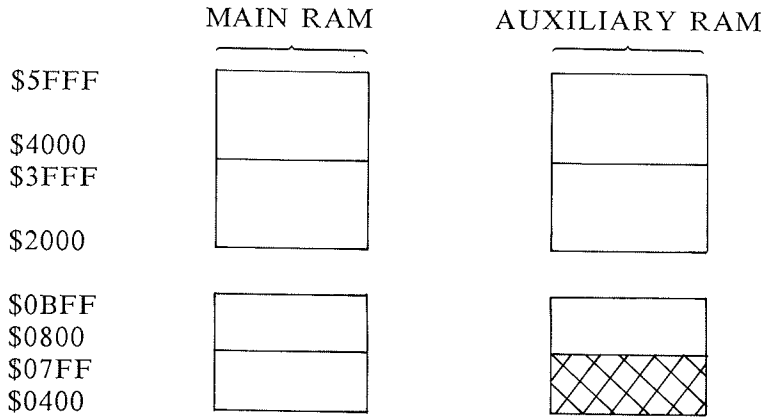
Operation:

W	$C001; on DOUBLE
W	$C054; off DP2
W	$C056; off HGR

Read status:

R7	$C018; Read DOUBLE; bit 7=1
R7	$C01C; Read DP2; bit 7=0
R7	$C01D; Read HGR; bit 7=0

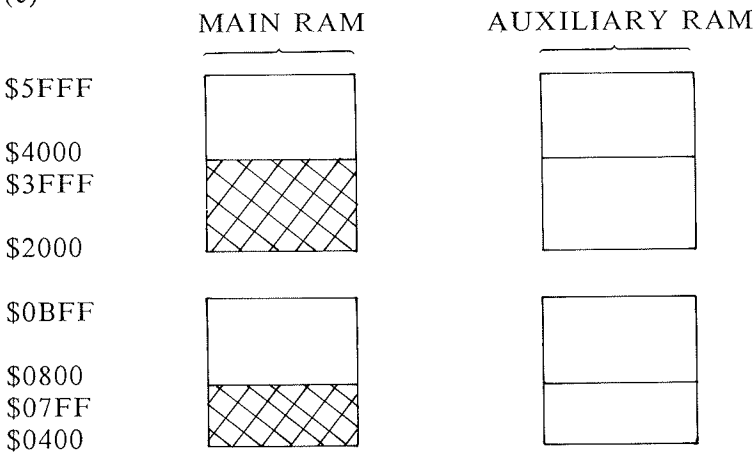
(b)



Operation:
W $C001; on DOUBLE
W $C055; on DP2
W $C056; off HGR

Read status:
R7 $C018; Read DOUBLE; bit 7=1
R7 $C01C; Read DP2; bit 7=1
R7 $C01D; Read HGR; bit 7=0

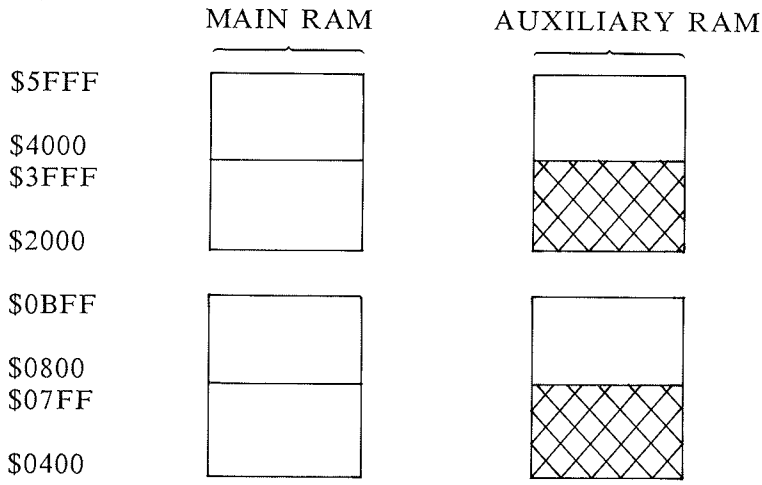
(c)



Operation:
W $C001; on DOUBLE
W $C054; off DP2
W $C057; on HGR

Read status:
R7 $C018; Read DOUBLE; bit 7=1
R7 $C01C; Read DP2; bit 7=0
R7 $C01D; Read HGR; bit 7=1

(d)



Operation:
W $C001; on DOUBLE
W $C055; on DP2
W $C057; on HGR

Read status:
R7 $C018; Read DOUBLE; bit 7=1
R7 $C01C; Read DP2; bit 7=1
R7 $C01D; Read HGR; bit 7=1

RAM read and write

Fig 4.14 The effect of DOUBLE, HGR and DP2 on display RAM area

(a) DOUBLE on HGR off and DP2 off

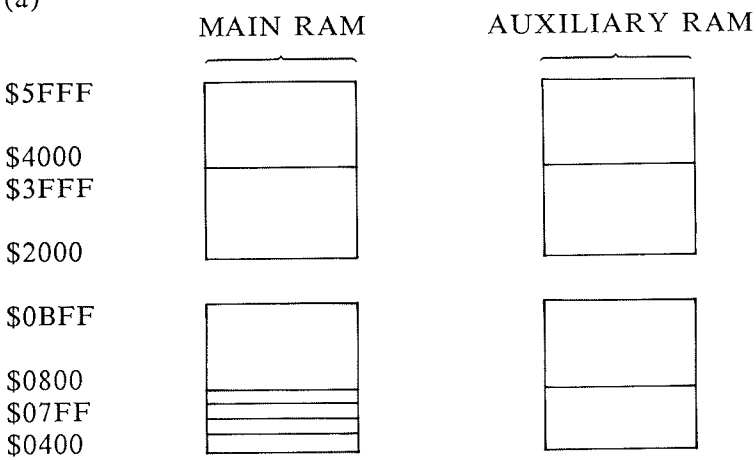
(b) DOUBLE on HGR off and DP2 on

(c) DOUBLE on HGR on and DP2 off

(d) DOUBLE on HGR on and DP2 on

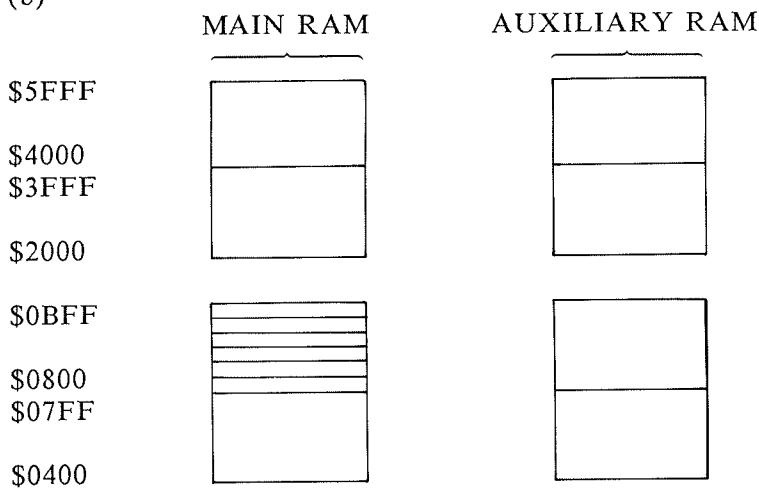
* When DOUBLE is off, ARAMRD and ARAMWR will control which bank of RAM to be read or written.

(a)



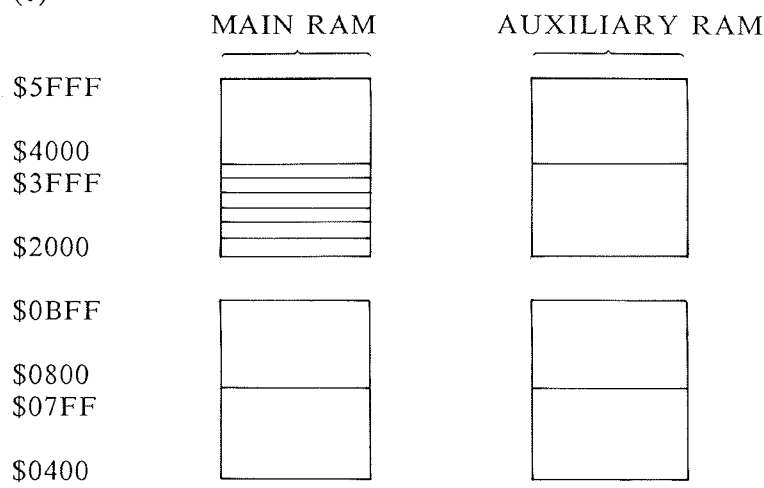
Operation:
W $C054; off DP2
W $C056; off HGR

(b)



Operation:
W $C055; on DP2
W $C056; off HGR

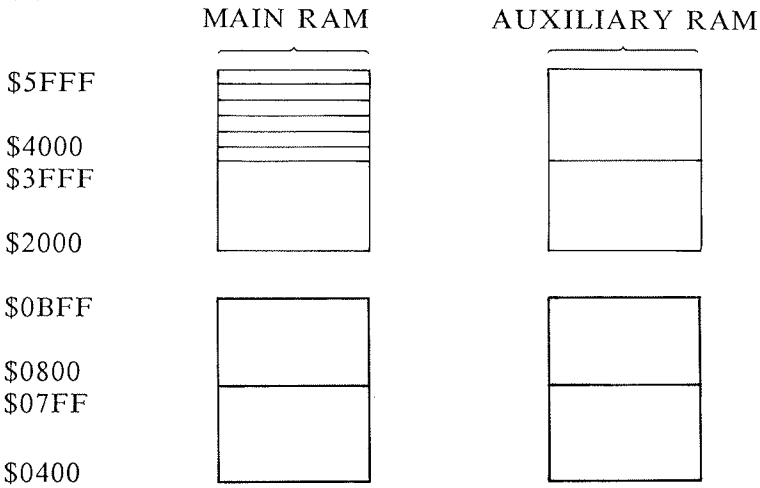
(c)



Operation:

W	$C054; off DP2
W	$C057; on HGR
W	$C050; off TEXT

(d)



Operation:
W $C055; on DP2
W $C057; on HGR
W $C050; off TEXT

memory to be displayed on screen

Fig 4.15 The operation relation of DOUBLE, DP2 and HGR on memory to display

(a) DOUBLE off HGR off and DP2 off

(b) DOUBLE off HGR off and DP2 on

(c) DOUBLE off HGR on and DP2 off

(d) DOUBLE off HGR on and DP2 on

When in double resolution modes, display buffer ($0400-$07FF or $2000-$3FFF) of auxiliary RAM will display simultaneously with display buffer ($0400-$07FF or $2000-$3FFF) of main RAM. The setting of DP2 will lost control in these modes.

4.3.4 RAM Timing

To access RAM, we need the $\overline{RAS}$ and $\overline{CAS}$ signal. Timing diagram is shown in Fig 4.16.

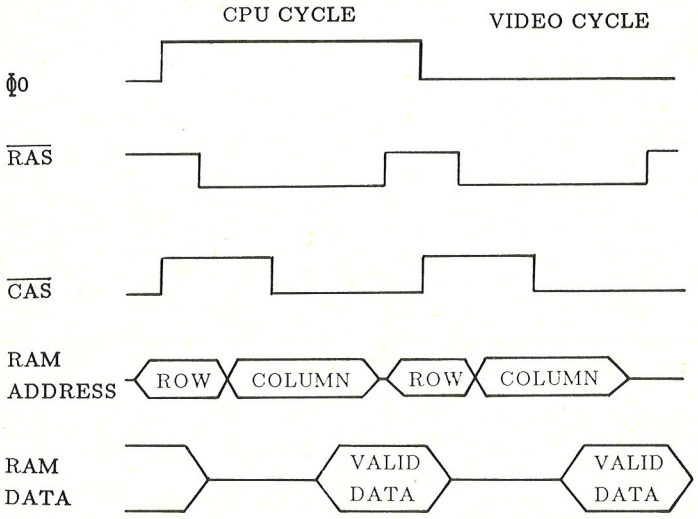


Fig 4.16 Dynamic RAM timing

Since the computer uses dynamic RAM it means we have to "refresh" the content of the RAM within some time limit. With skillful design of video circuitry, the RAM is refreshed within the time limit to retain the content of the RAM.

4.3.5 Schematic of RAM control

As described in previous sections, the RAM memory can be divided into different sub-banks. The control of these sub-banks of RAM memory depends on the address and soft-switches already set. Access of particular bank of RAM can be made by sending an appropriate $\overline{CAS}$ (Column Address Strobe) to that bank of memory. The $\overline{RAS}$ (Row Address Strobe) is always present for all banks. The diagram of Fig 4.17 illustrates the functional block of RAM control.

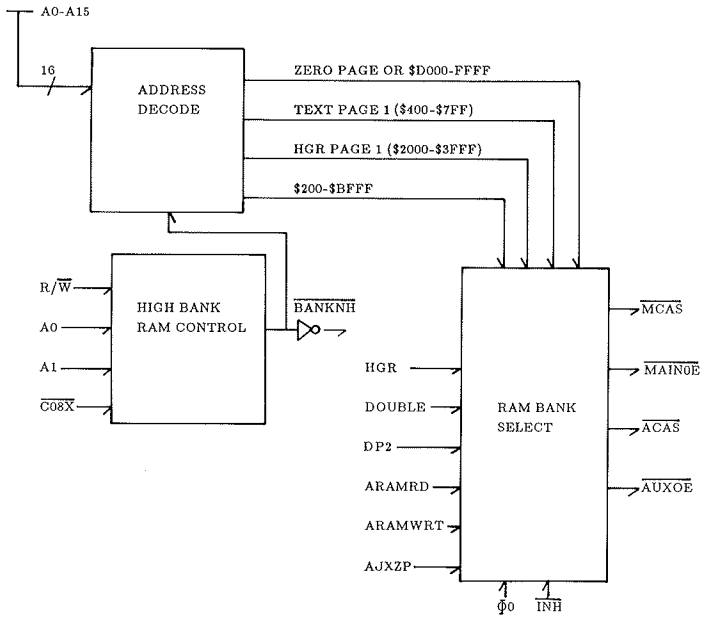


Fig 4.17 Block diagram of RAM select control

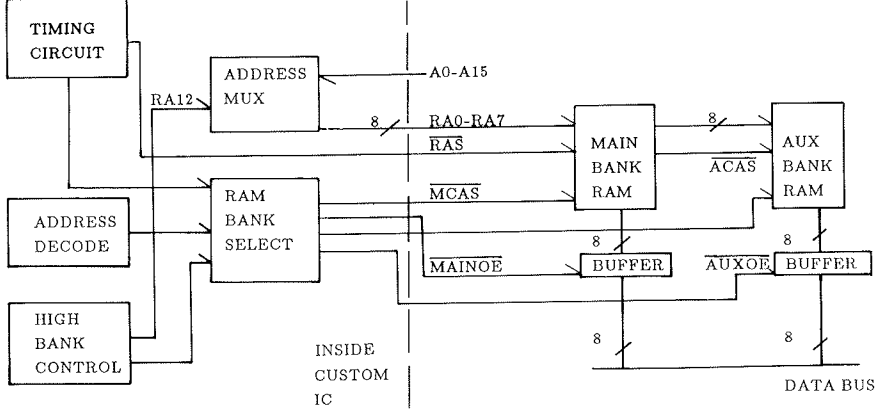


Fig 4.18 Block diagram of RAM memory system

4.4 HARDWARE PAGE ($C000 - C0FF)

There is no physical RAM or ROM in this page. This page is important to the operation of the system because all the soft-switches and their status bits are in this page. The operation of these soft-switches can be found in details in various chapters in this manual.

LOCATION	OPERATION	DESCRIPTION
C00X	R	BIT 7=KEY STROBE; BIT 0-6=KEYBOARD DATA
C000	W	OFF DOUBLE
C001	W	ON DOUBLE
C002	W	OFF ARAMRD
C003	W	ON ARAMRD
C004	W	OFF ARAMWR
C005	W	ON ARAMWR
C006	W	OFF INTIOROM
C007	W	ON INTIOROM
C008	W	OFF AUXZP
C009	W	ON AUXZP
C00A	W	OFF 80ROM
C00B	W	ON 80ROM
C00C	W	OFF TXT80
C00D	W	ON TXT80
C00E	W	OFF CHARSET2
C00F	W	ON CHARSET2
C01X	W	RESET KEYSTROBE

C010	R	BIT 7=1; A KEY IS BEING PRESSED
C011	R	BIT 7=1; BANK2 ON
C012	R	BIT 7=1; HRAMRD ON
C013	R	BIT 7=1; ARAMRD ON
C014	R	BIT 7=1; ARAMWR ON
C015	R	BIT 7=1; INTIOROM ON
C016	R	BIT 7=1; AUXZP ON
C017	R	BIT 7=1; 80ROM ON
C018	R	BIT 7=1; DOUBLE ON
C019	R	BIT 7=1; VERTICAL BACK DROP NOT ACTIVE
C01A	R	BIT 7=1; TEXT ON
C01B	R	BIT 7=1; MIX ON
C01C	R	BIT 7=1; DP2 ON
C01D	R	BIT 7=1; HGR ON
C01E	R	BIT 7=1; CHARSET2 ON
C01F	R	BIT 7=1; TXT80 ON
C02X	R/W	RESERVED
C03X	R/W	TOGGLE SPEAKER OUTPUT
C04X	R/W	RESERVED
C050	R/W	OFF TEXT
C051	R/W	ON TEXT
C052	R/W	OFF MIX
C053	R/W	ON MIX
C054	R/W	OFF DP2
C055	R/W	ON DP2
C056	R/W	OFF HGR
C057	R/W	ON HGR
C058-C05D	R/W	RESERVED
C05E	R/W	ON DOUBLE HIRES
C05F	R/W	OFF DOUBLE HIRES
C06X	W	RESERVED
C060	R	BIT 7=1; 40/80 SWITCH TO 40 POSITION
C061	R	READ SWITCH INPUT 0
C062	R	READ SWITCH INPUT 1
C063	R	READ MOUSE BUTTON
C064	R	READ TIMER 0
C065	R	READ TIMER 1
C066	R	READ MOUSE XDIR
C067	R	READ MOUSE YDIR
C068-C06F	R	RESERVED
C07X	R/W	RESET VERTICAL BACKDROP INTERRRUPT AND JOYSTICK PORT TIMERS
C080-C08F	W	RESERVED
C080	R	READ HIGH BANK2 RAM
C081	R TWICE	READ ROM AND WRITE HIGH BANK2 RAM
C082	R	READ ROM

C083	R TWICE	READ AND WRITE HIGH BANK2 RAM
C084-C087	R	REPEAT C080-C083 FUNCTION
C088	R	READ HIGH BANK1 RAM
C089	R TWICE	READ ROM AND WRITE HIGH BANK1 RAM
C08A	R	READ ROM
C08B	R TWICE	READ AND WRITE HIGH BANK1 RAM
C08C-C08F	R	REPEAT CO88-C08B FUNCTION
C09X	W	*PARALLEL PRINTER STROBE
C090-C097	R/W	RESERVED
C098	R/W	*ACIA1 RECEIVE/TRANSMIT DATA REGISTER
C099	R/W	*ACIA1 STATUS REGISTER
C09A	R/W	*ACIA1 COMMAND REGISTER
C09B	R/W	*ACIA1 CONTROL REGISTER
C09C-C09F	R/W	RESERVED
C0A0-C0A7	R/W	RESERVED
C0A8	R/W	ACIA2 RECEIVE/TRANSMIT DATA REGISTER
C0A9	R/W	ACIA2 STATUS REGISTER
C0AA	R/W	ACIA2 COMMAND REGISTER
C0AB	R/W	ACIA2 CONTROL REGISTER
C0AC-C0AF	R/W	RESERVED
C0BX	R/W	RESERVED
C0C0-C0C7	R	RESERVED
C0C0	W	ON MOUSE X-DIR RISING EDGE INTERRUPT
C0C1	W	ON MOUSE X-DIR FALLING EDGE INTERRRUPT
C0C2	W	ON MOUSE Y-DIR RISING EDGE INTERRUPT
C0C3	W	ON MOUSE Y-DIR FALLING EDGE INTERRUPT
C0C4	W	OFF MOUSE INTERRUPT SOURCE
C0C5	W	ON MOUSE INTERRUPT SOURCE
C0C6	W	OFF VERTICAL BACKDROP INTERRUPT
C0C7	W	ON VERTICAL BACKDROP INTERRUPT
C0C8-C0CE	W	RESERVED
C0C8	R	BIT 7=1/0; MOUSE X-DIR FALLING/RISING EDGE INTERRUPT SELECTED
C0C9	R	BIT 7=1/0; MOUSE Y-DIR FALLING/RISING EDGE INTERRUPT SELECTED
C0CD	R	BIT 7=1; MOUSE INTERRUPT ENABLED
C0CB	R	BIT 7=1; VERTICAL BACKDROP INTERRUPT ENABLED
C0CC	R	BIT 7=1; MOUSE X-DIR INTERRUPT OCCURRED
C0CD	R	BIT 7=1; MOUSE Y-DIR INTERRUPT OCCURRED
C0CE	R	BIT 7=1; VERTICAL BACKDROP INTERRUPT OCCURRED

C0CF	W	RESET MOUSE INTERRUPT
C0DX	R/W	RESERVED
C0EX	R/W	RESERVED FOR FLOPPY DISK DRIVE CONTROL
C0FX	R/W	RESERVED

* At any one time either parallel printer or ACIA1 is active depending on printer select switch setting.

NOTE 1 : Reading $C010 to $C01F will give the keyboard code (bit 0 - bit 6) and reading $C010 will reset KEYSTROBE.

NOTE 2 : $C081, $C083, $C089 and $C08B have to be read twice to achieve the described function in the table.

Table 4.5 Hardware page locations.

4.5 MEMORY EXPANSION

Inside the computer, you can find two rows of jumpers near the RAM chips. They are for future memory expansion. The pin assignment is as follows:

```
16                30
XXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX
1                 15
```

1	A0	20	$\overline{\text{AUXOE}}$-AUX-RAM data buffer enable (from Gate Array)
2	A1 system		
3	A2 address bus		
4	A3	21	$\overline{\text{C07X}}$-active low on ACCESS TO $C07X
5	D1	22	$\overline{\text{AUXOE1}}$-built-in aux-RAM data buffer enable
6	D0 data bus		
7	D2		
8	D3	23	ØO-CPU clock (1MHZ)
9	RA1	24	RA7
10	RA2 RAM address	25	RA5
11	RA0	26	RA4 RAM address
12	$\overline{\text{RAS}}$-RAM $\overline{\text{RAS}}$	27	RA3
13	R/W-read/write	28	RA6
14	$\overline{\text{ACAS}}$-aux RAM $\overline{\text{CAS}}$ (from Gate Array)	29	$\overline{\text{ACAS1}}$-built-in aux-RAM $\overline{\text{CAS}}$
15	GND	30	+5V
16	D7		
17	D6		
18	D5 data bus		
19	D4		

You may find two bow-ties near the expansion jumper. When you want to use the expansion jumper, you need to open the bow-tie. Then the on-board $\overline{\text{ACAS}}$ signal would be directed to the memory expansion board instead of going to the auxiliary 64K RAM on board.

You may then selectively enable the RAM on the expansion board or the auxiliary RAM on board by some logic on the expansion board.

The $\overline{\text{AUXOE}}$ signal from the gate array would enable the data buffer of the auxiliary RAM whenever an access to those RAM occurs. Also, you can use the signal $\overline{\text{C07X}}$ together with the three low order system address lines to select banks on the memory expansion board.

CHAPTER 5

THE VIDEO DISPLAY

The computer is capable of displaying both text and graphics and even a mixture of the two on the same screen. The available modes are:

A) Text Modes
   (a) 40 column x 24 rows.
   (b) 80 column x 24 rows.

B) Graphics modes
   (a) Low resolution (40 X 48)
   (b) Medium resolution (80 X 48)
   (c) High resolution (280 X 192)
   (d) Double-high resolution (560 X 192)

C) Mixed modes
   Mix text and any one of the 4 graphics modes.

The display mode is configured by a number of soft-switches as shown in TABLE 5.1.

ADDRESS	OPERATION	FUNCTION
$C000	W	off DOUBLE
$C001	W	on DOUBLE
$C00C	W	off TXT80
$C00D	W	on TXT80
$C00E	W	off CHARSET 2
$C00F	W	on CHARSET 2
$C050	R/W	off TEXT
$C051	R/W	on TEXT
$C052	R/W	off MIX
$C053	R/W	on MIX
$C054	R/W	off DP2
$C055	R/W	on DP2
$C056	R/W	off HGR
$C057	R/W	on HGR
$C05E	R/W	on DOUBLE HIRES
$C05F	R/W	off DOUBLE HIRES

Table 5.1 Softswitches of Display mode Configuration

5.1 TEXT MODES

Text mode is configured by setting TEXT on.

The text characters that can be displayed include upper and lowercase letters, numerals, punctuation marks and other special characters. Each character occupies a 7 dots wide by 8 dots high matrix on the screen. A character actually occupies a 5 x 7 matrix, leaving one dot column at both sides and one row at the bottom of the character block. The character matrix for the letter H is shown in Fig 5.1.

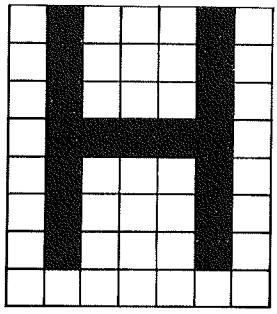


Fig 5.1 Character matrix for letter 'H'

The computer has a primary and an alternate character set which is selected by the soft-switch CHARSET2. With CHARSET2 off the primary set is displayed. The available display formats include: normal (white dots on a black background), inverse (black dots on a white background), flashing (alternating between normal and inverse). Uppercase letters, numerals and special characters can be displayed in all three formats while lowercase letters can only be displayed in normal format.

With CHARSET2 on, the alternate character set is displayed. Flashing format is not available for this mode but lowercase letters can now be displayed in inverse format. Moreover, mouse characters are available. The primary and alternate display character sets are illustrated in TABLE 5.2 (a) TABLE 5.2 (b) respectively.

	INVERSE				FLASHING				NORMAL							
	$Ø0	$1Ø	$2Ø	$3Ø	$4Ø	$5Ø	$6Ø	$7Ø	$8Ø	$9Ø	$AØ	$BØ	$CØ	$DØ	$EØ	$FØ
+$Ø	@	P		Ø	@	P		Ø	@	P		Ø	@	P	`	P
+$1	A	Q	!	I	A	Q	!	I	A	Q	!	I	A	Q	a	q
+$2	B	R	"	2	B	R	"	2	B	R	"	2	B	R	b	r
+$3	C	S	#	3	C	S	#	3	C	S	#	3	C	S	c	s
+$4	D	T	$	4	D	T	$	4	D	T	$	4	D	T	d	t
+$5	E	U	%	5	E	U	%	5	E	U	%	5	E	U	e	u
+$6	F	V	&	6	F	V	&	6	F	V	&	6	F	V	f	v
+$7	G	W	'	7	G	W	'	7	G	W	'	7	G	W	g	w
+$8	H	X	(	8	H	X	(	8	H	X	(	8	H	X	h	x
+$9	I	Y	)	9	I	Y	)	9	I	Y	)	9	I	Y	i	y
+$A	J	Z	*	:	J	Z	*	:	J	Z	*	:	J	Z	j	z
+$B	K	[	+	;	K	[	+	;	K	[	+	;	K	[	k	{
+$C	L	\	,	<	L	\	,	<	L	\	,	<	L	\	l	|
+$D	M	]	–	=	M	]	–	=	M	]	–	=	M	]	m	}
+$E	N	^	.	>	N	^	.	>	N	^	.	>	N	^	n	~
+$F	O	–	/	?	O	–	/	?	O	–	/	?	O	–	o	▒

Table 5.2 (a) Primary Display Character Set

	INVERSE				MOUSE		INVERSE		NORMAL							
	$Ø0	$1Ø	$2Ø	$3Ø	$4Ø	$5Ø	$6Ø	$7Ø	$8Ø	$9Ø	$AØ	$BØ	$CØ	$DØ	$EØ	$FØ
+$Ø	@	P		Ø	▲		`	p	@	P		Ø	@	P	`	P
+$1	A	Q	!	I	△		a	q	A	Q	!	I	A	Q	a	q
+$2	B	R	"	2			b	r	B	R	"	2	B	R	b	r
+$3	C	S	#	3			c	s	C	S	#	3	C	S	c	s
+$4	D	T	$	4			d	t	D	T	$	4	D	T	d	t
+$5	E	U	%	5			e	u	E	U	%	5	E	U	e	u
+$6	F	V	&	6			f	v	F	V	&	6	F	V	f	v
+$7	G	W	'	7			g	w	G	W	'	7	G	W	g	w
+$8	H	X	(	8			h	x	H	X	(	8	H	X	h	x
+$9	I	Y	)	9			i	y	I	Y	)	9	I	Y	i	y
+$A	J	Z	*	:			j	z	J	Z	*	:	J	Z	j	z
+$B	K	[	+	;			k	{	K	[	+	;	K	[	k	{
+$C	L	\	,	<			l	|	L	\	,	<	L	\	l	|
+$D	M	]	–	=			m	}	M	]	–	=	M	]	m	}
+$E	N	^	.	>			n	~	N	^	.	>	N	^	n	~
+$F	O	–	/	?			o	▒	O	–	/	?	O	–	o	▒

Table 5.2 (b) Alternate Display Character Set

40 column text mode is selected by resetting TXT80. In this mode, 40 characters can be displayed on a horizontal line and there are totally 24 horizontal lines. There are two display pages located at $400 - $7FF and $800 - $BFF respectively. The memory mapping scheme is illustrated in Fig 5.2.

PAGE1	PAGE2	$0	$27 $28	$4F $50	$77 $78	$7F
$400	$800	ROW0	ROW8	ROW16	RESERVED	
$480	$880	ROW1	ROW9	ROW17	RESERVED	
$500	$900	ROW2	ROW10	ROW18	RESERVED	
$580	$980	ROW3	ROW11	ROW19	RESERVED	
$600	$A00	ROW4	ROW12	ROW20	RESERVED	
$680	$A80	ROW5	ROW13	ROW21	RESERVED	
$700	$B00	ROW6	ROW14	ROW22	RESERVED	
$780	$B80	ROW7	ROW15	ROW23	RESERVED	

Fig 5.2 40 column text mode memory mapping

Notice that although the 40 bytes of each row are stored in contiguous memory locations, adjacent rows are not. As will be seen later, this scheme can reduce the number of " screen holes", i.e. memory locations inside the display memory which cannot be displayed.

The 80 column text mode is selected by setting TXT80. In this mode, 80 characters can be displayed on a line. The screen may be divided into even columns (0,2,4,.....,78) and add columns (1,3,5,....,79). The memory addresses for the even column bytes and odd column bytes are the same as in 40 column mode. However, even column bytes are addressed from auxiliary memory while odd column bytes are addressed from main memory. For example, the first character of row 0 is located at $400 of auxiliary memory while the second character is located at $400 of main memory. The addressing method for page 2 is the same.

When the soft-switch DOUBLE is turned on, DP2 has another function while its original function is inhibited. As a result, only page 1 can be displayed when DOUBLE is on.

5.2 GRAPHICS MODES

Graphics modes are selected by resetting TEXT and MIX. Individual graphics modes are described in the following sections.

5.2.1 Low resolution graphics (40 X 48)

This is selected by resetting HGR and either resetting TXT80 or turning off DOUBLE HIRES. It shares the same display pages with 40 column text. Each byte controls the color of two rectangular color blocks located one over the other. The size of these two blocks corresponds to the size of a character matrix in 40 column TEXT mode. As a result, there are a total of 48 blocks vertically and 40 blocks horizontally. The upper block is controlled by the low order 4-bit nibble of the byte while the lower block is controlled by the high order 4-bit nibble. The value of a nibble determines the color of block according to TABLE 5.3.

Color Code	Color	Color Code	Color
0	Black	8	Yellowish-Green
1	Dark Red	9	Orange
2	Dark Blue	10	Gray
3	Violet	11	Pink
4	Dark Green	12	Medium Green
5	Gray	13	Yellow
6	Medium Blue	14	Cyan
7	Light Blue	15	White

Table 5.3 Color codes and their associated colors

5.2.2 Medium resolution graphics (80 X 48)

This is selected by resetting HGR, DOUBLE HIRES and setting TXT80. It shares the same display pages with 80 column text and uses the same addressing method, i.e. odd columns are from main memory and even columns are from auxiliary memory. Each byte controls the color of two blocks as in low-res mode. For odd columns, the color codes are the same as that of low-res mode. For even columns, the color code is obtained by rotating the corresponding 4-bit color code for that color in low-res mode one bit to the right. The color codes for the 16 available colors are illustrated in TABLE 5.4.

COLOR CODE	COLOR		COLOR CODE	COLOR	
	EVEN COLUMN	ODD COLUMN		EVEN COLUMN	ODD COLUMN
0	BLACK	BLACK	8	DARK RED	YELLOWISH-GREEN
1	DARK BLUE	DARK RED	9	VIOLET	ORANGE
2	DARK GREEN	DARK BLUE	10	GRAY	GRAY
3	MEDIUM BLUE	VIOLET	11	LIGHT BLUE	PINK
4	YELLOWISH-GREEN	DARK GREEN	12	ORANGE	MEDIUM GREEN
5	GRAY	GRAY	13	PINK	YELLOW
6	MEDIUM GREEN	MEDIUM BLUE	14	YELLOW	CYAN
7	CYAN	LIGHT BLUE	15	WHITE	WHITE

Table 5.4 Medium resolution graphics color codes

5.2.3 High resolution graphics (280 X 192)

This is selected by setting HGR and either resetting TXT80 or turning off DOUBLE HIRES. There are two display pages located at $2000 - $3FFF and $4000 - $5FFF. The memory map is illustrated in Fig 5.3.

The bit-mapped graphics screen is divided into 560 horizontal plotting positions. When the COLOR/MONO switch is thrown to the "MONO" position, the seven LSBs of a byte in the display memory are mapped to seven adjacent dots on the screen, with bit 0 mapped to the leftmost dot. Each dot is either black or white and occupies two plotting positions so that the horizontal resolution is 280 dots. Each line is mapped to 40 adjacent memory locations, with the leftmost seven bits mapped to the first byte of the line buffer. Notice that line buffers for adjacent lines on the screen are not adjacent in memory.

Bit 7 of each byte in the display buffer controls the plotting position of the corresponding video dots. When bit 7 is on, the seven dots are shifted one plotting position to the right on the screen.

Page1	Page2	$0 $27	$28 $4F	$50 $77	$78 $7F	$80 $A7	$A8 $CF	$D0 $F7	$F8 $FF
$2000	$4000	line 0	line 64	line 128	reserved	line 8	line72	line 136	reserved
$2100	$4100	line 16	line 80	line 144	reserved	line 24	line 88	line 152	reserved
$2200	$4200	line 32	line 96	line 160	reserved	line 40	line 104	line 168	reserved
$2300	$4300	line 48	line 112	line 176	reserved	line 56	line 120	line 184	reserved
$2400	$4400	line 1	line 65	line 129	reserved	line 9	line 73	line 137	reserved
$2500	$4500	line 17	line 81	line 145	reserved	line 25	line 89	line 153	reserved
$2600	$4600	line 33	line 97	line 161	reserved	line 41	line 105	line 169	reserved
$2700	$4700	line 49	line 113	line 177	reserved	line 57	line 121	line 185	reserved
$2800	$4800	line 2	line 66	line 130	reserved	line 10	line 74	line 138	reserved
$2900	$4900	line 18	line 82	line 146	reserved	line 26	line 90	line 154	reserved
$2A00	$4A00	line 34	line 98	line 162	reserved	line 42	line 106	line 170	reserved
$2B00	$4B00	line 50	line 114	line 178	reserved	line 58	line 122	line 186	reserved
$2C00	$4C00	line 3	line 67	line 131	reserved	line 11	line 75	line 139	reserved

$2D00	$4D00	line 19	line 83	line 147	reserved	line 27	line 91	line 155	reserved
$2E00	$4E00	line 35	line 99	line 163	reserved	line 43	line 107	line 171	reserved
$2F00	$4F00	line 51	line 115	line 179	reserved	line 59	line 123	line 187	reserved
$3000	$5000	line 4	line 68	line 132	reserved	line 12	line 76	line 140	reserved
$3100	$5100	line 20	line 84	line 148	reserved	line 28	line 92	line 156	reserved
$3200	$5200	line 36	line 100	line 164	reserved	line 44	line 108	line 172	reserved
$3300	$5300	line 52	line 116	line 180	reserved	line 60	line 124	line 188	reserved
$3400	$5400	line 5	line 69	line 133	reserved	line 13	line 77	line 141	reserved
$3500	$5500	line 21	line 85	line 149	reserved	line 29	line 93	line 157	reserved
$3600	$5600	line 37	line 101	line 165	reserved	line 45	line 109	line 173	reserved
$3700	$5700	line 53	line 117	line 181	reserved	line 61	line 125	line 189	reserved
$3800	$5800	line 6	line 70	line 134	reserved	line 14	line 78	line 142	reserved
$3900	$5900	line 22	line 86	line 150	reserved	line 30	line 94	line 158	reserved
$3A00	$5A00	line 38	line 102	line 166	reserved	line 46	line 110	line 174	reserved
$3B00	$5B00	line 54	line 118	line 182	reserved	line 62	line 126	line 190	reserved
$3C00	$5C00	line 7	line 71	line 135	reserved	line 15	line 79	line 143	reserved
$3D00	$5D00	line 23	line 87	line 151	reserved	line 31	line 95	line 159	reserved
$3E00	$5E00	line 39	line 103	line 167	reserved	line 47	line 111	line 175	reserved
$3F00	$5F00	line 55	line 119	line 183	reserved	line 63	line 127	line 191	reserved

Fig 5.3 Hi-res mode 1 memory map

For example, the value $01 at $2000 plots a dot on plotting positions 0 and 1 of line 0 while the value $81 at the same memory location plots a dot on plotting positions 1 and 2.

When the COLOR/MONO switch is thrown to the "COLOR" position, each dot turned on can now be displayed in one of six colors depending on three factors:

1) The corresponding "dot" position on the screen (0 to 279) is even or odd.

2) Bit 7 of the byte to which that bit belong.

3) The states of adjacent bits.

A bit can only be displayed in color if its 2 adjacent bits are turned off. If two or more successive bits are turned on, then they will be displayed in white. Two or more adjacent dots which are turned off give rise to black. The dot position and bit 7 encodes the color of a dot according to TABLE 5.5

Each color dot will be extended by two plotting positions to the right on the screen so that it effectively occupies four plotting positions. The horizontal resolution is thus reduced to 140.

	Even dot	Odd dot
Bit 7 = 0	Violet	Medium Green
Bit 7 = 1	Medium Blue	Orange

Table 5.5 Hi-res mode / color encoding scheme

For example, the pattern $55 at location $2000 gives a short violet line at the top left hand corner of the display window. Notice that since only 7 bits of a byte are displayed, the even bit positions of a byte (namely bit 0,2,4,6) corresponds to even dot positions on the screen if that byte is even and becomes odd dots if that byte is odd. By even and odd bytes, we refer to the number of the byte being displayed (0-39). Software has to take into account this odd / even byte change in order to produce the correct color on the display.

5.2.4 **Double high resolution graphics (560 x 192)**

This is selected by setting HGR, TXT80 and turning on DOUBLE HIRES. Each horizontal line is the bit map of the 7 LSBs of 80 bytes of display memory. Dividing the display into 80 columns of 7 dots wide, each column of a line then corresponds to a single byte.

Grouping the even bytes in one group and odd bytes in another, we get two 280 X 192 displays. The two groups share the same display memory address which is the same as high resolution graphics. However, bytes for the even group are fetched from auxiliary memory while those for the odd group are fetched from main memory. The scheme is illustrated in Fig. 5.4.

A second display page located at $4000 - $5FFF of both main and auxiliary RAM is selected by setting DP2. As usual, setting DOUBLE inhibits DP2 so that only one display page is available in this case.

Dot / Line	Even 0-6	Odd 7-13	Even 14-20	Odd 21-27	Even 28-34	Odd 35-41	Even 42-48
0	$2000	$2000	$2001	$2001	$2002	$2002	$2003
1	$2400	$2400	$2401	$2401	$2402	$2403	
2	$2800	$2800	$2801	$2801	$2802		
3	$2C00	$2C00	$2C01	$2C02			
4	$3000	$3000	$3001				
5	$3400	$3400					
6	$3800						

Fig 5.4 Double-high resolution graphics memory scheme

With the COLOR / MONO switch set to MONO position, the horizontal resolution is 560 dots if viewed on a high resolution monochrome monitor, with each dot occupying one dot position. Bit 7 of the display memory bytes is not used.

With the switch set to COLOR position, the adjacent 4 dots are grouped together to form a single colored pixel and effective horizontal resolution is reduced to 140 pixels.

*Remainder / Pattern	0	1	2	3
1000	Dark Blue	Dark Green	Yellowish-Green	Dark Red
1001	Violet	Medium Blue	Medium Green	Orange
1010	Gray	Gray	Gray	Gray
1011	Pink	Light Blue	Cyan	Yellow
1100	Medium Blue	Medium Green	Orange	Violet
1101	Light Blue	Cyan	Yellow	Pink
1110	Cyan	Yellow	Pink	Light Blue
1111	White	White	White	White

* - remainder left when plotting position of leftmost bit of the 4-bit pattern is divided by four.

Table 5.6 Double-high resolution graphics color encoding scheme

When the switch is thrown to the "COLOR" position, each dot (one or more adjacent bits which are turned on) can now be displayed in one of sixteen colors depending on its plotting position and the

dot at the right, if any. The color encoding scheme is tabulated in TABLE 5.6.

5.3 MIX TEXT and GRAPHICS MODES

Mix mode is available for all the four graphics modes. This is selected by resetting TEXT and setting MIX. When active, four lines of text are displayed at the bottom of the screen, while the remaining area displays graphics. The four lines of text correspond to the bottom four lines of the active text page as selected by DP2. For low-res, hi-res, either 40 or 80 column text can be displayed by resetting or setting TXT80. Note that for the latter case, this means DOUBLE HIRES must be off. For med-res and double hi-res only 80 column text is available since TXT80 must be set for these modes. Mix mode display is illustrated in Fig 5.5.

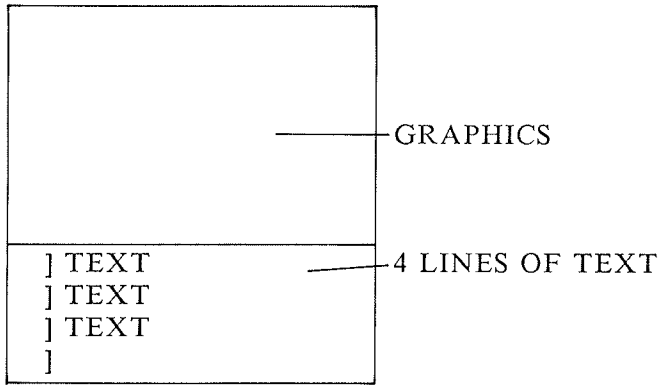


Fig. 5.5 Mix mode display

5.4 VIDEO DISPLAY GENERATOR (VDG) HARDWARE

The Video Display Generator may be logically divided into 7 sections:

- Video Counters
- Blanking, Sync and Burst Gate Generator
- Video Address Mappter and Dynamic RAM address multiplexer
- Video Data generator

- Color Encoder
- Chroma Generator
- Video Summer

Fig 5.6 shows the inter-relationship of the various logic building blocks comprising the video display generator.

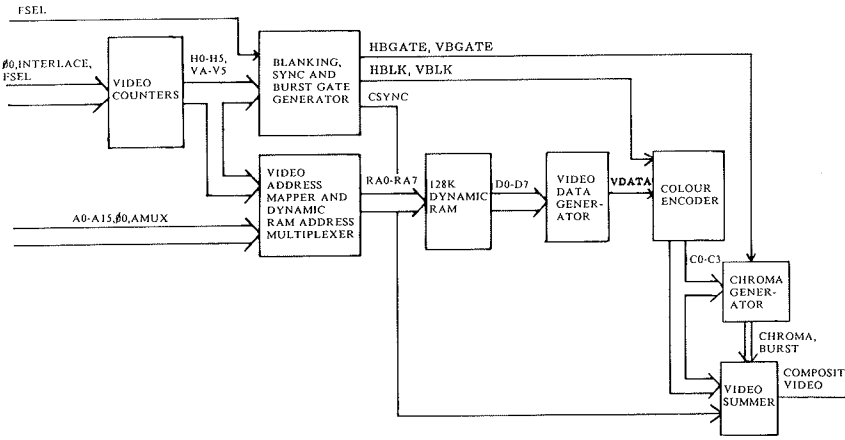


Fig 5.6 Block diagram of video display generator

5.4.1 Video Counters

The display synchronisation signals and video RAM addresses are all derived from the outputs of a 16-bit counter chain which is clocked at the rising edge of Φ0.

a) Horizontal Counter

It consists of the 6 low order bits of the counter chain, namely H0, H1, H2, H3, H4, H5. For NTSC system, the counter counts simply from 0 to 63 and starts again at 0. Hence, there are totally 64 horizontal counts.

For PAL system, count 0 is lengthened by one Φ0 cycle and hence extending total number of counts to 65.

b) Vertical Counter

It consists of the 10 high order bits of the counter chain, namely VA,VB,VC,V0,V1,V2,V3,V4,V5 and V6. It is activated only when

H0 to H5 are all equal to 1, which signals the end of a horizontal line. When VA to V5 counts up to 511, i.e. the end of a field, the vertical counter resumes counting at a starting count which depends on the TV standard adopted. The starting count for VA to V5 is shown in Table 5.7.

	VERTICAL START COUNT
NTSC	246
PAL	200

Table 5.7 Starting counts for VA-V5

A field contains an even number of lines, namely 266 lines for NTSC and 312 lines for PAL.

5.4.2 Blanking, Sync and Burst Gate Generator

This section of the VDG generates 3 types of control signals from the Video Counter outputs.

(a) Synchronization Pulses

(i) Horizontal Sync (HSYNC)

This triggers the horizontal retrace of the electron beam at the end of a line. (Fig 5.7)

(ii) Vertical Sync (VSYNC)

This triggers the vertical retrace of the electron beam at the end of a field. (Fig 5.8)

(iii) Composite Sync (CSYNC)

This combines HSYNC, VSYNC, equalizing pulses and vertical serrations into a single sync signal for use in composite video output. Equalizing pulses are added at both sides of the vertical sync pulse to improve vertical synchronization while vertical serrations are added to maintain horizontal synchronization during vertical sync interval. (Fig 5.9)

(b) Blanking Pulses

(i) Horizontal Blanking (HBLK)

This blanks out the electron beam during horizontal retrace. (Fig 5.7)

(ii) Vertical Blanking (VBLK)

This blanks out the election beam during vertical retrace. (Fig 5.8)

(c) Color Burst Gating Pulses

(i) Horizontal Burst Gate (HBGATE)

At the end of a horizontal line, right after the horizontal sync pulse, a short burst of color subcarrier frequency is transmitted to synchronize the internal oscillator of the TV set. It is the phase of the subsequent video data relative to this color burst that determines the color of a particular point on the display. The color burst is gated out when this signal is high. (Fig 5.7)

(ii) Vertical Burst Gate (VBGATE)

The color burst is not transmitted near the vertical sync interval to prevent it from interfering with vertical synchronization. When this signal is low, the burst is inhibited. (Fig 5.8)

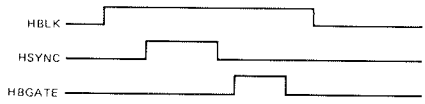


Fig. 5.7 Horizontal Display Timing

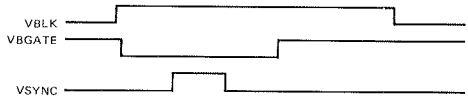


Fig. 5.8 Vertical Display Timing

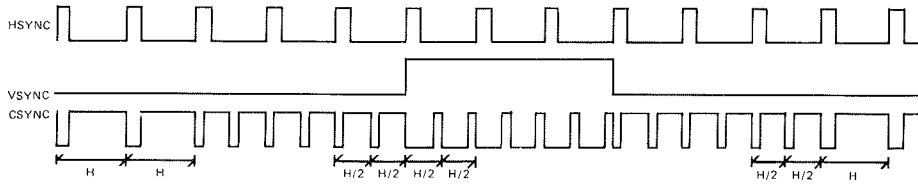


Fig. 5.9 Composite Sync pulse with serrations and equalizing pulses

5.4.3 Video Address Mapper and Dynamic RAM address multiplexer

Of the 64 (NTSC) or 65 (PAL) horizontal counts, only the 40 bytes addressed from count 24 to 63 are displayed. Count 0 to 23 are used for blanking, sync and color burst gate. Similarly, only the 192 lines addressed from count 0 to 191 of the 266 vertical counts in NTSC (312 vertical counts in PAL) are displayed. Using the horizontal and vertical counter bits directly as RAM address will leave "holes" in the display memory, these non-contiguous memory fragments cannot be used easily. As a result, to minimize the size of these "holes", the video addresses are mapped in such a way that adjacent rows are not stored in contiguous memory locations.

H0, H1, H2 are used as the low order 3 bits of the video address (VA0-VA2) so that each line is divided into 5 groups of 8 bytes each. The next 4 bits of the video address are formed from H3,H4,H5,V3,V4 as shown in Fig 5.10.

V4	V3	H5	H4	H3	VA6	VA5	VA4	VA3	V4	V3	H5	H4	H3	VA6	VA5	VA4	VA3
0	0	0	1	1	0	0	0	0	0	1	0	1	1	0	1	0	1
0	0	1	0	0	0	0	0	1	0	1	1	0	0	0	1	1	0
0	0	1	0	1	0	0	1	0	0	1	1	0	1	0	1	1	1
0	0	1	1	0	0	0	1	1	0	1	1	1	0	1	0	0	0
0	0	1	1	1	0	1	0	0	0	1	1	1	1	1	0	0	1
1	0	0	1	1	1	0	1	0									
1	0	1	0	0	1	0	1	1									
1	0	1	0	1	1	1	0	0									
1	0	1	1	0	1	1	0	1									
1	0	1	1	1	1	1	1	0									

Fig 5.10 Mapping for VA3 - VA6

V0, V1, V2 are used directly as VA7, VA8, VA9 respectively.

V4 and V3 divide the displayable portion of the screen into 3 groups while V2,V1 and V0 divide each group into 8 rows. Thus in text mode we have 24 rows totally. V2,V1 and V0 address 8 contiguous memory segments of 128 bytes each. 3 rows spaced 8 rows apart on the screen are grouped together and stored in the first 120 locations of a 128 byte segment, leaving 8 unused locations which we refer to as "holes". Hence a 40 column text page occupies 8 x 128 or 1024 bytes only.

Consider the uppermost group with V4V3 = 00. Observant readers will notice the following relationship: VA6 VA5 VA4 VA3 = $\overline{H5}$ $\overline{H5}$ H4 H3 + 0001

For the second group with V4V3 = 01, VA6 VA5 VA4 VA3 is just the corresponding value for the first group plus 0101. Similarly, for the third group, it is just the corresponding value for the first group plus 1010. Hence, the mapping may in fact be realized by an adding process:

VA6 VA5 VA4 VA3 = H5 H5 H4 H3 + 0001 + V4V3V4V3

VA10 - VA15 depends on the display mode and is shown in Fig. 5.11.

Video Address	Text/Lo-res/ med-res	Hi-res/Double Hi-res
VA10	$\overline{\text{DP2.DOUBLE}}$	VA
VA11	DP2.$\overline{\text{DOUBLE}}$	VB
VA12	0	VC
VA13	0	$\overline{\text{DP2.DOUBLE}}$
VA14	0	DP2.$\overline{\text{DOUBLE}}$
VA15	0	0

Fig 5.11 Mapping for VA10-VA15

Notice that DOUBLE inhibits DP2.
For the high resolution graphics modes, VC, VB and VA, which identify a line within a row, address 8 continuous memory segments of 1024 bytes each. A high resolution graphics page thus occupies 8192 locations.

The video address and CPU address are multiplexed into a 8-bit dynamic RAM address using Φ0 and AX.

For the timing diagram of Φ0 and AX, refer to Fig. 5.14.

Notice that for the video row address, RA0 - RA6 switches through all 128 combinations in one RA6 cycle and RA0 - RA7 switches through all 256 combinations in one RA7 cycle.

RA6 = VA8 = V1 or V0
RA7 = VA9 = V2 or V1
Period of V2 = 4 ms
Period of V1 = 2 ms

Hence the video row address automatically refreshes the dynamic RAM which requires 128 refresh cycles in 2ms or 256 refresh cycles in 4 ms.

	ϕ0	AX	RA0	RA1	RA2	RA3	RA4	RA5	RA6	RA7
1	0	0	VA4	VA5	VA6	VA10	VA11	VA13	VA14	VA15
2	0	1	VA0	VA1	VA2	VA3	VA12	VA7	VA8	VA9
3	1	0	A4	A5	A6	A10	A11	A13	A14	A15
4	1	1	A0	A1	A2	A3	RA12	A7	A8	A9

where 1 is Video Column Address
2 is Video Row Address (Refresh Address)
3 is CPU Column Address
4 is CPU Row Address

Fig 5.12 Dynamic RAM address multiplexing

5.4.4 Video Data Generator

Fig 5.13 is a block diagram of the Video Data Generator.

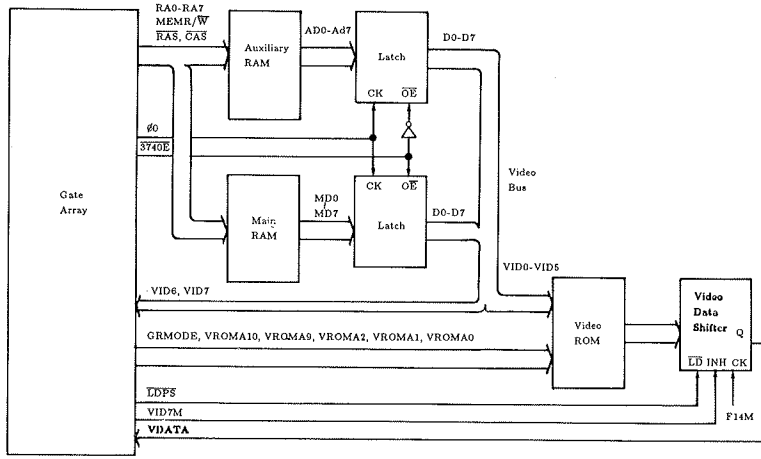


Fig 5.13 Blocking diagram of the Video Data Generator

The video address is sent to the main and auxiliary RAM bank at $\Phi 1$ high through RA0 to RA7 and strobed into the RAM by $\overline{CAS}$. MEMRW is high during $\Phi 1$ and hence data are being driven out of the RAM chips and are latched up at the next $\Phi 0$ rising edge. In 40 column mode, the main video latch drives the video bus at $\Phi 0$ while in 80 column, it drives the bus at $\Phi 1$. The auxiliary latch drives the video bus at $\Phi 1$ in 40 column mode and $\Phi 0$ in 80 column mode. High order two bits (VID7, VID6) are routed back to the Gate Array while the remaining six bits (VID0 - VID5), together with several outputs from the GA, forms the address input of the video ROM. The video ROM performs mapping according to the display mode and the latched data. Then, the video data is transferred to the Gate array.

The video ROM is divided into two 2K banks using the signal GRMODE which connects to A11. The upper 2K, identified by GRMODE = 1, contains the graphic mode video data while the lower 2K bank contains the character generator for text mode. VID0 - VID5 connected to VROMA10, VROMA9, VROMA2, VROMA1, VROMA0 from the GA. The GA transmits different signal to these address inputs of the ROM depending on the display mode:

(a) Text mode (GRMODE = 0)

A character on the screen is made up of a 7 rows by 5 columns dot matrix. However, each character displayed has to be separated from the adjacent one by spaces. Therefore, one row of space is added at the bottom and one column space is added at both sides of a character matrix. The actual size becomes 8 rows by 7 columns. In the character generator, each byte stores the dot pattern of one character line. Only 7 out of 8 bits are used. Eight contiguous bytes store a complete character pattern with each byte corresponding to one character line.

In text mode, VID0 - VID7 are in fact the ASCII code of the character to be displayed. VROMA10, VROMA9 together with VID0 - VID5 index directly into a table of character patterns in the Video ROM. VC,VB,VA are transmitted through VROMA2, VROMA1, VROMA0 respectively to address the dot pattern for the particular line of the character to displayed.

VID7, VID6 and the soft switch ALTCHARSET controls the character display mode through VROMA10 and VROMA9 as shown

in Fig 5.15 VROMA10 and VROMA9 divides the character generator into 4 sections as shown in Fig 5.16.

Note that flashing character may be obtained by alternately displaying contents of normal character generator and inverse character generator.

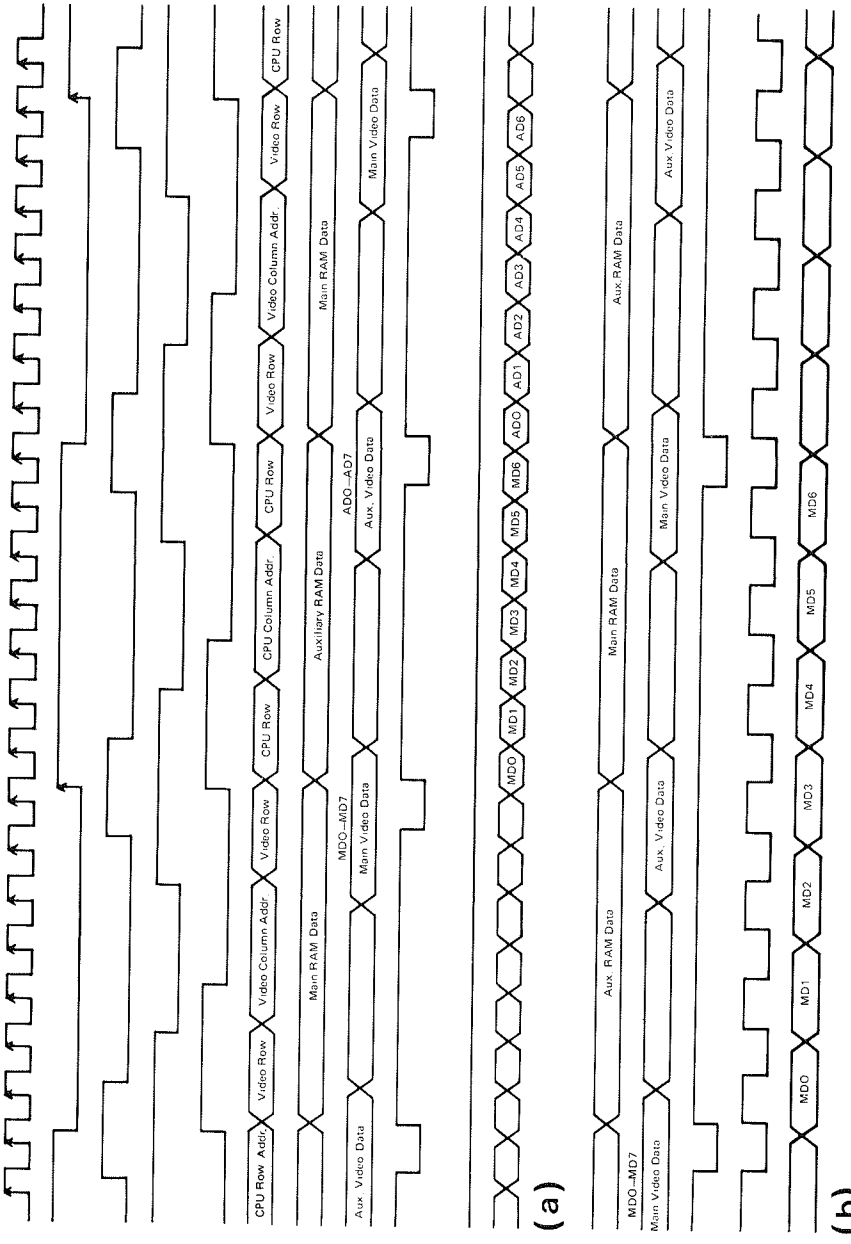


Fig 5.14 Video data generation timing: (a) 80 column mode, (b) 40 column mode

VID7	VID6	ALTCHARSET	VROMA10	VROMA9	DISPLAY MODE
0	0	0	0	0	INVERSE
0	0	1	0	0	INVERSE
0	1	0	*	0	FLASHING
0	1	1	0	1	INVERSE
1	0	0	1	0	NORMAL
1	0	1	1	0	NORMAL
1	1	0	1	1	NORMAL
1	1	1	1	1	NORMAL

*alternates between 0 and 1 at 1/32 field rate.

Fig 5.15 VROMA10, VROMA9 in text mode.

		VROMA10	VROMA9
$700-$7FF	NORMAL LOWERCASE CHARACTERS	1	1
$600-$6FF	NORMAL UPPERCASE CHARACTERS		
$500-$5FF	NORMAL NUMERALS AND SPECIAL CHARACTERS	1	0
$400-$4FF	NORMAL UPPERCASE CHARACTERS		
$300-$3FF	INVERSE LOWERCASE CHARACTERS	0	1
$200-$2FF	MOUSE CHARACTERS		
$100-$1FF	INVERSE NUMERALS AND SPECIAL CHARACTERS	0	0
$000-$0FF	INVERSE UPPERCASE CHARACTERS		

Fig 5.16 Character Generator Mapping.

(b) Graphics mode (GRMODE = 1)

VID7, VID6 are transmitted through VROMA10 and VROMA9 to the video ROM so that in effect all eight latched data bits are passed to it. Handling of VID0 - VID7 are different for low-res and Hi-res modes. Hence, the signal HGR is transmitted to the video ROM through VROMA1. This divides the 2K graphic data mapper into 2 banks. Notice that data of each bank does not occupy contiguous memory locations. For the bank corresponding to Hi-res mode, it is just a straight-through bit-map such that the video ROM outputs are VID0 - VID7.

For Low-res mode, VC, which toggles every four lines on the screen, are transmitted through VROMA2 to the Video ROM. This identifies whether the upper or lower 4-bit nibble of the latched data is to be used. H0, which toggles every horizontal count, is transmitted through VROMA0 to the ROM. The 4-bit nibble selected will be rotated two bits to the right if the pixel is to be displayed on an odd column on the screen (H0 = 1) so that the color encoder can generate a correct color code for the pixel. The final 4-bit pattern is repeated on both the upper and lower nibble of the video ROM output. For example, a pattern of 10011101 will generate 01100110 as the ROM output when VC = 1 and H0 = 1 and 11011101 when VC = 0 and H0 = 0.

5.4.5 Color Encoder

This encoder will receive the video data and then convert to a 4-bit color code (C0-C3). The value of color code depends on the bit position on a horizontal line as well as on the bit pattern fetched from the RAM. The 4-bit code will then feed to the chroma generator.

5.4.6 Chroma Generator

The chroma generator receives the 4-bit color code C0 - C3 from the color encoder and generates the chrominance signal, which has a definite phase relationship with the color burst.
The chroma generator divides the signal XTAL by 4 to obtain square waves of frequency equal to that of the color subcarrier for the particular TV system (NTSC or PAL). Eight phases differing by 45° from each other are generated as shown in Fig 5.17. These are selected by C0 - C3 as the chroma output.

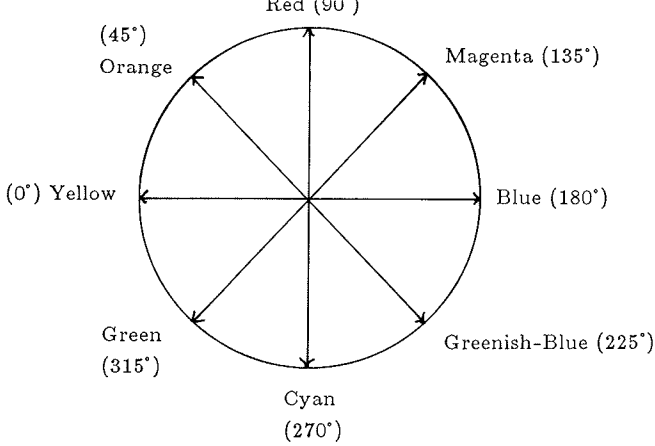


Fig 5.17 NTSC color circle

For the PAL system, the R-Y component reverses every line so that for one line, the color circle coincides with that of NTSC (NTSC line) while for the next line, the direction of the color circle is reversed (PAL line). For example, red is 90° (clockwise) for NTSC line, 90° (anti-clockwise) for PAL line. Color burst for NTSC system is located at 0° , i.e. yellow, while that for PAL system alternates between 45° clockwise and 45° anti-clockwise for NTSC line and PAL line respectively.

In order to have a stable display, each field must contain an integral number of color subcarrier cycles so that the chrominance signal has a constant phase relationship with the video data.

For the NTSC system, F14M = XTAL x 56/57 = 14.06698 MHz. Since there are 64 Φ0 cycles per line and 266 lines per field, therefore each field contains 266 x 64 x 14 x 57/56 x XTAL x 1/4 or 60648 x 3.58M cycles thus meeting the requirement. For the PAL system, due to the phase alternate property, it is also required that the number of lines per field be even. As a result, the chrominance signal cannot be synchronised with the video data unless each line contains an intergal number of color subcarrier cycles. Each field contains 312 lines, the second requirement is automatically satisified. To meet the first requirement, F14M is chosen to have the following relationship with XTAL:

F14M=XTAl x 4/5=17.73447MHz x 4/5=14.187577MHz

Each field thus contains $\frac{14x65x312x5/4}{4}$ or 88725

subcarrier cycles of 4.43MHz.

For the PAL version of the computer, there is a 50Hz/60Hz switch underneath the unit. The switch position is set in factory. However, if the LCD panel from Apple® is connected to the video expansion connector, the switch has to be set to the 60Hz position for proper functioning of the LCD panel.

5.4.7 Video Summer

The video summer combines the luminance signal which controls the intensity of the display, the chrominance signal which carries the color information, the composite sync, and the color burst into the composite video output.

Luminance is formed by summing C0-C3. The weighing of the 4 bits are the same so that only 4 levels of luminance variations are available.

Chroma and burst are capacitively coupled to the summer so that they will be centered at the luminance level and blanking level respectively and hence will not affect the average d.c. level of the latter.

The composite video output is a 1.2V p-p signal for use with video monitors.

CHAPTER 6

THE INPUT/OUTPUT PORT

The computer has nearly all useful I/O peripherals built-in. The customer would appreciate this benefit as he gets used to the computer. This chapter will help the user to exploit this outstanding feature of the computer. The I/O subsystem can be subdivided into

. keyboard
. sound
. game port (joystick / mouse)
. parallel printer port
. serial port 1-serial printer
. serial port 2-modem
. FDD (floppy disk drive)
. expansion slot

6.1 THE KEYBOARD

The keyboard of the computer has a typewriter layout, a numeric keypad and ten function keys. The specifications of the keyboard are listed in Table 6.1.

Table 6.1

Number of keys	:	90
Encoding format of character	:	ASCII
Special keys	:	**10 function keys,** [▲] **and** [△] [CONTROL] - [RESET]
Cursor keys	:	[←] [→] [↑] [↓]
Features	:	Auto-repeat

The computer keyboard layout can be changed by the keyboard switch on the back panel. The layout of the NTSC unit are the USA standard (Sholes) or the simplified (Dvorak). For other versions of the computer, their keyboard layout can be found in APPENDIX D.

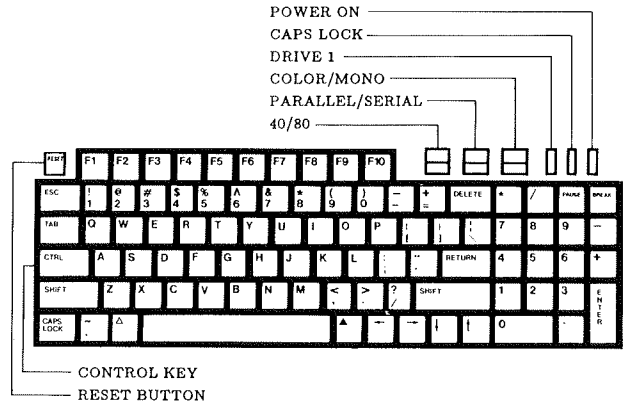


Fig 6.1 The computer keyboard switches and indication lights

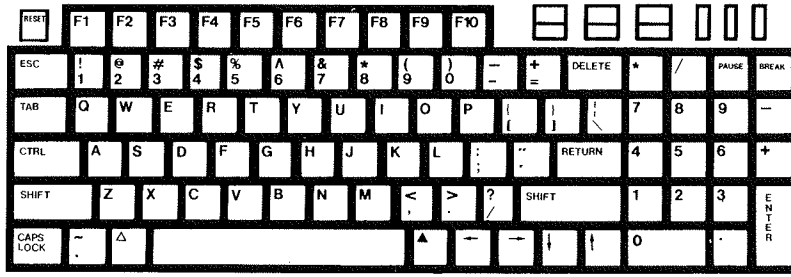


Fig 6.2 a) keyboard switch set to STD (The USA standard keyboard)

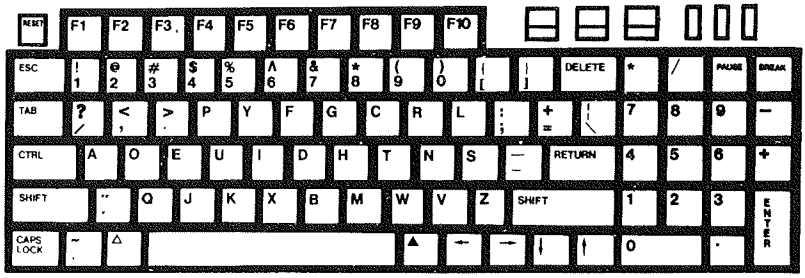


Fig 6.2 b) Keyboard switch is set to ALT (The simplified keyboard)

There are 3 indicator lights on the front panel. They are

a) POWER - when power is turned on, the light will be on.

b) DISK - when the built-in drive is accessed, this light will glow.

c) CAPSLOCK - when this light grows it indicates that the keyboard is in capslock mode, that is all letter keys will produce capital letters on the screen irrespective of the position of the shift keys.

The computer keyboard will generate ASCII code when any one key of the keyboard is pressed. The keys to generate ASCII codes are tabulated in Table 6.2.

Table 6.2 keys & the ASCII codes

KEY	NORMAL	CHAR	CONTROL	CHAR	SHIFT	CHAR	BOTH	CHAR
DELETE	7F	DEL	7F	DEL	7F	DEL	7F	DEL
←	08	BS	08	BS	08	BS	08	BS
TAB	09	HT	09	HT	09	HT	09	HT
↓	0A	LF	0A	LF	0A	LF	0A	LF
↑	0B	VT	0B	VT	0B	VT	0B	VT
RETURN	0D	CR	0D	CR	0D	CR	0D	CR
→	15	NAK	15	NAK	15	NAK	15	NAK
ESC	1B	ESC	1B	ESC	1B	ESC	1B	ESC
SPACE	20	SP	20	SP	20	SP	20	SP
'"	27	'	27	'	22	"	22	"
,<	2C	,	2C	,	3C	<	3C	<
-_	2D	-	1F	US	5F	_	1F	US
.>	2E	.	2E	.	3E	>	3E	>
/?	2F	/	2F	/	3F	?	3F	?
0)	30	0	30	0	29	)	29	)
1!	31	1	31	1	21	!	21	!
2@	32	2	00	NUL	40	@	00	NUL
3#	33	3	33	3	23	#	23	#
4$	34	4	34	4	24	$	24	$
5%	35	5	35	5	25	%	25	%
6^	36	6	1E	RS	5E	^	1E	RS
7&	37	7	37	7	26	&	26	&
8*	38	8	38	8	2A	*	2A	*
9(	39	9	39	9	28	(	28	(
;:	3B	;	3B	;	3A	:	3A	:
=+	3D	=	3D	=	2B	+	2B	+
[{	5B	[	1B	ESC	7B	{	1B	ESC
|	5C	\	1C	FS	7C	|	1C	FS
]}	5D	]	1D	GS	7D	}	1D	GS
`~	60	`	60	`	7E	~	7E	~
A	61	a	01	SOH	41	A	01	SOH
B	62	b	02	STX	42	B	02	STX
C	63	c	03	ETX	43	C	03	ETX
D	64	d	04	EOT	44	D	04	EOT
E	65	e	05	ENQ	45	E	05	ENQ
F	66	f	06	ACK	46	F	06	ACK
G	67	g	07	BEL	47	G	07	BEL
H	68	h	08	BS	48	H	08	BS
I	69	i	09	HT	49	I	09	HT
J	6A	j	0A	LF	4A	J	0A	LF
K	6B	k	0B	VT	4B	K	0B	VT
L	6C	l	0C	FF	4C	L	0C	FF

KEY	NORMAL	CHAR	CONTROL	CHAR	SHIFT	CHAR	BOTH	CHAR
M	6D	m	0D	CR	4D	M	0D	CR
N	6E	n	0E	SO	4E	N	0E	SO
O	6F	o	0F	SI	4F	O	0F	SI
P	70	p	10	DLE	50	P	10	DLE
Q	71	q	11	DC1	51	Q	11	DC1
R	72	r	12	DC2	52	R	12	DC2
S	73	s	13	DC3	53	S	13	DC3
T	74	t	14	DC4	54	T	14	DC4
U	75	u	15	NAK	55	U	15	NAK
V	76	v	16	SYN	56	V	16	SYN
W	77	w	17	ETB	57	W	17	ETB
X	78	x	18	CAN	58	X	18	CAN
Y	79	y	19	EM	59	Y	19	EM
Z	7A	z	1A	SUB	5A	Z	1A	SUB
0	30	0	30	0	30	0	30	0
1	31	1	31	1	31	1	31	1
2	32	2	32	2	32	2	32	2
3	33	3	33	3	33	3	33	3
4	34	4	34	4	34	4	34	4
5	35	5	35	5	35	5	35	5
6	36	6	36	6	36	6	36	6
7	37	7	37	7	37	7	37	7
8	38	8	38	8	38	8	38	8
9	39	9	39	9	39	9	39	9
.	2E	.	2E	.	2E	.	2E	.
+	2B	+	2B	+	2B	+	2B	+
-	2D	-	2D	-	2D	-	2D	-
*	2A	*	2A	*	2A	*	2A	*
/	2F	/	2F	/	2F	/	2F	/
PAUSE	13	DC3	13	DC3	13	DC3	13	DC3
BREAK	03	ETX	03	ETX	03	ETX	03	ETX
ENTER	0D	CR	0D	CR	0D	CR	0D	CR
F1	00	NUL	0	NUL	0	NUL	0	NUL
F2	01	SOH	1	SOH	1	SOH	1	SOH
F3	02	STX	2	STX	2	STX	2	STX
F4	03	ETX	3	ETX	3	ETX	3	ETX
F5	04	EOT	4	EOT	4	EOT	4	EOT
F6	05	ENQ	5	ENQ	5	ENQ	5	ENQ
F7	06	ACK	6	ACK	6	ACK	6	ACK
F8	07	BEL	7	BEL	7	BEL	7	BEL
F9	0C	FF	0C	FF	0C	FF	0C	FF
F10	18	CAN	18	CAN	18	CAN	18	CAN

6.1.1 Accessing the keyboard

The keyboard input status and the ASCII keycode can be accessed by reading the hardware locations in Table 6.3.

ADDRESS	OPERATION	FUNCTION
$C00X	R7	Bit 7=1; keyboard has been pressed Bit 7=0; no key has been ever pressed Bit 0-6 ASCII code of the pressing key
$C010	R7	Bit 7=1 ; Any one key is being pressed down Bit 7=0 ; No key is being pressed down
$C01X	R	Bit 0-6; ASCII code of the pressing key
$C010	R/W	reset the keyboard strobe latch
$C01X	W	reset the keyboard strobe latch.

Table 6.3 keyboard hardware locations

When any key is pressed, the keyboard strobe latch is set to 1. This keyboard strobe status is read by location $C00X. At the same time the ASCII code of that key is contained in Bit 0 - 6. Reading $C01X can also read the ASCII code, but it will reset the keyboard strobe status. To check if any one key is being pressed down, the location $C010 can be accessed but it will reset the keyboard strobe status at the same time. Usually we can check $C00X first to see if any key has been pressed before, then we can read the ASCII code by reading $C01X. Fig 6.3. illustrates this event.

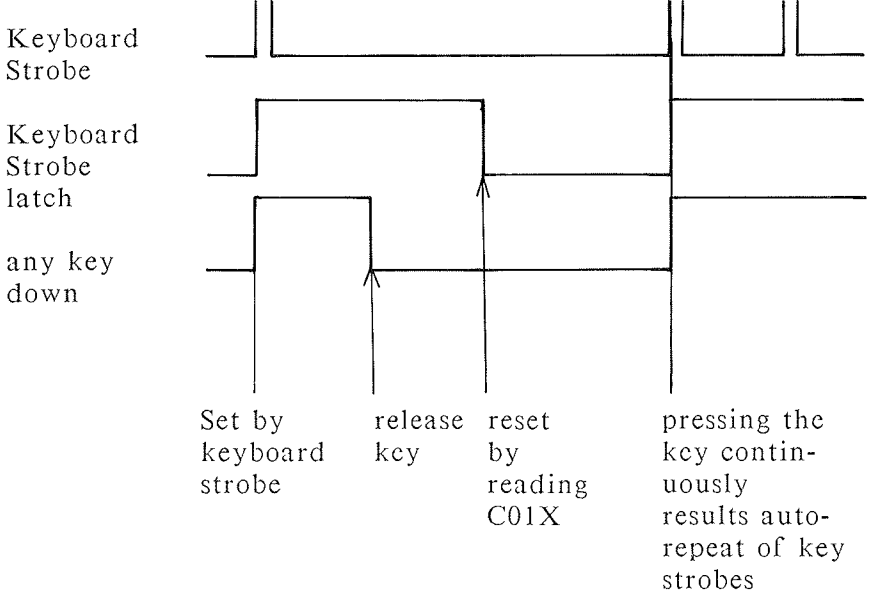


Fig 6.3 Timing of pressing the keyboard

The keyboard encoding IC has auto-repeat function. If you keep on pressing the key for over a time limit, that key will be generated automatically at a fixed rate.

6.1.2 Special function keys and switches

There are some keys which does not generate any ASCII code. These keys affects the system immediately

a) [CTRL] - [RESET]

Pressing these two keys simultaneously will reset the micro-computer system.

b) [△] , [▲]

The status of these two keys can be read from locations $C061 and $C062 respectively. These two keys can be used as game control keys.

c) [CAPS LOCK]

This is a toggling switch. When upper case letters are activated, the CAPS LOCK indication light will glow.

d) 40 / 80 switch

80 column text mode can be turned on only when the 40/80 switch is set to 80 position.

ADDRESS	OPERATION	FUNCTION
$C060	R7	BIT 7=1; 40/80 switch set to 40 position
		BIT 7=0; 40/80 switch set to 80 position
$C061	R7	BIT 7=1; △ pressed
$C062	R7	BIT 7=1; ▲ pressed

Table 6.4 Special keys and switch location

6.2 SOUND

ADDRESS	OPERATION	FUNCTION
$C03X	R	Speaker toggles when this is accessed

Table 6.5 Sound control hardware location

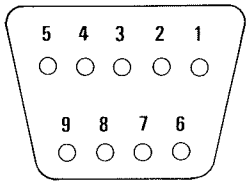
A speaker is built-in. Sound of different tones will be generated depending on the rate $C03X is being accessed. The loudness of the speaker can be adjusted by the volume control. The user can also use the ear-phone jack if he doesn't want to use the internal speaker.

6.3 GAME PORT

The game control port can be functionally divided into 2 parts namely:

a) The switch and analog (paddle) inputs

b) The mouse inputs



1 MOUSE SIGNATURE / GAMESW1
2 + 5v
3 GND
4 XDIR
5 XINT / PDL0
6 N.C
7 MOUSE BUTTON / GAMESW0
8 YDIR / PDL1
9 YINT

Fig 6.4 Game Port Connector Pin assignment

The schematic of the game port is drawn in Fig 6.5.

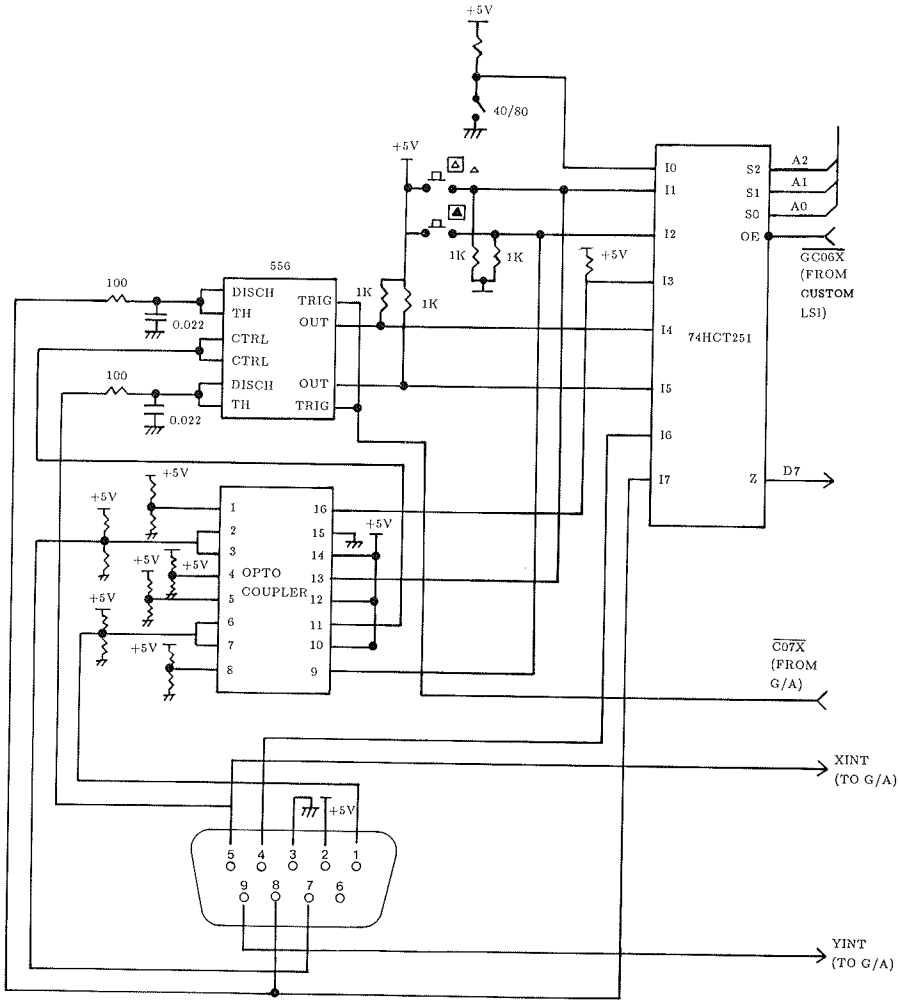


Fig 6.5 Game Port Schematic

6.3.1 Switch and analog (paddle) input

The connection of the paddle and switch input is shown in Fig 6.6.

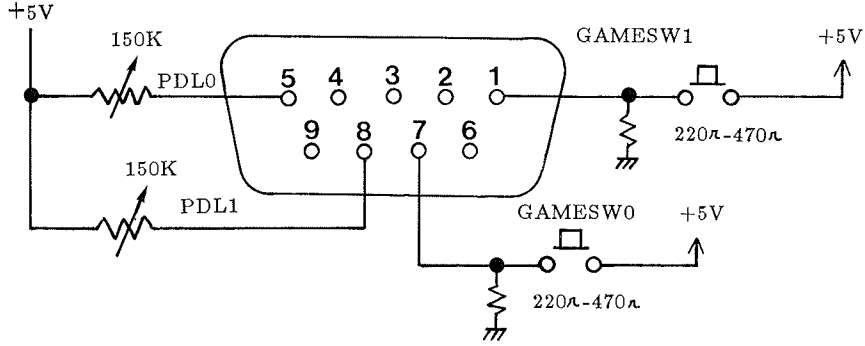


Fig 6.6 paddle and switch input requirement

The hardware page locations are listed in Table 6.6.

ADDRESS	OPERATION	DESCRIPTION
$C061	R7	Bit 7=1 and Bit 7 of $C063 equals 1 ; GAMESW0 is pressed: if only bit 7=1 then △ is pressed
$C062	R7	Bit 7=1 ; ▲ or GAMESW1 is pressed
$C063	R7	Bit 7=1 ; and bit 7 of $C061 is 1, then GAMESW0 is pressed. If bit 7=0 then mouse button is pressed.
$C064	R7	Analog (paddle 0) input
$C065	R7	Analog (paddle 1) input
$C07X	R/W	trigger paddle timer

Table 6.6 hardware page locations of switch and analog input

The analog inputs can be connected to two 150K paddles as game control. Usually, a two-axis joystick is connected to these inputs. The keys ▲ and △ , connect to these two locations permanently. The switch inputs GAMESW0 and GAMESW1 also connect to these locations. However, to distinguish these game switch inputs from mouse button, $C063 has to be read. When reading $C063 gives a one in bit 7 GAMESW0 is pressed. If the mouse button is pressed, the value will be 0 for bit 7 of $C063.

6.3.2 Mouse input

The game port of the computer also accepts mouse as its game port input. The connection of the mouse to the game port is shown in Fig 6.7.

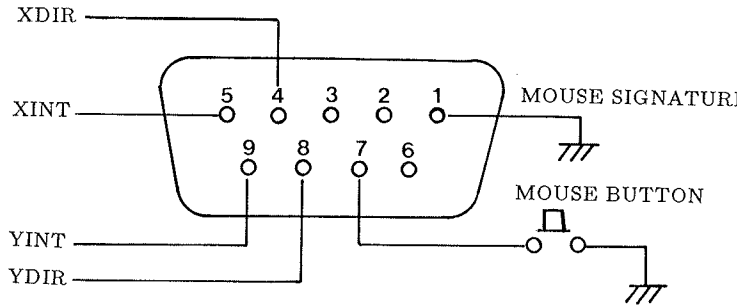
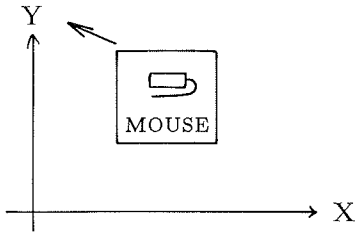


Fig 6.7 connection of game port to mouse

When a mouse is moved along a flat surface, square pulses are output on pin XINT, XDIR, YINT and YDIR. The XINT and YINT are the interrupt signals to the MPU. With software control, either the rising or the falling edge of the mouse interrupt signal can cause interrupt to the MPU. The direction of movement of the mouse can be observed from the XDIR or YDIR signals.

To illustrate the operation, we can study an example. The XINT rising edge causes the interrupt. When the MPU receives the interrupt, it immediately checks the XDIR. If the XDIR is at a high level, the mouse moves in -X direction. Similarly the value of YDIR is '0' immediately after YINT interrupt edge, the mouse must move in a +Y direction.



mouse movement direction

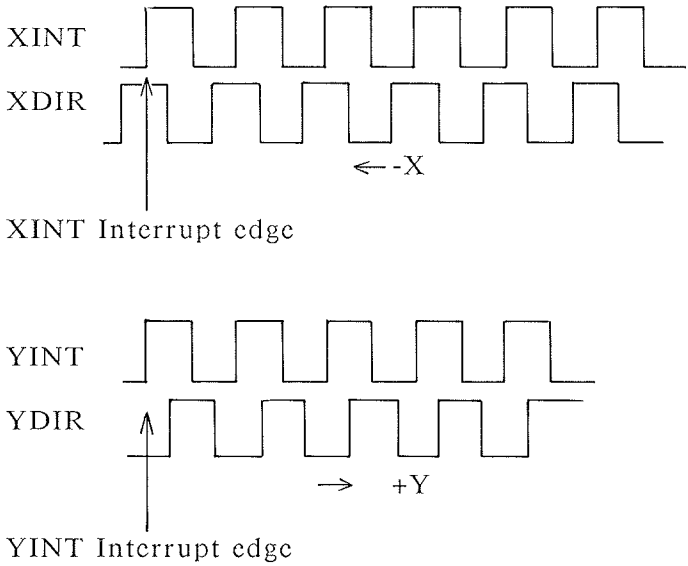


Fig 6.8 Example to illustrate the mouse operation direction

The mouse signature pin is connected to ground so as to disable the paddle analog input. The mouse button status can be read from $C063.

The vertical backdrop signal (VBDRP) can cause an interrupt to the MPU if this interrupt source is enabled. The programmer can then enable the mouse and update the mouse position information if there is any mouse movement. The hardware page locations which control all these interrupts are listed in Table 6.7.

ADDRESS	OPERATION	FUNCTION
$C063	R7	Bit 7=0 : mouse button pressed
$C0C0	W	Select rising edge of XINT as int. source

$C0C1	W	Select falling edge of XINT as int. source
$C0C2	W	Select rising edge of YINT as int. source
$C0C3	W	Select falling edge of YINT as int. source
$C0C4	W	Disable XINT and YINT interrupt
$C0C5	W	Enable XINT and YINT interrupt
$C0C6	W	Disable VBDRP interrupt
$C0C7	W	Enable VBDRP interrupt
$C0C8	R7	Read selected XINT int edge; Bit 7 = 1 ; falling edge
$C0C9	R7	Read selected YINT int. edge; Bit 7 = 1 ; falling edge
$C0CA	R7	Read XINT and YINT int. enable flag Bit7= 1 ; XINT and YINT enabled
$C0CB	R7	Read VBDRP int. enable flag; Bit 7=1; VBDRP enabled
$C0CC	R7	Read XINT interrupt status Bit 7=1; XINT interrupt occurred

$C0CD	R7	Read YINT interrupt status Bit 7=1; YINT interrupt occurred
$C0CE	R7	Read VBDRP interrupt status Bit 7=1; VBDRP interrupt occurred
$C0CF	W	Reset XINT and YINT interrupt status
$C07X	W	Reset VBDRP interrupt status

Table 6.7 Mouse Hardware Page Locations

There are routines in the monitor program to handle the mouse movement. The programmer may use these routines instead of writing their own.

6.4. PARALLEL PRINTER PORT

The computer can support both parallel printer and serial printer. The option is selected by a small slide switch on the front panel. The parallel printer port is a D-type connector next to the game port.

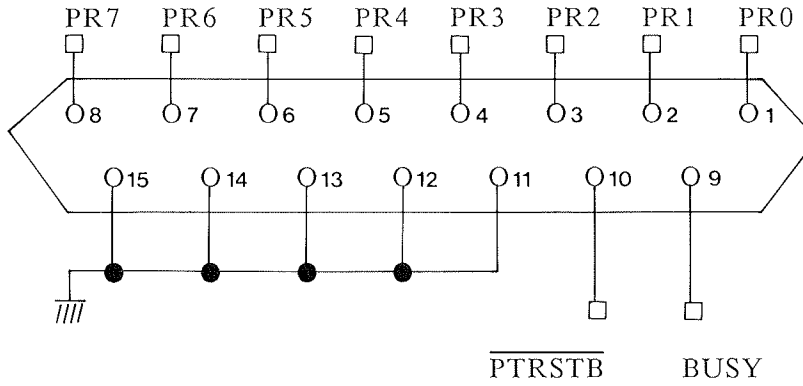


Fig 6.9 parallel printer port pin assignment

The timing diagram of sending data to the printer is shown in Fig 6.10. Before sending data to a printer, the printer busy signal is checked first. If the printer is busy, the microcomputer will wait until the printer is not busy.

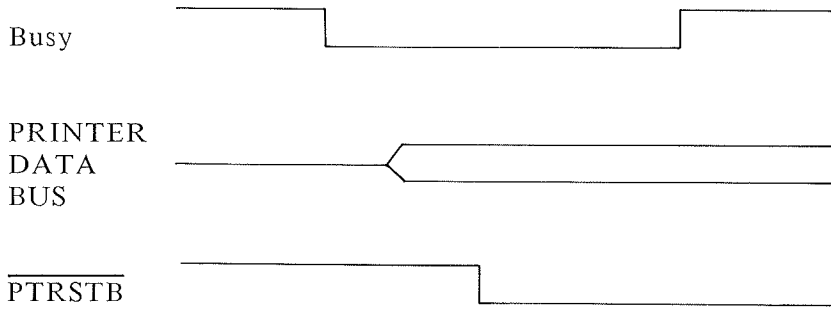


Fig 6.10

The hardware page locations of manipulating the parallel printer are listed in Table 6.8.

ADDRESS	OPERATION	DESCRIPTION
$C09X	W	Send PRINTER STROBE to printer
$C1C1	R7	Bit 7 = 1 ; printer Busy

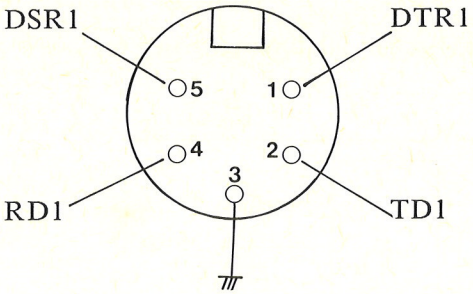
Table 6.8 Hardware page locations of parallel printer

6.5 SERIAL I/O

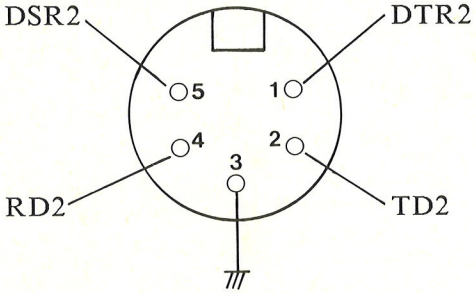
The computer has two serial ports built-in. Port 1 is used for the serial printer and PORT 2 is for the modem. However, with the help of system utility programs, the two I/O ports characteristics can be reconfigured. The hardware page locations of the two serial I/O ports are listed in Table 6.9. The pin assignmemts of the connector are shown in Fig 6.11.

ADDRESS	DESCRIPTION
$C098	Port 1 ACIA receive/transmit data register.
$C099	Port 1 ACIA status register
$C09A	Port 1 ACIA command register
$C09B	Port 1 ACIA control register
$C0A8	Port 2 ACIA receive/transmit data register.
$C0A9	Port 2 ACIA status register
$C0AA	Port 2 ACIA command register
$C0AB	Port 2 ACIA control register

Table 6.9 Hardware page locations of ACIA of the serial I/O ports



Serial Port 1



Serial Port 2

Pin	Pin name	Description
1	DTR1 DTR2	Data Terminal Ready output
2	TD1 TD2	Transmit Data output
3	GND	Power and signal common ground
4	RD1 RD2	Receive Data input
5	DSR1 DSR2	Data Set Ready input

Fig 6.11 Pin assignment of serial port 1 and 2

The schematic diagram of the serial ports is shown in Fig 6.12.

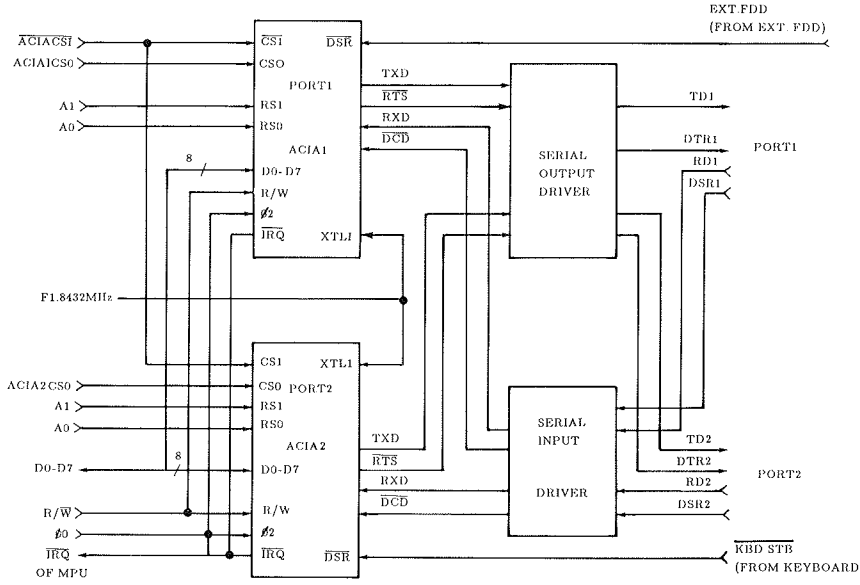
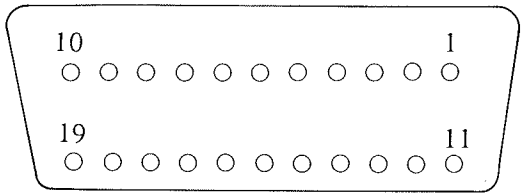


Fig 6.12 Block diagram of the two serial I/O ports

6.6 THE FLOPPY DISK DRIVE (FDD)

In the computer, a 5 1/4" floppy disk drive is built-in. The control logic of the drive is inside a custom-made LSI. One external drive can be connected via the external drive connector.



Pin	Description
1	GND
2	GND
3	GND
4	GND
5	-12V
6	+5V
7	+12V
8	+12V
9	EXT FDD; connect to serial port 1 DSR1
10	WPROT
11	PHI0
12	PHI1
13	PHI2
14	PHI3
15	WREQ
16	SIDE 1
17	EXT FDD ENABLE : active low
18	RDATA
19	WDATA

Fig 6.13 Pin assignment of the external drive connector

6.6.1 Floppy disk drive controller

The computer has a built-in floppy disk drive, which allows you to read or store program by another custom-made integrated circuit, GA2, which

- converts data from computer memory to proper disk format
- retrieves data from disk to a readable form for CPU
- house keeping operations such as drive motor on or off, drives select, step-in and step-out of disk magnetic heads

Table 6.10 shows the hardware page locations of floppy disk drive.

ADDRESS	OPERATION	FUNCTION
$C0E0	R/W	PHASE 0 OFF
$C0E1	R/W	PHASE 0 ON
$C0E2	R/W	PHASE 1 OFF
$C0E3	R/W	PHASE 1 ON
$C0E4	R/W	PHASE 2 OFF
$C0E5	R/W	PHASE 2 ON
$C0E6	R/W	PHASE 3 OFF
$C0E7	R/W	PHASE 3 ON
$C0E8	R/W	DRIVE MOTOR OFF
$C0E9	R/W	DRIVE MOTOR ON
$C0EA	R/W	SELECT BUILT-IN DRIVE
$C0EB	R/W	SELECT EXTERNAL DRIVE
$C0EC	R/W	SHIFT/READ DATA
$C0ED	R/W	LOAD LATCH/READ WRITE PROTECT
$C0EE	R/W	READ
$C0EF	R/W	WRITE

Table 6.10 Hardware page locations of floppy disk drive

6.6.2. Hardware locations and their functions

$C0E0~$C0E7

They control the voltages applied to the four phases of the stepping motor that carries the magnetic read/write head over different tracks of the floppy diskette. (Tracks are conceptual concentric circles on disk surface that store data). To move the head inwards (towards the disk center), access these switches in ascending order:

C0E0 C0E1 C0E2 C0E3

To move the head outwards (away from the disk center), access these switches in descending order:

C0E7 C0E6 C0E5 C0E4

Enough settling time must be allowed between successive accesses to these switches. Typically this requires a minimum of 25ms.

$C0E8~$C0E9

An access to $C0E8 turns off all drives while $C0E9 turns on the built-in drive.

$C0EA~$C0EB

$C0EA enables built-in drive $C0EB enables the optional external drive. Drive must be enabled before disk I/O operations can be accomplished.

$C0EC~$C0EF

These four soft-switches control the read/write operation of the disk drive and to determine the "WRITE PROTECT TAB" status. Basically they can be defined as follows (the data register is an 8 bit register that temporarily holds data from computer or from diskette):

$C0EC - shift data out of data register to diskette
$C0ED - load data to data register for subsequent shift out
$C0EE - Read data from diskette
$C0EF - Write data to diskette.

Based on their elementary functions, Table 6.11 shows the normal sequence of access:

$C0EC $C0EF	set up write data mode and strobe data latch
$C0ED $C0EF	set up write data mode and load data register
$C0EC $C0EE	set up read data mode and shift data into data register
$C0ED $C0EE	set up read data mode and read write-protect switch

Table 6.11 Access sequence to hardware location of drive control

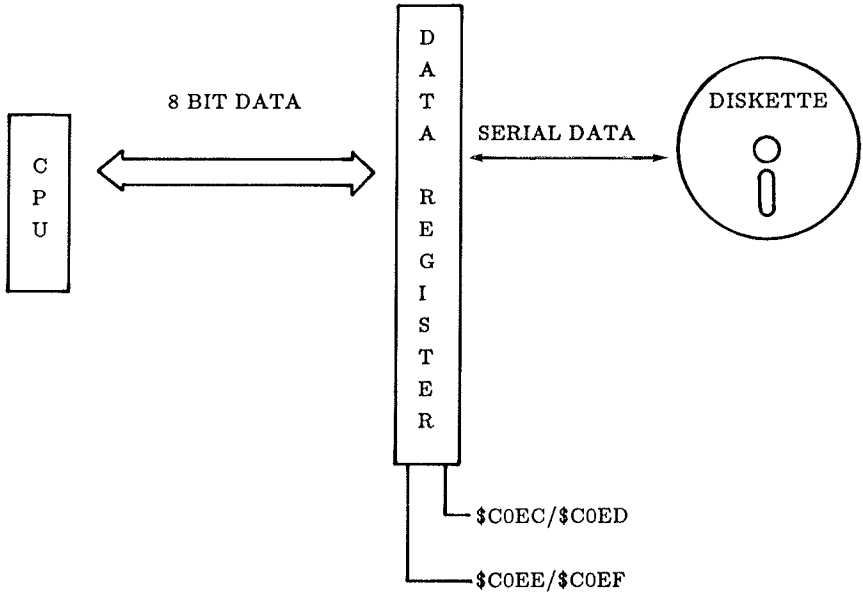


Fig 6.14 Data conversion of the drive controller

Certain timing requirement must be met for correct I/O operation.

i) **WRITE**
In order to continuously shift out information in 8 bit groups, the CPU must store data at $C08D every 32 cycles, then immediately enable shifting with a read access to $C08C. For example, to write a byte to the disk then begins to write another byte, we can have the following program segment:

```
LDX   $60
LDA   $C08D, X     ;LOAD
LDA   $C08E, X     ;READ
BMI   ERRDR        ;WRITE PROTECT ERR
LDA   DATA1
STA   $C08F, X     ;WRITE DATA1
CMP   $C08C, X     ;SHIFT (4CP)
PHA                ;(3CP)
PLA                ;(4CP)
PHA                ;(3CP)
PLA                ;(4CP)
BIT   $0           ;(3CP)
NOP                ;(2CP)
```

```
LDA   DATA2       ;(4CP)
STA   $C08D,X     ;LOAD (5CP) TOTAL:32CP
CMP   $C08C,X     ;SHIFT
```

ii) Read

The circuitry inside the custom chip handles all the necessary conversion from the magnetic flux changes to meaningful data. The user will simply check the availability of valid data by a simple program loop.

```
        LDA  C0EC  ; SHIFT
AGAIN : LDA  C0EE  ;
        BPL  AGAIN; TRY IF DATA NOT VALID.
```

Since the MSB of a valid data must be a 1, the above program will load the accumulator with a valid data byte just passing over the read/write head.

6.7 EXPANSION SLOT

One expansion slot is housed on the left side of the computer. The pin assignment of this expansion slot is listed in Table 6.12. It is a 25 pin x 2 socket. The pin-out is shown Fig 6.15.

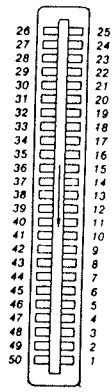


Fig 6.15 Pin numbering of expansion slot

Pin number	Signal name	Description
1	$\overline{C7XX}$	active low on access to slot $C700 - $C7FF
2	A0	address bus
3	A1	"
4	A2	"
5	A3	"
6	A4	"
7	A5	"
8	A6	"
9	A7	"
10	A8	"
11	A9	"
12	A10	"
13	A11	"
14	A12	"
15	A13	"
16	A14	"
17	A15	"
18	$R/\overline{W}$	Read / Write
19	$\overline{CSYNC}$	Composite sync
20	$\overline{IOSTB}$	active low on access slot $C800 - $CFFF
21	RDY	Ready signal to 65C02
22	$\overline{DMA}$	Direct Memory Access
23	NC	
24	NC	
25	VDD	+5V
26	GND	
27	NC	
28	$\overline{C0DX}$	active low on access to $C0D0 - $C0DF
29	$\overline{NMI}$	Non-Maskable Interrupt to 65C02
30	$\overline{IRQ}$	Interrupt Request to 65C02
31	$\overline{UPRST}$	RESET signal
32	$\overline{INH}$	memory inhibit signal
33	-12V	

34	$\overline{\text{C5XX}}$	active low on access to $C500 - $C5FF
35	F3.58M	3.58 MHz clock
36	F7M	7 MHz clock
37	Q3	2MHz clock
38	Ø1	1MHz clock (180° out of phase with Ø0)
39	µPSYNC	Sync signal to 65C02
40	Ø0	1MHz clock
41	$\overline{\text{C0FX}}$	active low on access to $C0F0 - $C0FF
42	D7	system data bus
43	D6	"
44	D5	"
45	D4	"
46	D3	"
47	D2	"
48	D1	"
49	D0	"
50	+12V	

Table 6.12 Expansion slot pin assignment

CHAPTER 7

THE COMPUTER FIRMWARE

The firmware that is stored permanently in the ROMs of the computer can be divided into three main areas:

1. $C100 - $CFFF Various I/O drivers
2. $D000 - $F7FF The computer Basic
3. $F800 - $FFFF System code

7.1 SYSTEM CODE

Most of the "System code" resides at $F800 - $FFFF, though a few system routines lie in other areas of ROM. The System code forms the low-level basis for how the computer works. It can be divided into three main categories:

1. Power-up, interrupt, BRK, and CONTROL-RESET handling
2. Miscellaneous routines available for use by application programs
3. The System Monitor program

7.1.1 Power-up and CONTROL-RESET

When the computer is turned on, circuitry inside the computer notices that the computer is being turned on and responds by asserting the RESET line for a moment. (This action is equivalent to a user pressing CONTROL-RESET.)

Any time RESET is asserted, whether by turning the computer on or by pressing CONTROL-RESET, other circuitry inside the computer responds instantly with several actions (described in detail elsewhere in this manual). The most important actions are that the main bank of RAM is enabled, and the ROM area is enabled for reading. This puts the computer in a "standard" memory configuration for every Reset.

The 65C02 microprocessor also responds to the Reset. It abandons whatever it was previously doing, and reads the two-byte address stored in locations $FFFC,$FFFD. This two-byte address comes from the ROM, since the ROM is always enabled at this point.

The microprocessor begins executing the code found at the address it just read. This code is also in the ROM area.

The Reset-handling code in ROM begins by initializing several system locations in memory, and setting up a standard 40-column text-only video display. It checks the setting of the 40/80 column switch to determine whether or not to "map in" the 80-column firmware at $C300-$C3FF to make the 80-column display available.

It then checks if the hollow triangle key is currently being held down (part of CONTROL-HOLLOW TRIANGLE-RESET). If it is, then it "blasts" one byte of memory in every page from page $01 to page $BF. The byte blasted in page $03 is the "application Reset checksum" (described below). The stack pointer is also reset to $FF.

Next, the Reset handler beeps the speaker for about a tenth of a second. It then checks if the "P" key (upper or lower case P or CTRL-P) is being held down (CONTROL-P-RESET). If it is, then it goes to execute the port configuration program described in the computer owner's guide.

Otherwise, it next checks the "application Reset vector and checksum". An application can store the address of its own Reset handler into the application Reset vector at locations $3F2, $3F3. For this vector to be recognized, the application must also exclusive-OR the high byte of the address with the constant $A5, and store the result into the application Reset checksum at location $3F4.

The Reset handler itself exclusive-ORs the high byte of the application Reset vector with $A5 and compares it to the application Reset checksum. If they are identical, then the Reset handler "knows" that the Reset vector was correctly set up. If it was, the Reset handler does one more check: if the Reset vector points to $E000, then it changes the vector to point to $E003 and jumps directly to $E000. (This allows Basic to be "coldstarted" on the first Reset or power-up, and "warmstarted" on subsequent Resets.) If the vector does not point to $E000, then the Reset handler simply jumps to the address contained in the vector.

Note: The application Reset vector will not match the checksum if the computer was just turned on (since nothing has initialized it yet) or if CONTROL-HOLLOW TRIANGLE-RESET was pressed (since the byte-blasting that occurred stored an invalid Reset checksum).

If the application Reset vector did not match the checksum, then the Reset handler takes several more steps. It initializes the application Reset vector and checksum to correctly point to $E000 (so that if CONTROL-RESET is subsequently pressed, then Basic will be coldstarted). It clears the screen and displays the logo on the centre of the top line. It then checks if a bootable disk device is plugged into the expansion slot ("slot 7") on the side of the computer. If one is, then the handler jumps to it so that the disk device will boot. If not, then the built-in floppy driver is called to boot a disk in the built-in floppy disk drive.

7.1.2 Interrupt and BRK Handling

Interrupt handling can usually be ignored by application programs on the computer. The mouse and the two serial ports on the computer can generate interrupts, but these are normally handled transparently by computer firmware.

When the 65C02 microprocessor has determined that an IRQ interrupt has occurred or a BRK instruction has been executed, it pushes the program counter and status register values onto the stack, reads the two-byte address from locations $FFFE,$FFFF, and jumps to that address. Note, however, that locations $FFFE,$FFFF can be occupied by either ROM, main RAM, or auxiliary RAM. If an application program is using either the main or auxiliary upper 16K of RAM and requires interrupts to work and/or BRK instructions to be handled correctly, it must store a valid interrupt handler address into $FFFE, $FFFF of these areas. (Initializing the mouse will automatically copy the address of the built-in interrupt handler into both the main and auxiliary upper 16K RAM areas.)

In the built-in firmware in ROM, $FFFE,$FFFF point to the computer interrupt handler that begins in the $C4 page. In the computer, $C400-$C4FF is shared between mouse routines and interrupt handler rountines. This page of ROM is always mapped in and always available, regardless of what memory configuration soft-switches have been accessed. This makes it the ideal entry point for the interrupt handler, because it means that interrupts can occur and be handled correctly with any memory configuration.

The interrupt handler in page $C4 saves the contents of the registers along with a byte representing the current memory

configuration, then switches to main memory, ROM active. It then checks whether a BRK or a real interrupt occurred.

If a BRK occurred, it transfers information about the BRK into several zero page locations:

$3A,$3B Program counter value (The number stored here will actually be 2 greater than the address of the BRK that occurred.)

$44 Memory configuration - Each bit is a Boolean value (1=TRUE, 0=FALSE)

- Bit 7 = Auxiliary zero page, stack, & upper 16K are switched in
- Bit 6 = Both 80-column store and "page 2" access are switched in
- Bit 5 = Auxiliary 48K RAM is switched in for reading
- Bit 4 = Auxiliary 48K RAM is switched in for writing
- Bit 3 = Upper 16K is switched in for reading
- Bit 2 = Bank 1 of the upper 16K is switched in
- Bit 1 = Bank 2 of the upper 16K is switched in
- Bit 0 = Internal $Cxxx ROM is switched in

$45 Accumulator value
$46 X-register value
$47 Y-register value
$48 Processor status register value
$49 Stack pointer value

The BRK handler then reads the two-byte address from $3F0,$3F1, and jumps to the routine at this address. Unless altered by an application program, this vector will point to the "second half" of the BRK handler. This second half switches to a standard 40-column text-only display, and prints:

BRK: $nnnn

where nnnn is 2 greater than the address of the BRK instruction, then enters the System Monitor. A technically inclined user can then examine location $44-$49 to determine the circumstances surrounding the BRK.

(Note: The BRK handler will always save the correct register values into zero page, even if the BRK occurs while the auxiliary memory stack and zero page are active, and the stack needs to be switched.)

If an interrupt occurred (rather than a BRK instruction), the interrupt handler determines if the source of the interrupt was either an "invisible" mouse interrupt or a serial port handshake-line interrupt. In either case, the interrupt is simply handled internally, then control is returned back to the interrupted program.

If the source of the interrupt was either a "visible" mouse interrupt, a modem port receiver or transmitter interrupt, or an interrupt generated by a circuit card plugged into the expansion slot, then it pushes some return information onto the stack, reads the two-byte address stored at locations $3FE,$3FF, and jumps to this address. This vector must be initialized by the application ahead of time! Otherwise, the computer will hang or behave erratically.

The application interrupt routine should exit with an RTI instruction. The RTI will return control to the computer firmware, which will unstack the configuration information, restore the memory configuration and register values, then return back to the interrupted program.

Note: Applications which use the mouse should ALWAYS use the built-in interrupt handling firmware, rather than handling the interrupts directly themselves. The internal handling of the mouse on the computer is not identical to other computers. (A couple of current mouse-based software programs do bypass the built-in interrupt handler. The computer's mouse firmware is designed to accomodate these programs, but these accomodations should not be relied upon for future products.) Application programs which need visible mouse interrupts should connect their interrupt routines through the application interrupt vector at $3FE,$3FF rather than intercepting $FFFE,$FFFF.

7.1.3 Miscellaneous Routines

The Computer System code contains a large number of useful routines and entry points. Application programs can call these routines to help accomplish a variety of tasks. Listed below are entries for each of the supported routines. Each entry includes the routine address, a brief description of the routine, and any other information that might be necessary to use it.

WARNING: Do not attempt to call or make use of any System code that is not documented below. Any undocumented routines are subject to possible changes in future versions of the computer

ROMs, and will not correspond to routines in other 6502/65C02 based computers. (A few entry points which are not documented are provided for the sake of compatibility with current software products, but these entry points are not guaranteed in future ROM versions, and should not be relied upon for future software development.) Also, do not expect 6502/65C02 registers to contain specific values on return, unless specified below.

$C305 Get a character from the keyboard, displaying a solid box cursor using 80-column firmware routines. Returns: Acc = character typed. The 80-column firmware must be initialized for this to work. A preferred method is to initialize the 80-column firmware and obtain characters through the routine at $FD0C.

$C307 Print a character to the text display, using 80-column firmware routines. Pass in: Acc = character to print. Returns: Acc, Xreg, Yreg unchanged. The 80-column firmware must be initialized for this to work. A preferred method is to initialize the 80-column firmware and print characters through the routine at $FDED.

$C311 Move a block of data from main 48K RAM to auxiliary 48K RAM, or vice versa. Pass in: $3C,$3D contain two-byte address of first byte in block to move, $3E, $3F contain two-byte address of last byte in block to move, $42,$43 contain two-byte address of where to move block to, carry flag clear to move from auxiliary memory to main memory, else carry flag set to move from main memory to auxiliary memory. The computer 40/80-column switch must be set to 80 for this routine to be available.

$C314 Switch program execution from main 48K RAM to auxiliary 48K RAM, or vice versa, also specifying which zero page and stack area to use. Pass in: $3ED,$3EE contain two-byte address of code to execute next, carry flag clear for execution in main 48K RAM, else carry flag set for execution in auxiliary 48K RAM, overflow flag clear for main zero page and stack, else overflow flag set for auxiliary zero page and stack. The Computer 40/80-column switch must be set to 80 for this routine to be available.

$F800 Plot a block in current color on the low-resolution graphics screen. Pass in: Yreg = horizontal coordinate, Acc = vertical coordinate.

$F819 Draw a horizontal bar in current color on the low-resolution graphics screen. Pass in: Yreg = left coordinate, $2C = right coordinate, Acc = vertical coordinate.

$F828 Draw a vertical bar in current color on the low-resolution graphics screen. Pass in: Yreg = horizontal coordinate, Acc = top coordinate, $2D = bottom coordinate.

$F832 Clear the entire low-resolution graphics screen to black.

$F836 Clear the top 40 lines of the low-resolution graphics screen to black. (If in mixed text/graphics mode, this leaves the bottom four lines of text unaffected.)

$F864 Set the current color for low-resolution graphics. Pass in: Acc = color number (0 - 15). The color for each color number can be found in the BASIC manual.

$F871 Determine the color at a given coordinate on the low-resolution graphics screen. Pass in: Yreg = horizontal coordinate, Acc = vertical coordinate. Returns: Acc = color number.

$F941 Print a 4-digit hex number. Pass in: Acc = high byte of number, Xreg = low byte of number.

$F948 Print three spaces.

$F94A Print 1 to 256 spaces. Pass in: Xreg = number of spaces to print.

$FB1E Read a game paddle or one axis of a joystick. Pass in: Xreg = paddle number (0 or 1). Returns: Yreg = value of paddle (0 to 255).

$FB2F Switch in the standard text display and initialize the window to the full screen. If 80-columns are already active, will initialize to 80-column text display, otherwise will initialize to 40-column display.

$FB39 Alternate entry to $FB2F, does not enforce display page 1 or (invisible) low-resolution graphics mode.

$FB40 Switch in the mixed low-resolution graphics and text display, and set a 4-line text window at the bottom of the screen.

$FB5B Set the vertical cursor position. Pass in: Acc = new vertical position.

$FB6F Set the correct application Reset checksum. Pass in: $3F2,$3F3 contain desired application Reset vector address. Returns: $3F4 contains correct checksum for the Reset vector address.

$FBDD Pause for about one hundredth of a second, then beep the speaker for about a tenth of a second. (If you want to beep the speaker repeatedly - admittedly an annoying idea - the pause is just long enough to separate the sound into distinct beeps.)

$FBE2 Beep the speaker for about a tenth of a second (no pause).

$FBF4 Advance the cursor one position.

$FC10 Back up the cursor one position.

$FC1A Move the cursor up one line.

$FC22 Set the vertical cursor position. Pass in: $25 contains new vertical position.

$FC42 Clear from the current cursor position to the bottom of the text window.

$FC58 Clear the entire text window and place the cursor at the upper left corner of the text window.

$FC62 Move the cursor to the leftmost position on the next line down.

$FC66 Move the cursor down one line.

$FC70 Scroll the text window up one line.

$FC9C Clear from the current cursor position to the end of the line.

$FCA8 Pause for a moment. Pass in: Acc = how long to pause. Approximate example times: $01 = 32 microseconds, $04 = 110 microseconds, $10 = 1 millisecond, $40 = 11 milliseconds, $80 = 45 milliseconds, $00 = 170 milliseconds (.17 seconds).

$FD0C Call the current character input routine. Returns: Acc = character input.

$FD1B Character input routine for 40-column display. Blink a checkerboard cursor at the current cursor position, waiting for a key to be pressed, generating a random number in locatins $4E, $4F. Returns: Acc = key pressed. A preferred method is to initialize 40-column I/O, then call $FD0C instead.)

$FD35 Character input routine, handling ESCape codes for either 40 or 80 column display.

$FD67 Print carriage return, prompt character, and read a line of text into page $02 from the current character input routine, terminated with carriage return. Handles ESCape codes, left and right arrow keys (for backspace and retype), and CTRL-X to cancel line. Returns: Xreg = number of characters in line, not including carriage return.

$FD6A Similar to $FD67, but does not print initial carriage return.

$FD6F Similar to $FD6A, but does not print prompt character.

$FD8B Clear from current cursor position to end of line, then print carriage return.

$FD8E Print carriage return to current character output port.

$FDDA Print a two-digit hex number. Pass in: Acc = number to print.

$FDE3 Print a single-digit hex number. Pass in: Acc = number to print.

$FDED Print character through current character output port. Pass in: Acc = character to print.

$FDF0 Print character to text display. Pass in: Acc = character to print. Returns: Acc, Xreg, Yreg unchanged. A preferred method is to initialize the 40-column display and print characters through the routine at $FDED.

$FE1F "Computer compatibility routine" - a "null" routine in the computer.

$FE2C Copy a block of data from one area of memory to another. Pass in: $3C,$3D contain address of first byte of block to copy, $3E,$3F contain address of last byte of block to copy, $42,$43 contain address of where to copy block to.

$FE80 Set normal (white-on-black) character display.

$FE84 Set inverse (black-on-white) character display.

$FE89 Set character input to standard 40-column routines.

$FE8B Set character input to given port. Pass in: Acc = desired port number.

$FE93 Set character output to standard 40-column routines.

$FE95 Set character output to given port. Pass in: Acc = desired port number.

$FF2D Print "ERR" and beep the speaker.

$FF3A Print CTRL-G (bell character) to the current character output port.

$FF3F Retrieve register contents from zero page: $45 -> Acc, $46 -> Xreg, $47 -> Yreg, $48 -> Processor status register.

$FF4A Save register contents into zero page: Acc -> $45, Xreg -> $46, Yreg -> $47, Processor status register -> $48, stack pointer -> $49.

$FF58 Documented location of an RTS instruction.

$FF59 Initialize standard 40-column display and character input/output routines, clear decimal mode, beep the speaker, and enter System Monitor program.

$FF65 Clear decimal mode, beep the speaker, and enter System Monitor program.

$FF69 Enter System Monitor program.

$FFA7 Attempt to parse a hex number from the line of text in page $02. Pass in: Yreg = offset into line of first character to parse. Returns: Acc = first non-hex-digit character, Xreg = nonzero if number found, zero if number not found, Yreg = offset

into line 1 past first non-hex-digit character, $3E, $3F contain parsed number, or zero if no number to parse.

$FFFA,$FFFB Contains address of NMI handler: $03FB. (Handler must be set up by application.)

$FFFC,$FFFD Contains address of Reset handler in ROM.

$FFFE,$FFFF Contains address of IRQ interrupt handler in ROM.

7.1.4 The System Monitor

Included in the Computer System code is a "System monitor", a program designed to allow technically inclined users to examine and change memory locations directly. The System Monitor includes commands for displaying the contents of memory as hexadecimal numbers, or optionally as ASCII characters and disassembled 65C02 instructions. Commands are also available for moving blocks of data from one area of memory to another, executing a 65C02 routine, and reading the 6-digit version code of the ROM. The System Monitor works strictly with hexadecimal numbers.

Warning: Because of the way input/output is mapped in the computer, changing or even examining certain areas of memory can cause the computer to hang or behave erratically. In general, you should avoid accessing locations from $C000 to $CFFF. If you want to examine these areas, you should understand the I/O mapping in this area and take appropriate care. (The I/O mapping is explained in chapter 4 of this manual.)

The three entry points into the System Monitor, as described in the last section, are $FF59,$FF65, and $FF69. The easiest way to enter the Monitor is to turn on the computer, immediately press CTRL-RESET to stop the drive, then type:

]CALL -151

(Decimal -151 is equivalent to $FF69.)

You'll see an asterisk and a blinking checkerboard cursor. You're now in the System Monitor program. Here are the System Monitor commands:

addr RETURN

To examine the content of a single memory location, type the hex address and press RETURN. You'll see the address, a dash, and the value at that address.

addr1.addr2 RETURN
To see the contents of a range of locations, type the starting address, a period, the ending address, and press RETURN. The monitor will print one or more lines, with an address, a dash, and up to eight data values on each line, covering the range you specified.

RETURN
To see the contents of a few memory locations beyond where you last looked, just press RETURN. You'll see up to eight more data values.

add:data data data... data RETURN
To change the contents of one or more consecutive memory location, type the starting address and a colon, followed by one-or-two digit data values separated by spaces. Press RETURN after the last data value. The contents of the memory locations will be changed to the values you specified.

addrL RETURN
L RETURN

To disassemble the instructions in an area of memory, type the starting address and the letter "L", and press RETURN. Twenty lines will be printed, each with the address of the instruction, the bytes shown as ASCII characters, the bytes shown as hexadecimal numbers, and the disassembled instruction. To continue disassembling, type "L" and RETURN. (Note: because of the way various ROM areas are automatically switched in and out by the computer firmware, not all areas of ROM can be directly viewed or disassembled.)

addr1<addr2.addr3M RETURN
To move a block of data from one area of memory to another, type the starting destination address, a less-than sign, the starting source address, a period, the ending source address, and the letter "M". The data will be moved from the source area to the destination area.

addrG RETURN

To execute a 65C02 routine, type the address of the routine and the letter "G". The routine will execute. If it ends with an RTS, the routine will return to the System Monitor.

Multiple commands can also be entered on one line, separated by spaces. If you want to enter bytes into memory and do another command on the same line, you can separate the bytes entered from the other commands with the letter "N". For example:
*addr:data data data N addrG RETURN

7.2 INPUT/OUTPUT ROUTINES

The input/output routines in the computer are divided into 8 ports, numbered 0 to 7. They are:
Port 0 - built-in 40-column display
Port 1 - printer port
Port 2 - serial communications (modem) port
Port 3 - built-in 80-column display
Port 4 - mouse port
Port 6 - disk drive
(Port 5 and 7 are reserved for the expansion connector on the left side of the computer.)

Most of the I/O firmware resides in the area $C100 - $CFFF. Below is a memory map for this area.

	"Internal"	Printer	Modem	Mouse/IRQ	Disk
	----------	-------	-------	----------	------
C100	----------	-------			
		Port 1			
C200		-------	------		
			Port 2		
C300			------		
	Port 3				
C400				------	
	Port 4			Port 4	
C500	------			------	
C600					------
					Port 6
C700					------
C800		------	------		------
CFFF		------	------		------

Fig 7.1 LOGICAL INPUT/OUTPUT MEMORY MAPPING

The computer hardware allows different portions of ROM to be mapped in and out. The actual switching arrangement is described in chapter 4 of in this manual, but is really unimportant as far as most application programs are concerned. The ROM is divided into an "internal" portion that spans $C100 to $CFFF, and several page areas for ports 1, 2, 3, 4, and 6. Port 1, 2, 3, and 6 also can switch in additional areas (only one at a time!) at $C800 to $CFFF.

The "internal" portion of ROM contains additional internal code for 40 and 80 column displays, mouse and IRQ routines, etc. This portion is switched in only by the firmware as long as necessary, then switched out again to make ports 1,2,4, and 6 (as well as the actual expansion connector) available. If the 40/80 switch on the front of the computer is set to 80, then the port 3 area in the internal portion is always available, regardless of whether the rest of the internal portion is switched in.

The usual method for activating these ports from machine language is to load the port number into the Accumulator and call the routine at $FE95 to set an output port or $FE8B to set an input port. (See the Miscellaneous Routines listed earlier. These are equivalent to PR#n and IN#n.) $FE95 will store the actual address of the port driver code into the output port vector at locations $36,$37. $FE8B will store the address of the port driver code into the input port vector at locations $38,$39. Then to print a character through an output port, load the character to be printed into the Accumulator and call $FDED. This routine will actually jump "through" $36,$37 to the current output driver code. To read a character from a port, call $FD0C. This routine will jump through $38,$39 to the current input driver code.

Note: The port 1 printer routines and the port 3 80-column display routines will actually change the addresses stored in $36-$39 as part of the port initialization when they are first called. Some operating systems will also change the addresses stored here for internal bookkeeping reasons. Programs should normally call the $FDED and $FD0C routines to print and get characters through whatever port or operating system is "current", rather than access the ports directly.

For programs that do not use the System firmware at $F800 - $FFFF (for example, if the upper 16K RAM area is active), there are alternate entry points for ports 1, 2, 3, and 4. (But port 4 is a special case; see the section later on using the port 4 mouse firmware.) Ports 1, 2, and 3 contain special entry points to initialize the port, write characters to the port, read characters

from the port, and determine the status of the port. These entry points are used by Pascal, newer versions of CP/M,® and some application programs.

The initialization entry point needs to be called once, before reading from or writing to the port. After the read entry point is called, the character read will be returned in the Accumulator. Before the write entry point is called, the character to be printed should be loaded into the Accumulator. For the status call, the Accumulator should contain a "0" to check write-ready status, or a "1" to check read-ready status. It will return either carry set meaning yes-ready, or carry clear meaning not-ready.

Port 1:

Init address	=	$C100 + contents of $C10D
Read address	=	$C100 + contents of $C10E
Write address	=	$C100 + contents of $C10F
Status address	=	$C100 + contents of $C110

Port 2:

Init address	=	$C200 + contents of $C20D
Read address	=	$C200 + contents of $C20E
Write address	=	$C200 + contents of $C20F
Status address	=	$C200 + contents of $C210

Port 3:

Init address	=	$C300 + contents of $C30D
Read address	=	$C300 + contents of $C30E
Write address	=	$C300 + contents of $C30F
Status address	=	$C300 + contents of $C310

7.2.1 Port 0: 40-column Display Routines

As the computer displays each character on the 40-column screen, the firmware maintains an invisible cursor position, marking where the next character will be printed. The horizontal position ($00 to $27 for 40 columns) is kept in location $24, and the vertical position ($00 to $17) is kept in location $25.

To set a new horizontal position directly, a program can store a new value into location $24. There are two ways to directly change the vertical position: either load the new value into the Accumulator and call $FB5B, or store the value into location $25 and call $FC22.

The 40-column display routines normally print to the entire 40-column by 24-line display, though this can be changed. The video display routines (described earlier with the miscellaneous System routines) limit themselves to the current "text window". The text window is a rectangular area on the screen in which all printing takes place. Four zero-page locations determine the bounds of the text window:

$20 - horizontal position of left edge of window
$21 - width of windlow (number of characters)
$22 - vertical position of top line in window
$23 - vertical position of first line below bottom edge of window

The routines at $FB2F and $FB39 set a full-screen 40 by 24 text window. A program can also change the text window values directly. The position of the left edge added to the width should not exceed 40 for a 40-column display, or 80 for an 80-column display. The left edge of the window can be on either an even or an odd column, and the window width can be either an even or odd number of columns. If the window size or position is changed, the cursor should be placed inside of the window before any subsequent printing is done.

Besides displaying the normal printable characters, the port 0 output routines also recognize four printed control characters:

CTRL-G (ASCII $87) beeps the computer speaker.
CTRL-H (ASCII $88) moves the cursor left one position.
CTRL-J (ASCII $8A) moves the cursor down one line.
CTRL-M (RETURN, ASCII $8D) moves the cursor to the leftmost position and down one line.

When the port 0 input firmware is called to get a character from the keyboard, a checkerboard cursor blinks at the current cursor position. When a key is pressed, the original character is restored, and the ASCII value (high bit on) of the key pressed is returned in the Accumulator. An alternate entry point at $FD35 allows several special ESCape characters to be recognized from the keyboard. To use the ESCape codes, first press the ESCape key, then press the desired key:

ESC @ moves the cursor to the upper-left corner and clears the text window.

ESC E clears from the cursor position to the end of the line.
ESC F clears from the cursor position to the bottom of the text window.

The following keys can be pressed repeatedly after pressing ESCape:

ESC I or up arrow:	moves the cursor up one line.
ESC J or left arrow:	moves the cursor left one position.
ESC K or right arrow:	moves the cursor right one position.
ESC M or down arrow:	moves the cursor down one line.

7.2.2 Port 3: 80-column Display Routines

The 80-column routines are basically an enhancement of the 40-column routines. When port 3 is initialized, the video circuitry displays each character half as wide to allow up to 80 characters on each line. Locations $24 and $25 are still used to keep track of horizontal and vertical cursor position. Another location, $57B, also maintains the 80-column horizontal cursor position. Location $24, however, does maintain the horizontal position correctly for both 40 and 80 column displays on the computer.

The 80-column output firmware recognizes the same four control characters as the 40-column firmware, as well as several new ones. These control character are intended to be printed from a program. (Typing these characters directly at the keyboard may or may not cause them to be printed through the output firmware. For keyboard control, see the ESCape codes described below.)

CTRL-E: If using PASCAL, (alternate write-character entry point) "turns on" the visible cursor, displaying it as each character is printed. This is the usual setting for PASCAL.

CTRL-F: If using PASCAL, (alternate write-character entry point) "turns off" the visible cursor. Text display is faster with the visible cursor off.

CTRL-K: Clears the display from the cursor position to the bottom of the text window.

CTRL-L: Moves the cursor to the upper-left corner of the text window, and clears the entire text window.

CTRL-N: Displays subsequent text as "normal", white characters on a black background.

CTRL-O: Displays subsequent text as "inverse", black characters on a white background.

CTRL-Q: Switches to a 40-column display while keeping 80-column features.

CTRL-R: Switches back to an 80-column display.

CTRL-U: Switches to a 40-column display and turns off 80-column features.

CTRL-W: Scrolls the contents of the text window up one line, without moving the cursor.

CTRL-X: Inhibits special graphics characters display.

CTRL-Y: Moves cursor to upper-left corner of text window; does not clear text window.

CTRL-Z: Clears the entire line that the cursor is on; does not move the cursor.

CTRL-[: Displays special graphics characters instead of capital letters if "inverse" text is also set.

CTRL-\: Moves the cursor one position to the right.

CTRL-]: Clears from the cursor position to the end of the line.

CTRL-^: Moves the cursor up one line.

When the 80-column input routine is called to get a keypress, it displays an inverse-block cursor (the character at the cursor position is made inverse) while waiting for a keypress. If the alternate ESCape code entry point at $FD35 is called, additional ESCape codes become available from the keyboard. If the ESCape key is pressed, the cursor changes to an inverse plus sign.

In addition to the 40-column ESCape codes, these codes are also recognized:

ESC 4: Switches to a 40-column display while keeping 80-column features.

ESC 8: Switches back to an 80-column display.

ESC CTRL-Q: Switches to a 40-column display and turns off 80-column features.

ESC CTRL-D: Disables recognition of extra 80-column control character commands.

ESC CTRL-E: Enables recognition of extra 80-column control character commands.

7.2.3 Port 1: Printer Routines

The printer port is an output-only port. You can print characters through port 1 to a printer, but there is no provision for character input (since printers do not normally send characters back to a computer).

Besides simply sending the printed characters on to the printer, the port 1 firmware provides several other functions, depending on its current settings:

1. It will usually mask off the high bit of each character sent to a parallel printer (since many parallel printers use the high bit to represent some special codes).

2. It can count the number of characters printed between each carriage return. If the number exceeds a given line width, it can send its own carriage return to the printer to start a new line.

3. It can follow any carriage return (whether printed by a program or added by the firmware) with a linefeed character, in case the printer needs one to begin printing on the next line down.

4. It can watch if the horizontal cursor position has been changed since the last character was printed (if an HTAB or Basic command tab occurred), and send a series of spaces to the printer to accomodate the tab.

5. It can optionally also print the text to the video display.

6. It can watch for special command character sequences, for changing any of these features.

The Serial/Parallel switch on the front of the computer determines whether the serial or the parallel printer port is active. When the computer is first turned on, port 1 uses these settings:

· 80 characters per line

· Insert linefeeds after carriage return
· Do not echo characters back to the video display
· Command character is a CTRL-I
· Mask off high bit of each character (if parallel printer)
· Format is 8 data bits, 2 stop bits (if serial printer)
· Baud is 9600 (if serial printer)
· Parity is set for No parity (if serial printer)

Several of these options can be changed by the Port Configuration Program, which is described in the computer User's Guide. The options set by the Port Configuration Program remain in effect as long as the computer is on. If you want to temporarily override those settings from within a program, you can issue individual command to the printer port. Here are the command character sequences to print:

Parallel printer commands:

CTRL-I H: Send all 8 bits of each character to the printer. This is useful for some graphics printing.

CTRL-I I: Echo the text being printed back to the screen. This may cause problems for line widths other that 40 or 80 columns.

CTRL-I K: Don't automatically print a linefeed character after carriage return.

CTRL-I L: Automatically print a linefeed character after carriage return.

CTRL-I nnnN: Turn off screen echo and set the line width to nnn (where nnn is a number from 0 to 255). CTRL-I 0N sets "no line width" (do not insert carriage returns)

CTRL-I X: Send 7 (not 8) bits of each character to the printer. This is the usual setting.

CTRL-I Z: Do not format the text being printed in any way; don't insert carriage returns or linefeeds, and do not check for any more commands.

CTRL-I control-character: Change the printer command character from CTRL-I to the following control character.

Serial printer commands:

CTRL-I nnB: Set the baud rate according to the number nn, where nn is:

1	50 baud
2	75 baud
3	110 baud
4	135 baud
5	150 baud
6	300 baud
7	600 baud
8	1200 baud
9	1800 baud
10	2400 baud
11	3600 baud
12	4800 baud
13	7200 baud
14	9600 baud
15	19200 baud

CTRL-I nD: Set the data format according to the number n, where n is:

0	8 data bits, 1 stop bit
1	7 data bits, 1 stop bit
2	6 data bits, 1 stop bit
3	5 data bits, 1 stop bit
4	8 data bits, 2 stop bits
5	7 data bits, 2 stop bits
6	6 data bits, 2 stop bits
7	5 data bits, 2 stop bits

CTRL-I I: Echo the text being printed back to the screen. This may cause problems for line widths other than 40 or 80 columns.

CTRL-I K: Don't automatically print a linefeed character after carriage return.

CTRL-I L: Automatically print a linefeed character after carriage return.

CTRL-I nnnN: Turn off screen echo and set the line width to nnn (where nnn is a number from 0 to 255). CTRL-I 0N sets "no line width" (do not insert carriage returns).

CTRL-I nP: Set the parity according to the number n, where n is:

0	no parity
1	odd parity
3	even parity
5	mark parity
7	space parity

CTRL-I Z: Do not format the text being printed in any way; don't insert carriage returns or linefeeds, and do not check for any more commands.

CTRL-I control-character: Change the printer command character from CTRL-I to the following control character.

7.2.4 Port 2: Serial Communications Routines

Port 2 is designed for two-way communication with modems, computer terminals, and other serial communications devices. A program can use port 2 to receive characters (after initializing with the routine at $FE8B or the Basic command IN#2) or send characters (after initializing with $FE95 or PR#2), or both.

Since the serial input is not buffered, the firmware routines are best suited for communication at slower baud rates. For example, consider a program that gets characters from port 2 and prints them to the screen. Most of the screen display routines are quite fast, but scrolling operations take a little time. If too much time elapses between subsequent calls to read from port 2, one or more characters coming into the port may be missed or lost, because the program wasn't yet ready to receive them. For reliable high-speed communication, a modem program or terminal program that accesses the computer serial hardware directly is recommended.

Besides simply sending and receiving characters with port 2, the serial communications firmware provides several other functions, depending on its current settings:

1. It can follow any carriage return sent by the program with a linefeed character, in case the modem or terminal needs one at the end of every line.

2. It can optionally print the text to the video display.

3. It can watch for special command character sequences, for changing any of these features.

When the computer is first turned on, port 2 uses these settings:

Insert linefeeds after carriage return
Do not echo characters back to the video display
Command character is a CTRL-A
Format is 8 data bits, 1 stop bit
Baud is 300
Parity is set for No parity

As with the printer, several of the port 2 communications options can be changed by the Port Configuration Program. The options set by the Port Configuration Program remain in effect as long as the computer is on. If you want to temporarily override those settings from within a program, you can issue individual commands to the port 2. Here are the command character sequences to send:

CTRL-A nnB: Set the baud rate according to the number nn, where nn is:

1	50 baud
2	75 baud
3	110 baud
4	135 baud
5	150 baud
6	300 baud
8	1200 baud
9	1800 baud
10	2400 baud
11	3600 baud
12	4800 baud
13	7200 baud
14	9600 baud
15	19200 baud

CTRL-A nD: Set the data format according to the number n, where n is:

0	8 data bits, 1 stop bit
1	7 data bits, 1 stop bit
2	6 data bits, 1 stop bit
3	5 data bits, 1 stop bit
4	8 data bits, 2 stop bits
5	7 data bits, 2 stop bits
6	6 data bits, 2 stop bits
7	5 data bits, 2 stop bits

CTRL-A I: Echo the text being printed back to the screen. This may cause problems for line widths other than 40 or 80 columns.

CTRL-A K: Don't automatically print a linefeed character after carriage return.

CTRL-A L: Automatically print a linefeed character after carriage return

CTRL-A nP: Set the parity according to the number n, where n is:

0	no parity
1	odd parity
3	even parity
5	mark parity
7	space parity

CTRL-A Z: Do not format the text being printed in any way; don't insert carriage returns or linefeeds, and do not check for any more commands.

CTRL-A control-character: Change the command character from CTRL-A to the following control character.

CHAPTER 8

SERVICING OF THE COMPUTER

8.1 SYSTEM OVERVIEW

The computer has five principal LSIs:

.65C02 CPU
This is the Central Processing Unit which controls all operations of the computer.

.65C51 ACIA
This is the Asynchronous Communication Interface Adapter which controls the serial communication between the computer and other serial devices.

.KB-3600-PRO Keyboard Encoder
This LSI handles the keyboard input and encodes the input into ASCII codes.

.Gate Array I
This LSI deals with all the memory management and input/output, generates the video signals and all the necessary timing signals for the whole module.

.Gate Array II
This LSI controls the operations of the disk drives and generates the necessary timing signals for the optional internal Z80 expansion module.

The operational descriptions of the computer in subsequent sections can be divided into 17 different parts as follows:

.CPU
.RAM
.ROM
.Address decoding and bank selection
.Timing signals generation
.RESET circuit
.Keyboard
.Speaker
.Video circuit
.Parallel printer
.Serial printer

.Serial port 2
.Mouse
.Joystick
.Disk drive
.Power supply
.Expansion slots

Two custom-designed gate arrays are used in the computer. They are named GA1 and GA2.

The GA1 gate array is a 100-pin flat-packed chip. It is responsible for timing signals generation, video signals generation, address decoding for RAM, ROM and I/O devices, interfacing to the keyboard circuit, and generation of control signals for the whole system.

The GA2 gate array is a 64-pin DIL-packed chip. It is mainly responsible for the operations of the internal and external drives and the optional internal Z80 module.

The pin-out and signal descriptions of the two gate arrays can be found in Appendix B.

8.2 CPU

The 65C02 CPU is a complete 8-bit parallel microprocessing unit that builds on the CMOS technology. It is pin-to-pin compatible to the NMOS 6502 CPU with some improvements: this new CPU has 12 new instructions and two new addressing modes over the old 6502 CPU.

The CPU inside the computer runs at a clock frequency of 1MHz. This timing signal is derived from GA1. Fig 8.1 shows the interface to the 65C02 CPU.

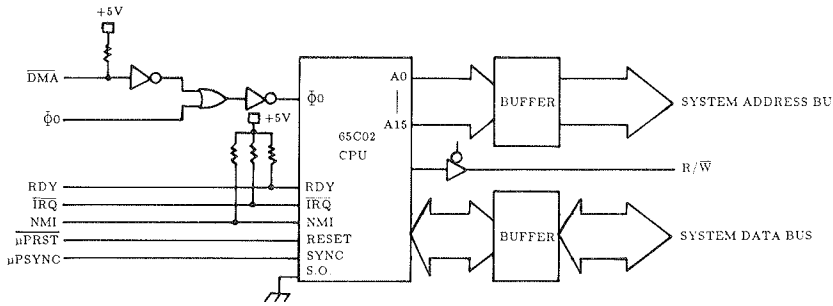


Fig 8.1 Interface to 65C02 CPU

For details of the 65C02 CPU, please refer to Appendix A of this manual.

8.3 ROM

The computer has a total of 32K bytes ROM. The ROM may consist of a 32K x 8 bits ROM/EPROM or two 16K x 8 bits EPROMs.

In factory, correct configuration has been made so that the ROM/EPROM(s) can function properly. You may find four bow-ties near the ROM/EPROM(s) on the component side of the PCB. Two of them are normally open and two are normally shorted. These configurations are for one 32K x 8 bits ROM/EPROM. If two 16K x 8 bits EPROMs are used, the triangular bow-ties should be cut open and the round bow-ties should be shorted. The connection diagram is shown in Fig 8.2.

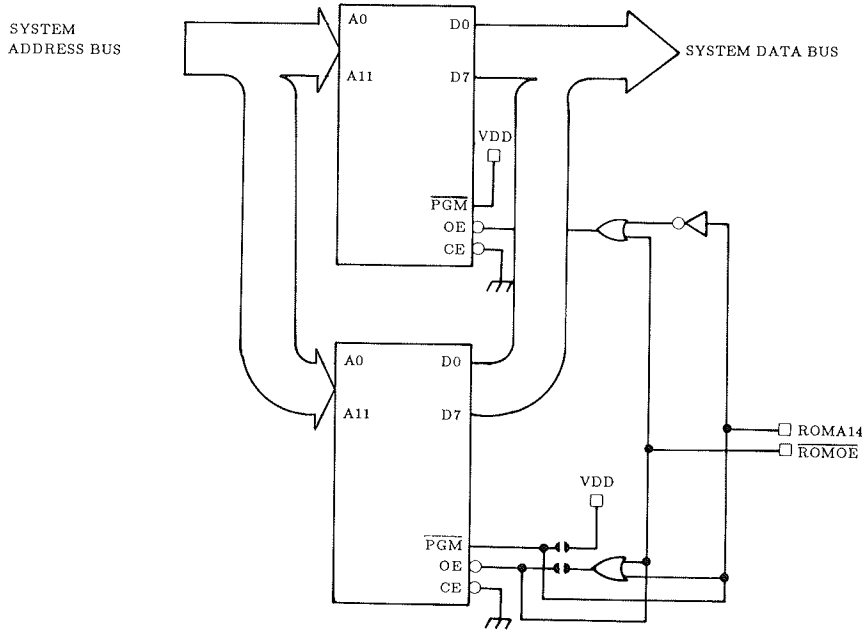


Fig 8.2 Schematic circuit diagram of ROM

You can find that the high order address of the ROM/EPROM(s) are connected to the ROM address bus from GA1 instead of the system address bus. This decoding scheme is to bank switch the ROM/EPROM(s).

The output enable, $\overline{\text{ROMOE}}$, signal to the ROM is generated inside GA1. This signal is controlled by the logic inside the gate array so that the high bank of RAM and ROM can share the common addresses from $D000 to $FFFF.

The timing diagram of the ROM/EPROM is shown in Fig. 8.3.

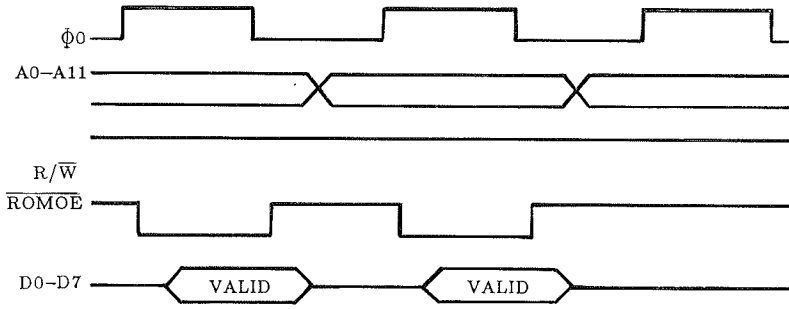


Fig 8.3 Timing diagram of ROM read

8.4 RAM

The computer has a total of 128K bytes RAM which splits into two 64K bytes banks, the main bank and the auxiliary bank.

Within a 64K bytes bank, the RAM can be subdivided into several sub-banks as shown in Fig. 8.4.

The bank-switching of RAM and ROM is controlled by the logic inside GA1. The main bank and the auxiliary bank of RAM can also be bank-switched through soft-switches. You can find more details on the bank-switching features in subsequent sections.

The RAMs are enabled by the $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals. The main bank and the auxiliary bank of RAM share the common $\overline{\text{RAS}}$ signal. Their $\overline{\text{CAS}}$ signals differ so that we can obtain bank switching. The timing diagram of the RAMs can be found in Fig. 8.5.

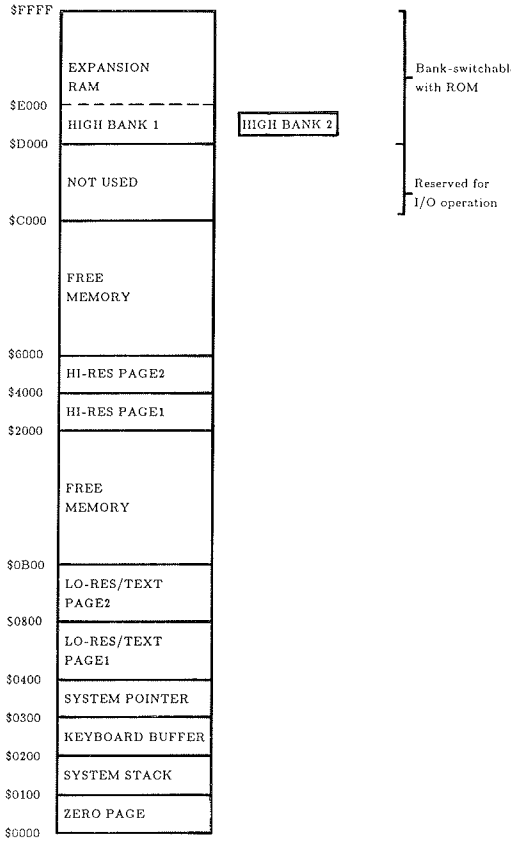


Fig 8.4 Memory map of RAM

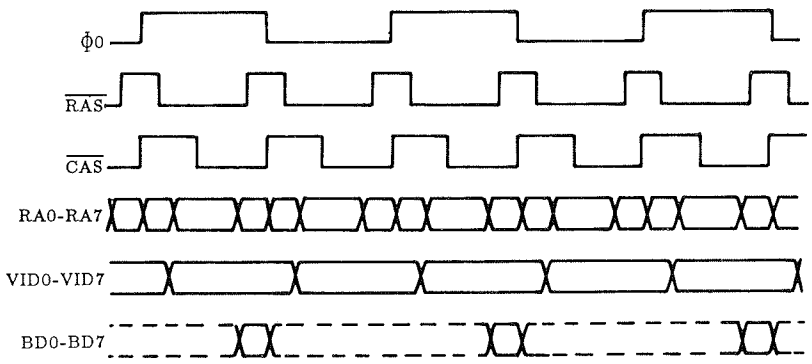


Fig 8.5 Timing diagram of RAM

8.5 ADDRESS DECODING AND BANK SWITCHING

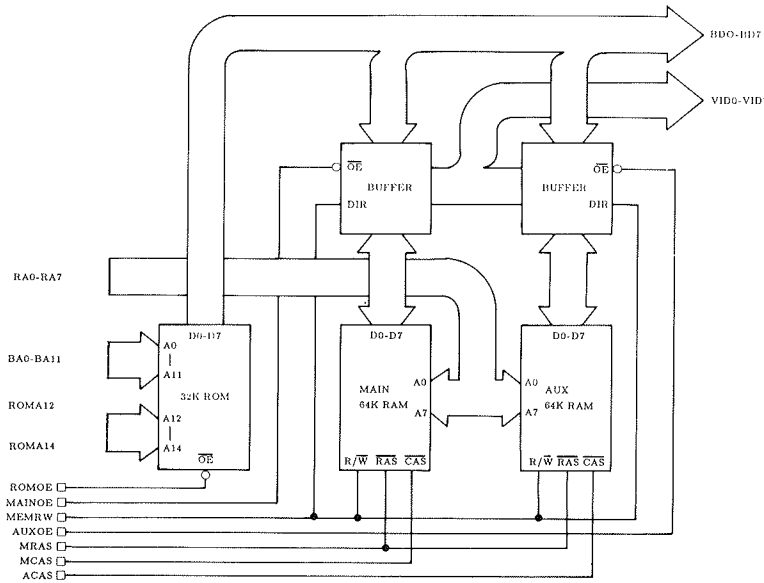


Fig 8.6 Memory block diagram

The address decoding of the system memory is completely inside GA1. The user can bank-switch memories by referencing certain soft-switches. Because I/O firmwares of different slots share the same addresses from $C800 to $CFFF for their I/O subroutines, the high order addresses of the ROM must be decoded to bank-switch suitable routines into addresses from $C800 to $CFFF.

Besides, the addresses from $D000 to $FFFF are shared by the ROM/EPROM(s) and both main and auxiliary 64K RAM. Different enable signals are generated, inside GA1, to enable either the ROM/EPROM(s) or the high bank RAMs. These signals include $\overline{\text{ROMOE}}$, $\overline{\text{MAINOE}}$, $\overline{\text{AUXOE}}$, $\overline{\text{ACAS}}$ and $\overline{\text{MCAS}}$.

$\overline{\text{RAS}}$ is common to both auxiliary and main RAM so that both RAM can be refreshed simultaneously through the video circuit.

All the enable signals of RAM and ROM/EPROM(s) are clocked inside GA1 with $\Phi 0$ so that data on system bus is only valid during $\Phi 0$ being high.

With $\Phi 0$ being low, the video circuit takes control of the RAM and latches data from RAM into the video signal generator.

Therefore any device attempting to take control of RAM during $\Phi 0$ low would hang the computer.

Details of soft-switches controlling bank-switching of RAMs and ROM/EPROM(s) can be found in other chapters.

8.6 TIMING SIGNALS GENERATION

The primary source of the system timing signals is a crystal oscillator as shown in Fig 8.7.

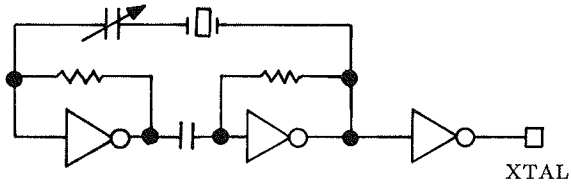


Fig. 8.7 Schematic diagram of the crystal oscillator

The signal, XTAL, generated is for the chroma phase primarily. The system's F14M signal is then generated by different circuits in different models.

8.6.1 Circuit of the phase-locked loop (PLL)

In NTSC models and PAL models, the system clock signal F14M is generated by a PLL circuit. The circuit composed of an analog PLL IC, NE564, and two dividers to generate the F14M.

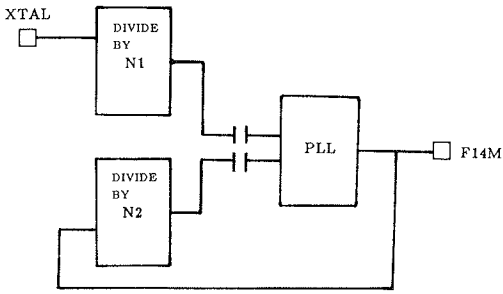


Fig. 8.8 Block diagram of the F14M circuit

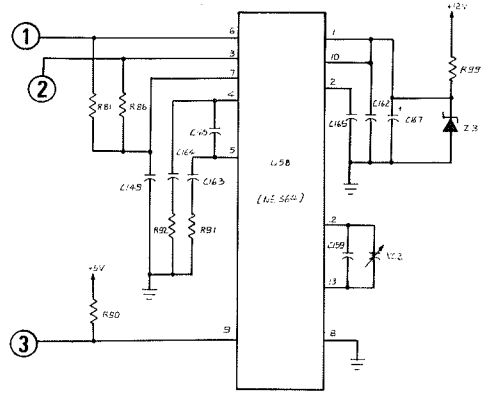


Fig 8.9 The schematic of NE564 (PLL)

8.6.2 NTSC models

In NTSC models, the modulo-56 divider and modulo-57 divider are added to the PLL to generate the F14M signal (14.0669MHz) from XTAL (14.31818MHz). Four 4-bit binary counters 74HCT163 are used for the dividers.

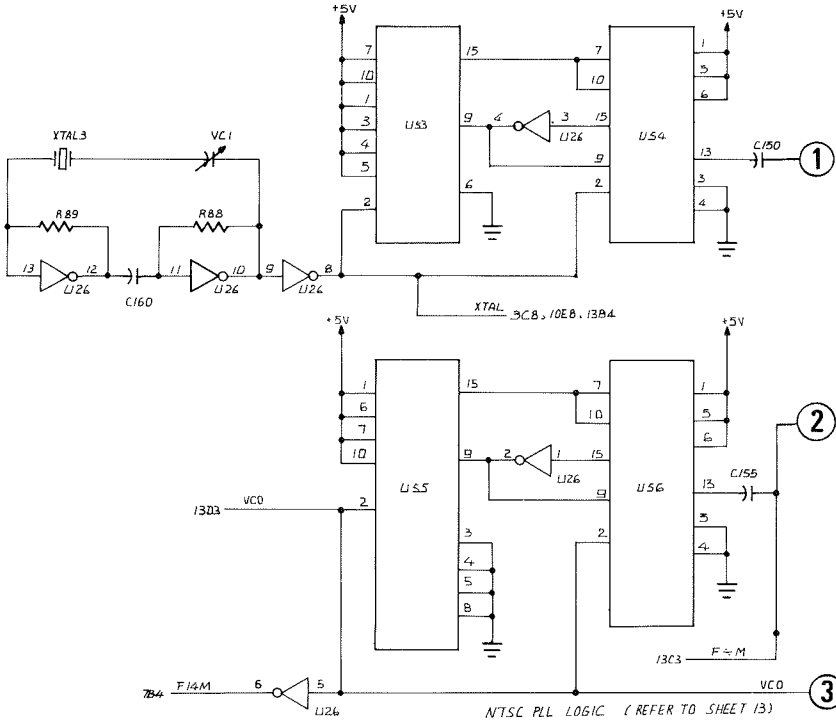


Fig 8.10(a) The schematic of dividers for NTSC

8.6.3 PAL models

In PAL models, two 4-bit binary counters 74HCT163 are used for the modulo-4 dividers and modulo-5 divider. The F14M signal of PAL model is 14.1875MHz.

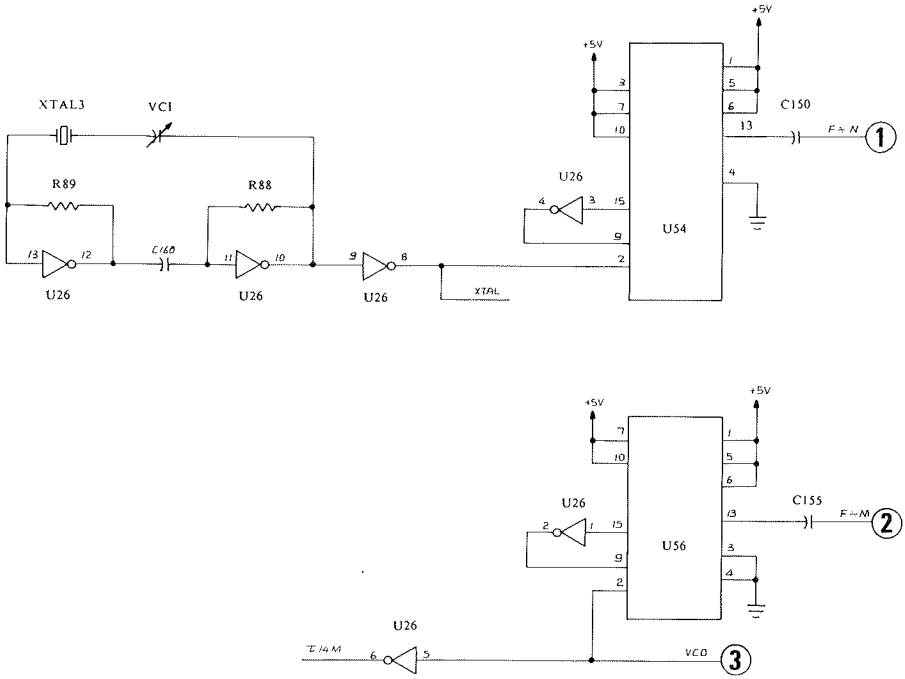


Fig 8.10(b) The schematic of the dividers for PAL

8.6.4 Frequency fine tuning

Under extreme operating conditions, e.g. high temperature, fine tuning of PLL circuit is necessary if shaking characters are observed on the screen. The top cabinet can be removed. Underneath the keyboard, a trimmer head can be found through the hole of the metal shield. You can fine-tune the trimmer to get a stable picture. However, take great care in doing so. The trimmer is trimmed with precise equipment before ex-factory. If the trimmer is offset too much from its desired position, the picture quality may get worse.

8.7 RESET CIRCUIT

The RESET circuit consists of two parts: the CPU reset circuit and the gate array reset circuit. The two circuits are described as follows:

8.7.1 The CPU reset circuit

The schematic circuit diagram is shown in Fig 8.11.

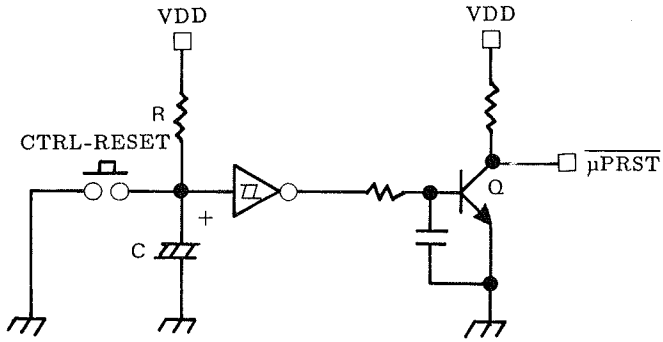


Fig 8.11 Schematic diagram of CPU reset circuit

On power-up or when CTRL-RESET is pressed, C is discharged. The Schmitt-tiggered inverter output is turned high and the transister, Q is turned on. $\overline{\mu PRST}$ signal is then held low until the reset condition is removed. C is then charged up through R until a "high" voltage level is reached. The output of the inverter then turns low and Q is turned off. $\overline{\mu PRST}$ signal then turns high and the reset procedure is finished.

8.7.2 The gate array reset circuit

The schematic circuit diagram is shown in Fig 8.12.

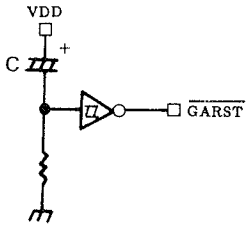


Fig 8.12 Schematic diagram of gate array reset circuit

The circuit is effective on power-up only.

On power-up, C starts to be charged up. The Schmitt-triggered inverter output is initially low. When C is charged up to a "high" voltage level, the Schmitt-triggered inverter output then turns high and the reset procedure is finished.

8.8 KEYBOARD

The keyboard of the computer generates encoded ASCII codes. The keyboard circuit consists of a keyboard encoder, KB-3600-PRO, a ROM/EPROM and a keyboard matrix.

When a key is pressed, the ASCII code generated is then latched and its MSB is set. As the location is read, the MSB is cleared to indicate that the key has already been read.

When a key is held down for a period, say 1 second, the keyboard encoder will generate that code continuously until the key is released. All keys on the keyboard have this auto-repeating feature except the RESET key and the two keys "OPEN TRIANGLE" and "CLOSE TRIANGLE".

The schematic diagram of the keyboard circuit is shown in Fig 8.13.

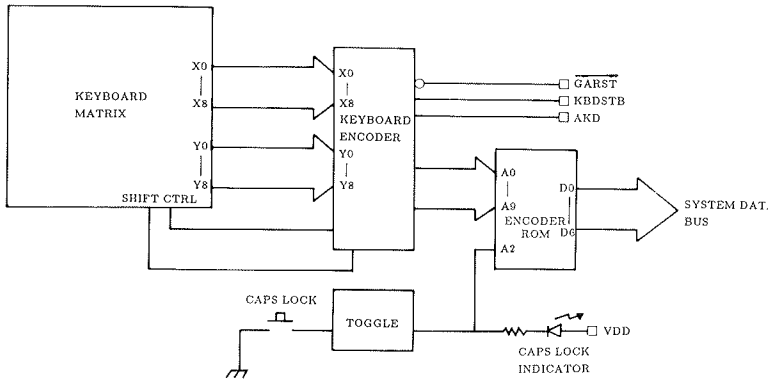
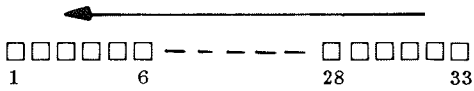


Fig 8.13 Schematic diagram of keyboard circuit

The keyboard matrix is connected to the main system board through a flat cable jumper. The pin-out diagram of the jumper and signal descriptions are shown in TABLE 8.1.



KEYBOARD SIDE

Pin no.	Signal name	Descriptions
1	CLOSE	connected directly to SW0 of hand control
2	OPEN	connected directly to SW1 of hand control
3	CAPS LOCK	toggles the caps lock switch
4	SHIFT	connected directly to SHIFT key of keyboard
5	CTRL	connected directly to CTRL key of keyboard
6	RST	connected directly to RESET key of keyboard
7	GND	
8--16	X7--X0	connected directly to x-coordinate of keyboard matrix
17--26	Y9--Y0	connected directly to y-coordinate of keyboard matrix
27	POWER	connected to the POWER LED
28	CAPSLED	connected to the CAPS LOCK LED
29	DRV	connected to the DRIVE LED
30	VCC	+5V
31	80COL	connected to the 40/80 switch

32	P/S	connected to the PARALLEL/SERIAL switch
33	COLOR	connected to the COLOR/MONO switch

Table 8.1 Keyboard signals descriptions

8.9 SPEAKER

The computer has a 2 1/4" speaker. Besides, an ear-phone jack is also available which allows you to use ear-phone instead of the internal speaker to generate sound.

The audio circuit consists of a transistor booster and several other elements as shown in Fig. 8.14.

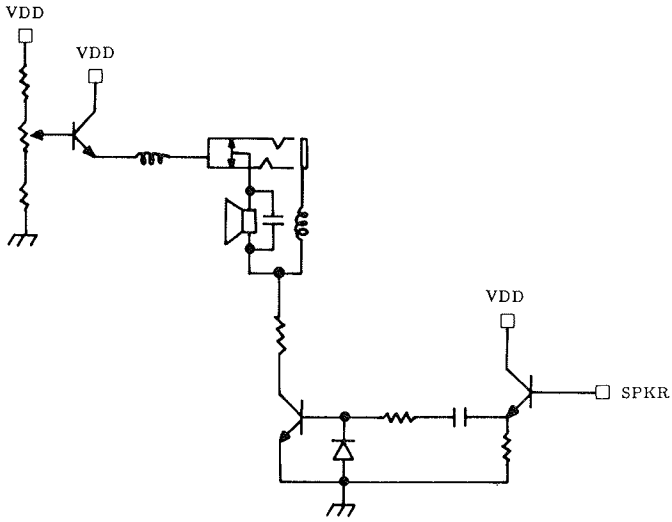


Fig 8.14 Schematic diagram of speaker circuit

8.10 THE VIDEO CIRCUIT

The computer can display both 40-column and 80-column text. Besides, it can handle up to 560 x 192 dots graphic screen.

In 40-column or normal graphic mode, VID7M is a 7MHz signal to gate with the F14M to the 74HCT166.

In 80-column or double resolution mode, VID7M is pulled low inside GA1.

All video signals are generated inside GA1. These signals include:

1. HSYNC, VSYNC that control the synchronisation of the monitor.

2. C0, C1, C2, C3 that contain the color information.

3. $\overline{374OE}$ that latches the video data from RAM into the shift register, U52.

4. $\overline{LDPS}$ loads the shift register on active low.

5. **VIDEO DATA output from the shift register.**

6. VID6, VID7 are the 7-th and the 8-th bit of the video data.

7. VROM0, VROM1, VROM2, VROM9 and VROM10 are the address lines of the character generator.

8. GR indicates the graphic modes on active high.

9. COLOR is an input to GA1. It indicates that the computer is in color mode, and is active high.

The block diagram of the video circuit is shown in Fig 8.15.

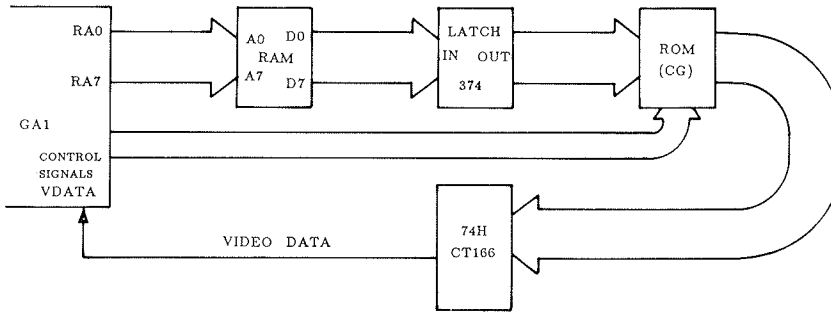


Fig 8.15 Block diagram of the video circuit

8.11 PARALLEL PRINTER

The computer has built-in parallel printer interface which is Centronics compatible. The firmware to control the parallel printer is built in the system ROM/EPROM(s). The schematic circuit diagram is shown in Fig 8.16.

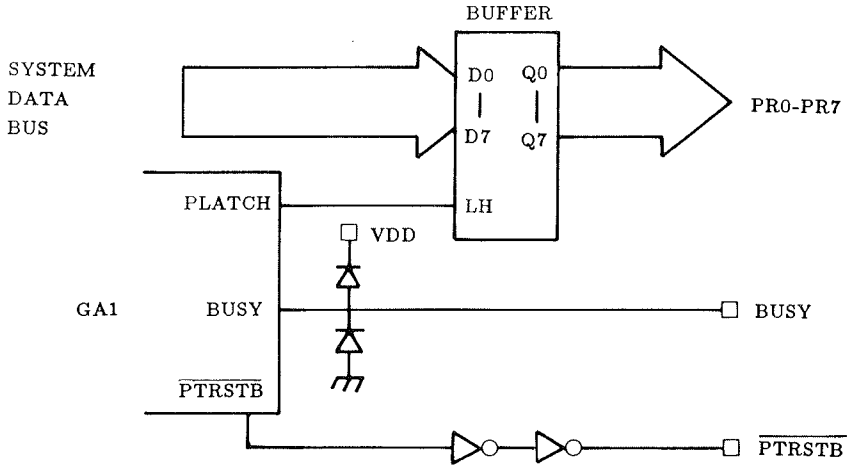


Fig 8.16 Schematic diagram of parallel printer circuit

The BUSY signal from the printer indicates that the printer is not to accept data input. It is active high. The $\overline{\text{PTRSTB}}$ signal to the printer indicates that a byte of data has already been sent to the printer bus. It is active low. The timing between the two signals is shown in Fig 8.17.

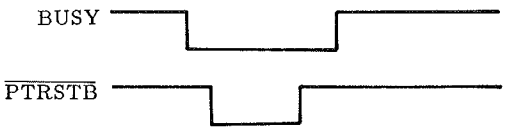


Fig 8.17 Timing of BUSY and $\overline{\text{PTRSTB}}$

8.12 SERIAL I/O PORTS

The computer has two serial I/O ports, PORT 1 and PORT 2 that are RS232 compatible. One 6551/65C51 ACIA is used for each port.

The firmware of the serial ports are built in the system ROM/EPROM(s).

The schematic diagram is shown in Fig 8.18.

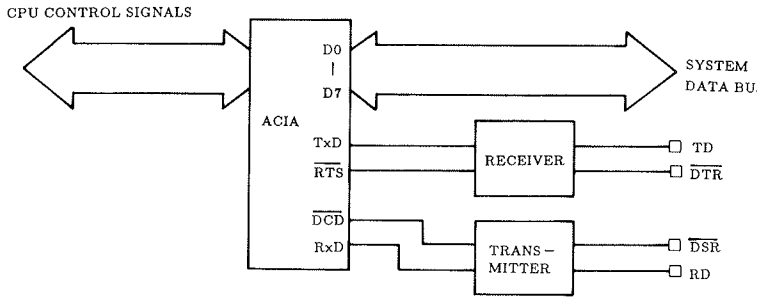


Fig 8.18 Schematic diagram of serial ports

The ACIA has its own baud rate generator and necessary circuit to interface with the CPU.

One point must be noted when using Serial Port 1. If a parallel printer is used, the PARALLEL/SERIAL switch on the front panel must be set to "PARALLEL". If a serial printer is used, the switch must be set to "SERIAL".

8.13 MOUSE

The computer has a built-in interface for the mouse. The details of the operation of the mouse can be found in the Chapter 6.

When the mouse is moved, interrupts to the CPU are generated by the XINT or YINT signals. Then the directions of movement is checked and the appropriate screen holes are updated if necessary.

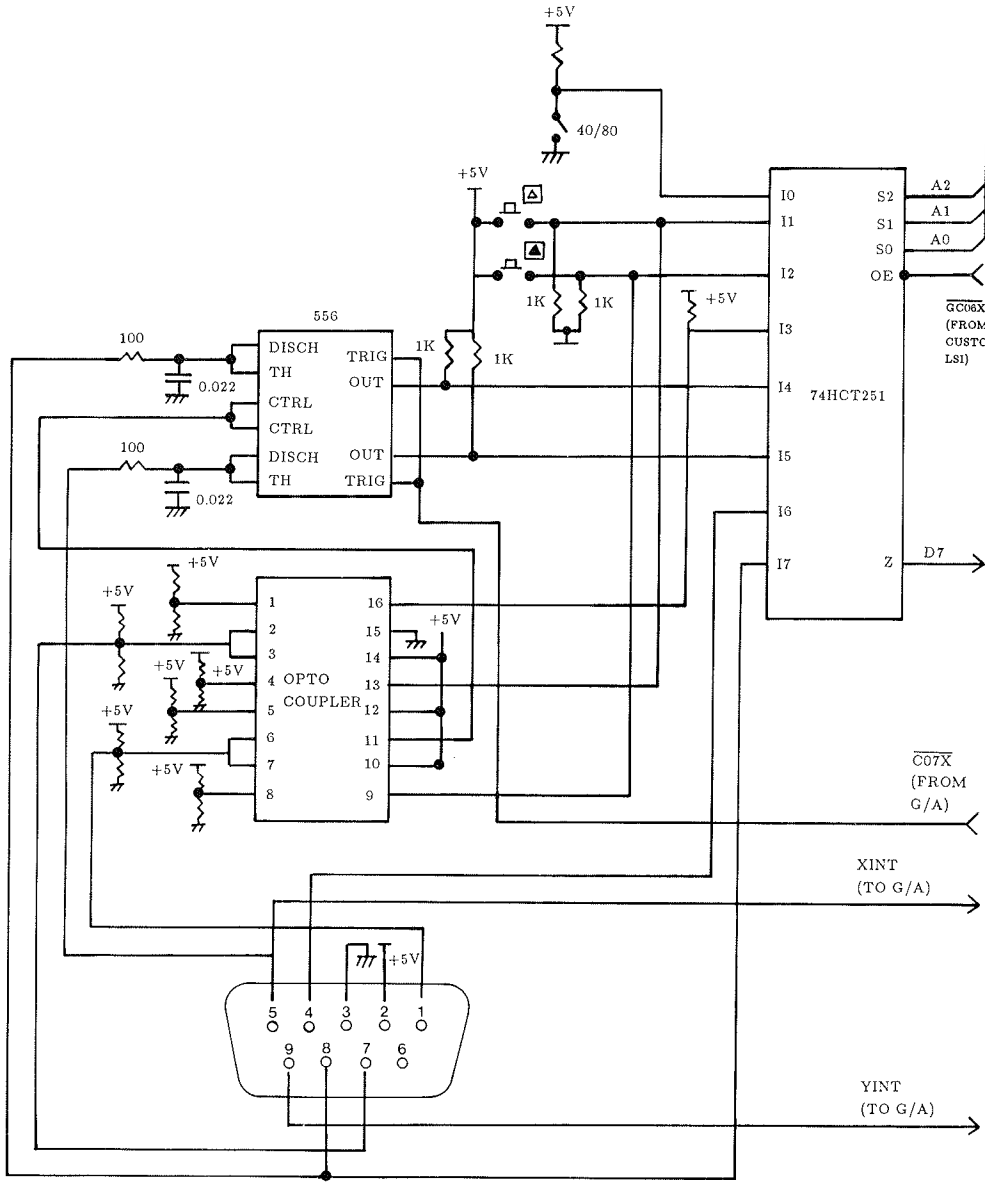


Fig 8.19 Schematic diagram of mouse and joystick

8.14 HAND CONTROL/JOYSTICK

The computer also supports hand controls/joystick. The circuit consists mainly of two timers. Each timer is responsible for one paddle.

When a paddle is rotated, the corresponding resistance of the variable resister changes, the oscillator frequency is changed and we get a new value for that paddle.

The schematic circuit diagram of the hand control/joystick interface is shown in Fig 8.19.

8.15 DISK DRIVE

The computer supports two 5 1/4" floppy disk drives, one built-in and one external. The necessary signals are generated inside GA2 and the supporting firmware is stored in system ROM/EPROM(s).

The hardware uses soft-switches $C0E0 to $C0EF to control different operations of the drives. Details on usage of these soft-switches can be found in chapter 6.

The schematic diagram of the disk drive interface is shown in Fig 8.20.

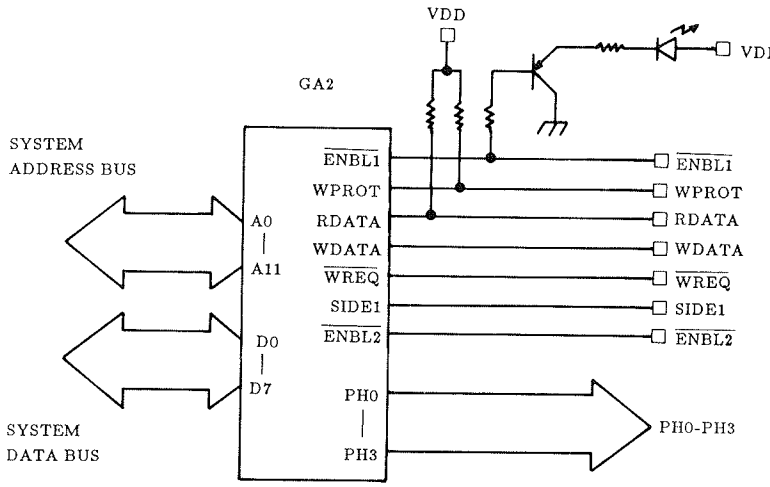


Fig 8.20 Schematic diagram of disk drive interface

The following is a description of the signals used by the disk drive interface:

1. PH0 to PH3 are signals controlling the four phases of the stepper motor and they are active high.

2. WPROT indicates a diskette being write-protected and it is active high.

3. RDATA is the serial data read from the diskette.

4. WDATA is the serial data written onto the diskette.

5. $\overline{\text{WREQ}}$ signals the drive that a write-operation is to be performed.

6. SIDE1 is a signal to indicate which side the diskette is to be accessed and it is active high. It is toggled by writing to locations from $C600 to $C6FF.

7. $\overline{\text{ENBL1}}$ and $\overline{\text{ENBL2}}$ are two enable signals to select one of the two disk drives to be accessed. The two signals are active low.

The internal disk drive is connected to the system board through a 10 pins x 2 jumper. The pin-out and the signal names are listed in TABLE 8.2.

Pin no.	Signal name	Pin no.	Signal name
1	PH0	2	GND
3	PH1	4	GND
5	PH2	6	GND
7	PH3	8	GND
9	$\overline{\text{WREQ}}$	10	SIDE1
11	+5V	12	+5V
13	$\overline{\text{ENB1}}$	14	+12V
15	RDATA	16	+12V
17	WDATA	18	+12V
19	WPROT	20	+12V

Table 8.2 Pin-out diagram of the connector for internal disk drive

There are some instances when the speed of the disk drive is critical to an application program and it may be necessary to adjust the speed of the disk drive. To adjust the speed of the internal disk drive, you first find a small hole, labelled "DRIVE SPEED ADJUST", near the door of drive on the bottom of the computer. You can then adjust the speed by turning a trimmer inside the hole with a small screw driver. You may read the speed of the disk drive with a program like DRIVE SPEED inside the FILER + disk. The normal speed of the disk drive is 200 mS per revolution, +/- 2 mS.

8.16 THE POWER SUPPLY

The power supply of the computer consists of two parts: the external AC/DC adapter and the internal regulator box.

8.16.1 AC/DC adapter

It is a device to transform the main AC into 17V DC. The specifications of the adapter are listed below:

normal voltage = 17.0V
max tolerance = +/- 5%
max current = 1.8A

8.16.2 The regulator box

The regulator works in switching mode and outputs 3 levels of voltages, including +12V, +5V and -12V. The specifications of the switching supply are listed below:

norminal voltage	maximum tolerance
+5V	+/- 5%
+12V	+/- 5%
-12V	+/- 10%

and the current rating is:

norminal voltage	maximum current
+5V	1.5A
+12V	1.0A
-12V	0.1A

The circuit diagram of the power supply is shown in Fig 8.21.

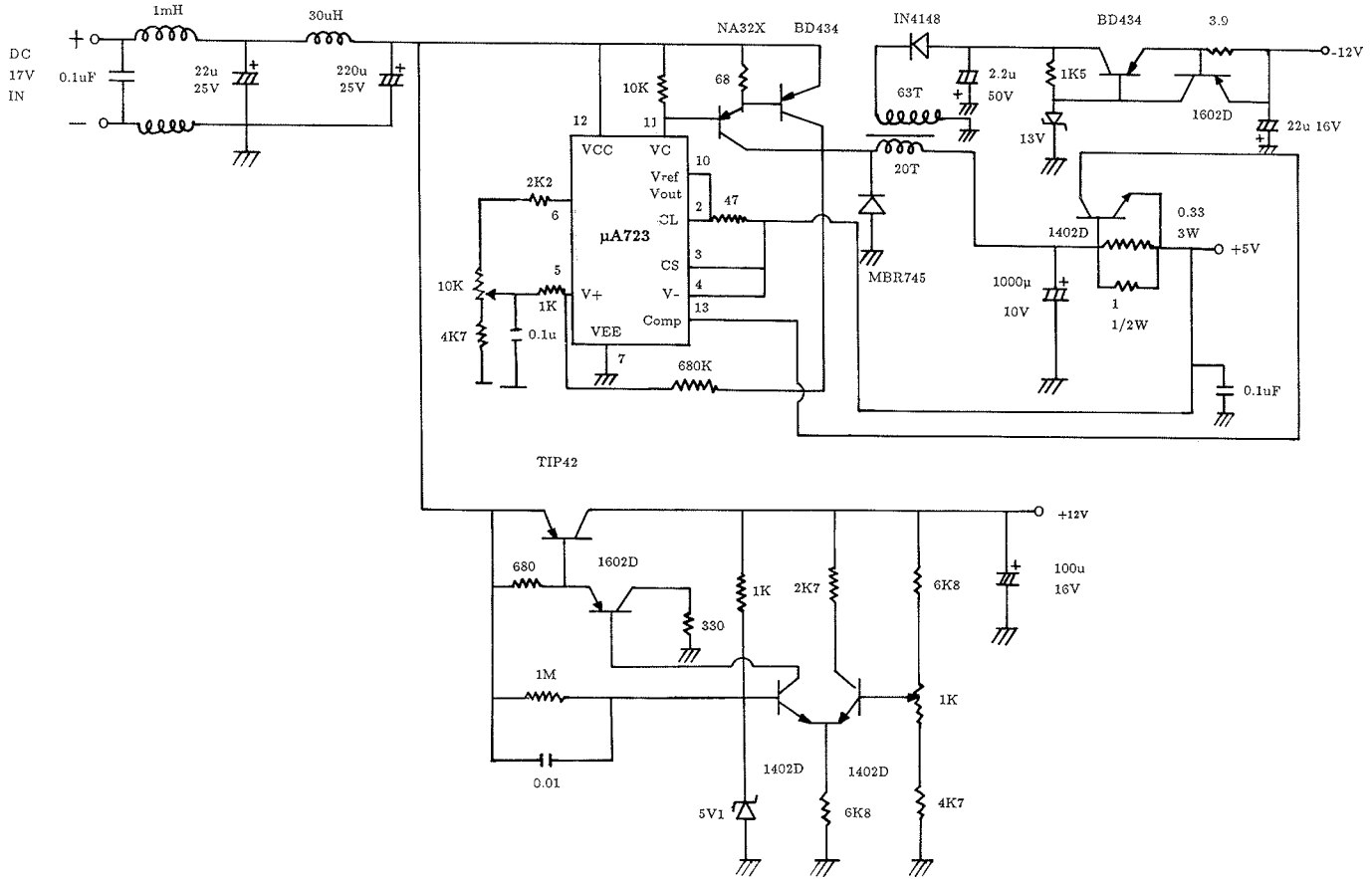
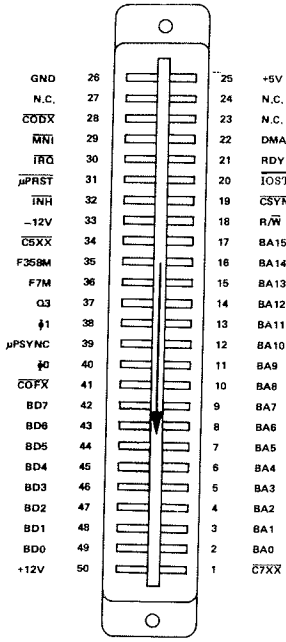


Fig 8.21 Schematic diagram of power supply

8.17 EXPANSION SLOTS

There are two expansion connectors on the main board. One is a 25 pins x 2 edge connector and the other one is a 15 pins x 2 jumper connector.

The 25 pins x 2 edge connector is Apple® compatible with several modifications. It is used for installation of other specific expansion modules. The pin-out diagram and signals description is shown in TABLE 8.3.



Pin no	Signal name	Descriptions
1	$\overline{C7XX}$	active low when I/O locations $C700 to $C7FF are accessed
2--17	BA0--BA15	system address bus
18	R/$\overline{W}$	system Read/$\overline{Write}$ line
19	$\overline{CSYNC}$	composite sync of video signal

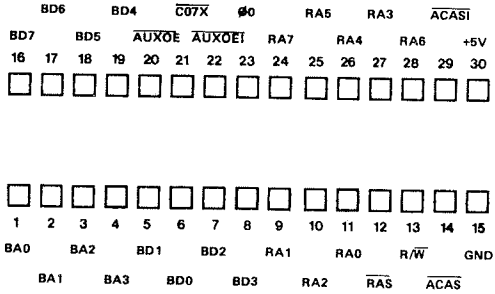
20	$\overline{\text{IOSTB}}$	active low when I/O locations $C800 to $CFFF are accessed
21	$\overline{\text{RDY}}$	connected directly to 65C02's $\overline{\text{RDY}}$
22	$\overline{\text{DMA}}$	system $\overline{\text{DMA}}$ and is active low
23.24	N.C.	
25	+5V	max. current is 500mA
26	GND	
27	N.C.	
28	$\overline{\text{CODX}}$	active low when locations $C0D0 to $C0DF are accessed
29	$\overline{\text{NMI}}$	connected directly to 65C02's $\overline{\text{NMI}}$
30	$\overline{\text{IRQ}}$	connected directly to 65C02's $\overline{\text{IRQ}}$
31	$\overline{\mu\text{PRST}}$	connected directly to 65C02's $\overline{\text{RESET}}$
32	$\overline{\text{INH}}$	active low to inhibit the system memory
33	-12V	max current is 200mA
34	$\overline{\text{C5XX}}$	active low when locations $C500 to $C5FF are accessed
35	F358M	system 3.58MHz clock
36	F7M	system 7MHz clock
37	Q3	2MHz asymmetrical clock
38	$\overline{\Phi 1}$	65C02's phase 1 clock
39	µPSYNC	connected directly to 65C02's SYNC

40	$\Phi 0$	65C02's phase 0 clock
41	$\overline{C0FX}$	active low when locations $C0F0 to $C0FF are accessed
42--49	BD7--BD0	system data bus
50	+12V	max current is 250mA

Table 8.3 Pin out diagram and signal descriptions of external expansion slot

There are two slots reserved for external modules if the internal Z80 module is not installed. If two cards are to be installed, an expansion box must be used. The main purpose of the expansion box is to split the single 25 pins x 2 socket into two 25 pins x 2 sockets so that two expansion modules may be connected.

The other expansion connector is a 15 pins x 2 jumper connector. Its main purpose is for memory expansion. The pin-out diagram and signal descriptions are shown in TABLE 8.4.

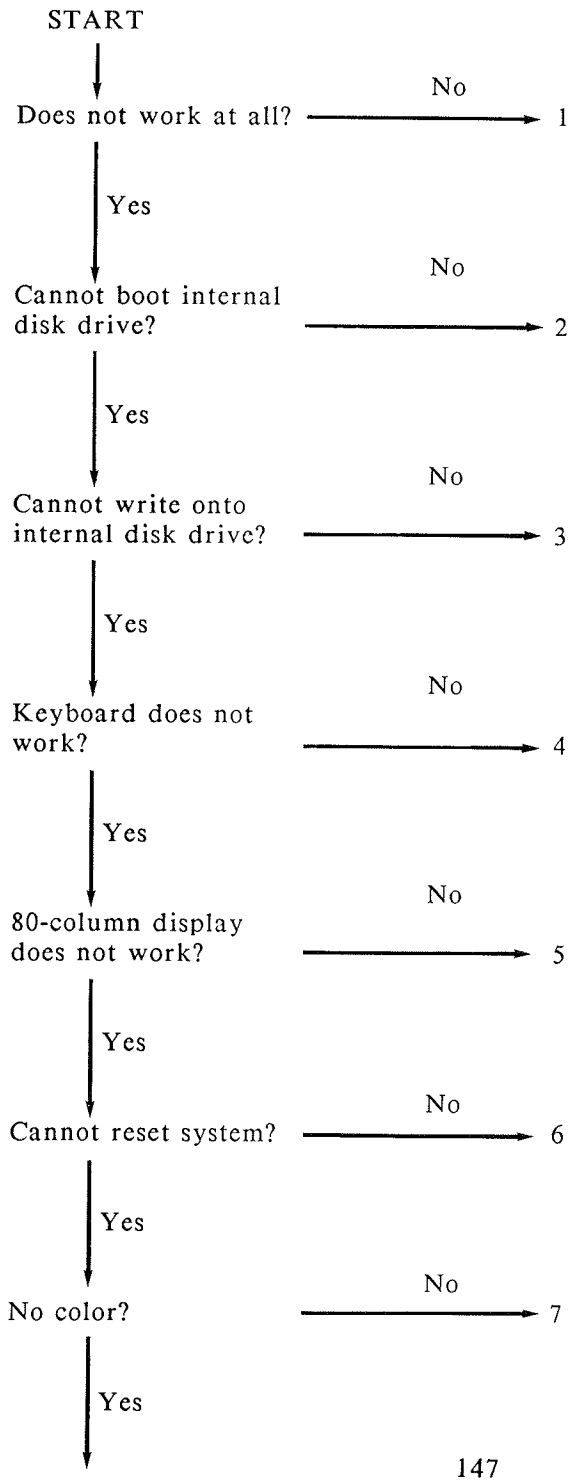


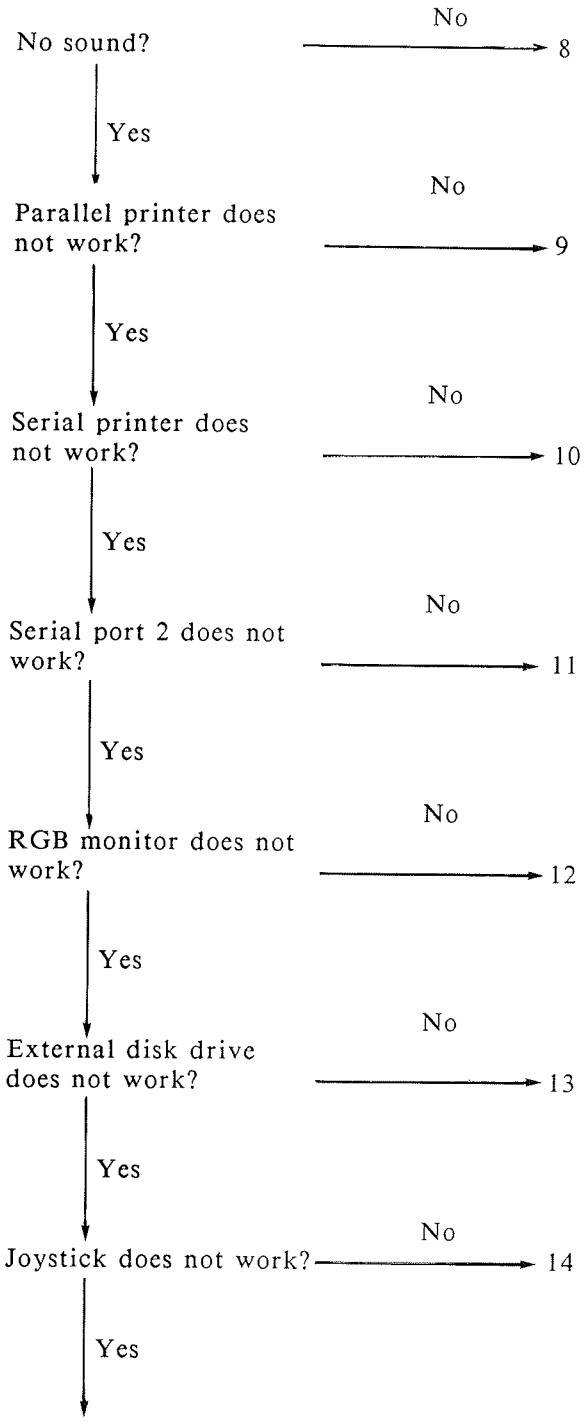
Pin no.	Signal name	Descriptions
1--4	BA0--BA3	system address bus
5--8	BD0--BD3	system data bus
9--11	RA0--RA2	RAM address bus
12	$\overline{RAS}$	RAM $\overline{RAS}$
13	R/$\overline{W}$	system Read/$\overline{Write}$ line
14	$\overline{ACAS}$	auxiliary RAM $\overline{CAS}$ (input to RAM expansion module)
15	GND	
16--19	BD7--BD4	system data bus

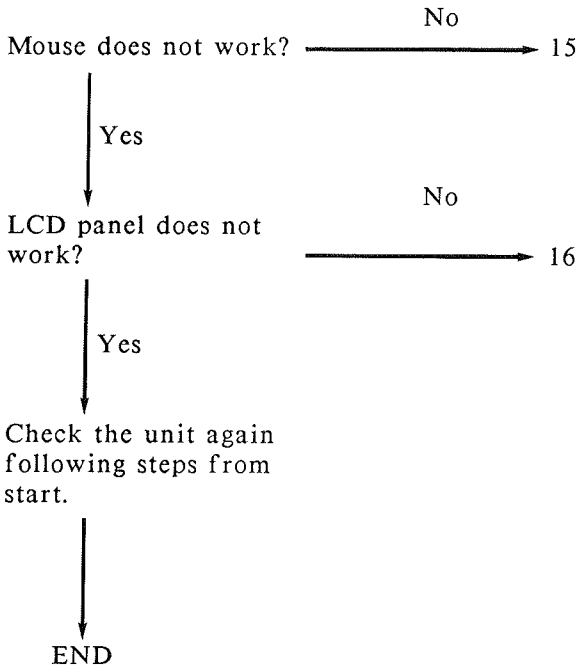
20	$\overline{\text{AUXOE}}$	auxiliary RAM buffer enable (input to module)
21	$\overline{\text{C07X}}$	active low when locations $C070 to $C07F are accessed
22	$\overline{\text{AUX0E1}}$	auxiliary RAM buffer enable (output from RAM expansion modul
23	$\Phi 0$	65C02's phase 0 clock
24--28	RA7--RA3	RAM address line
29	$\overline{\text{ACAS1}}$	auxiliary RAM $\overline{\text{CAS}}$ (output from RAM expansion module)
30	+5V	

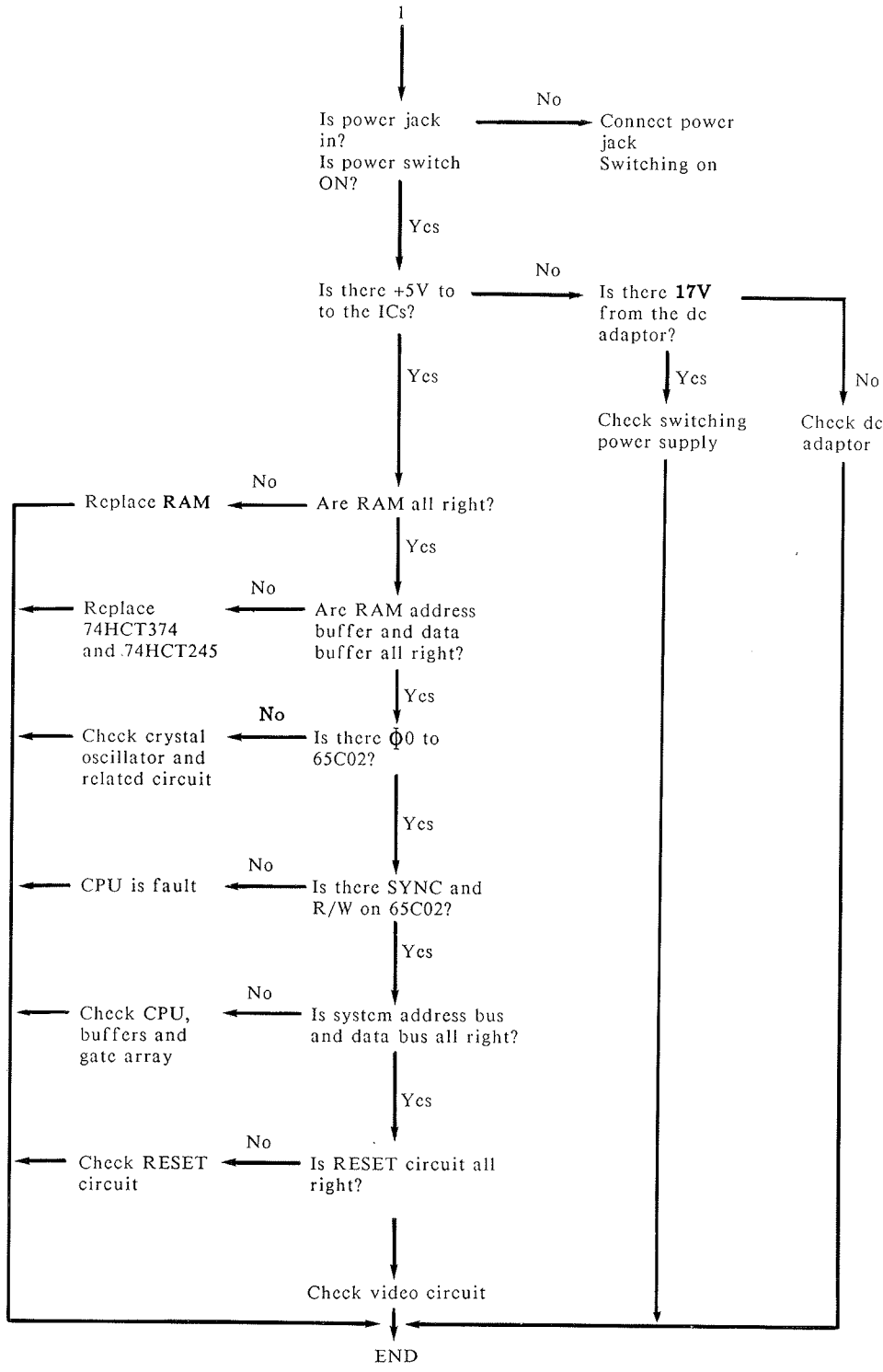
Table 8.4 Pin-out diagram and signal descriptions of internal memory expansion connector.

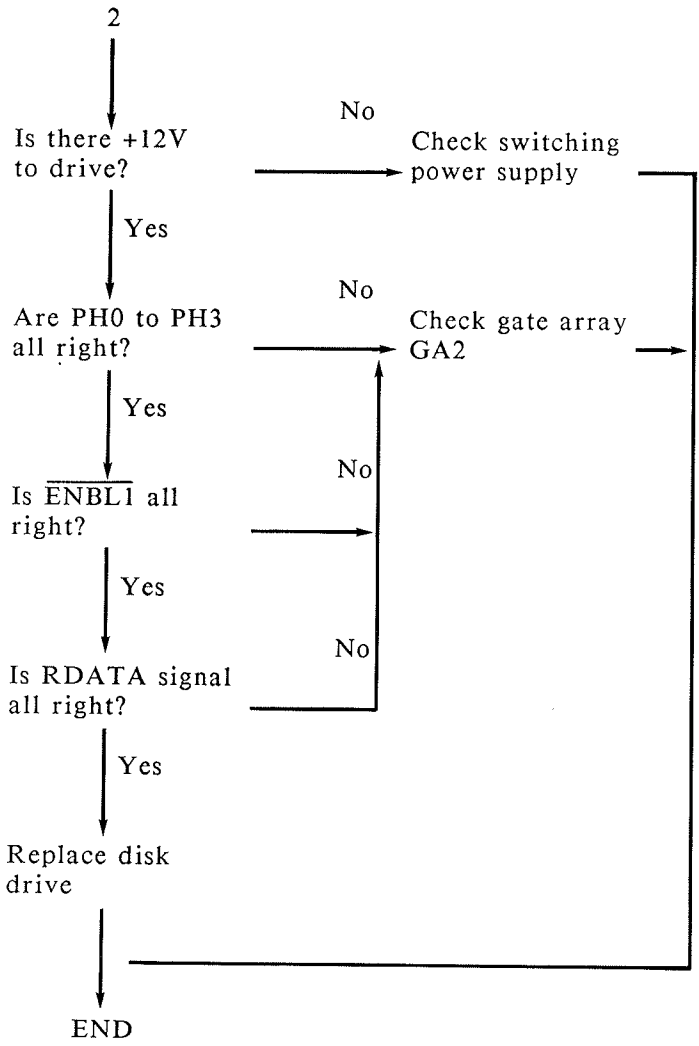
8.18 SERVICE FLOW CHART

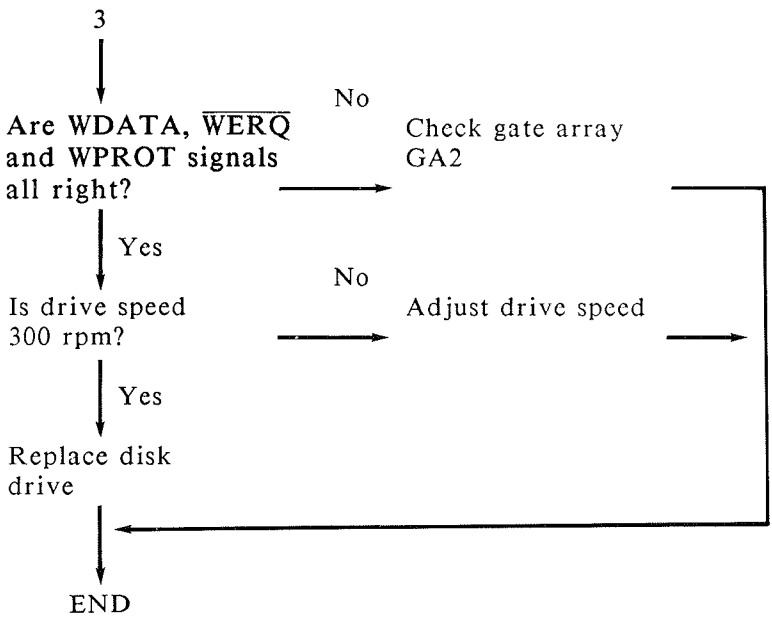


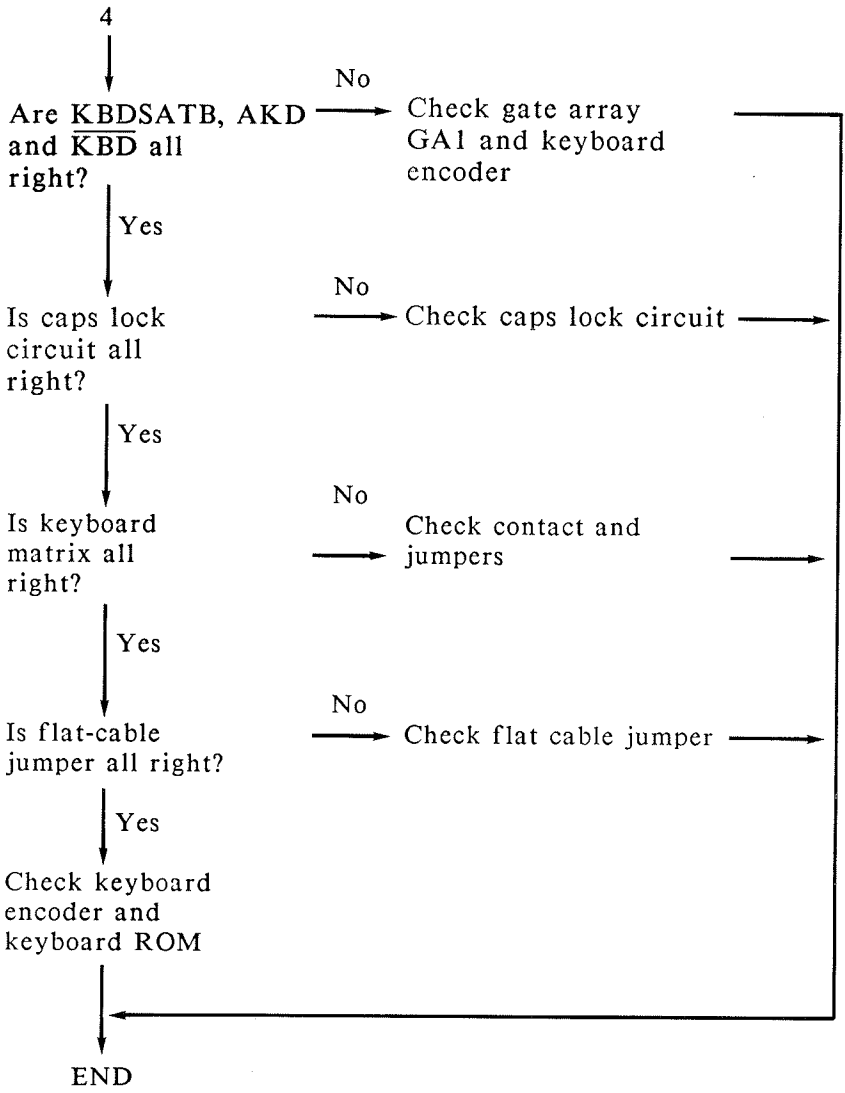

No sound?
No
8
Yes
Parallel printer does not work?
No
9
Yes
Serial printer does not work?
No
10
Yes
Serial port 2 does not work?
No
11
Yes
RGB monitor does not work?
No
12
Yes
External disk drive does not work?
No
13
Yes
Joystick does not work?
No
14
Yes

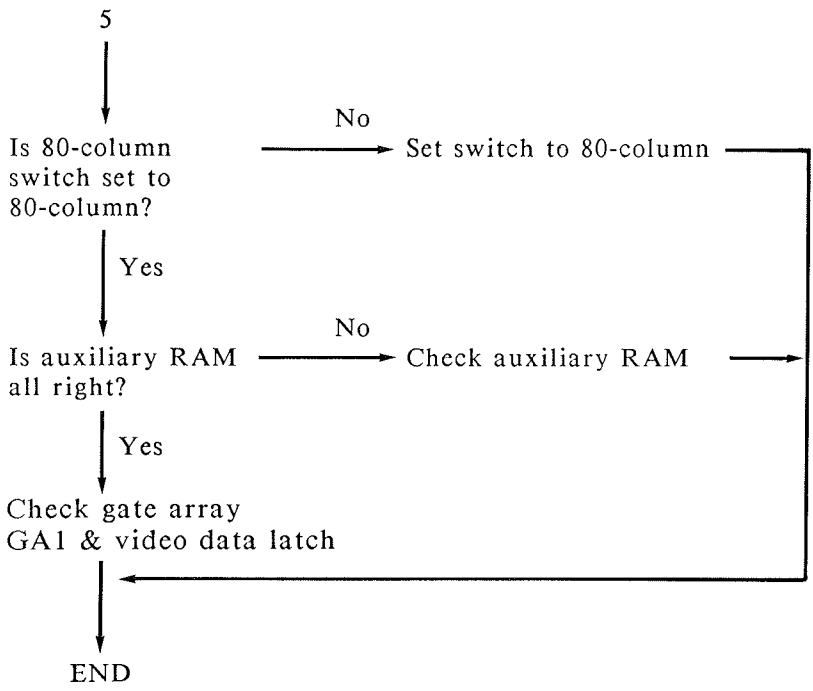

Mouse does not work?
No
15
Yes
LCD panel does not work?
No
16
Yes
Check the unit again following steps from start.
END

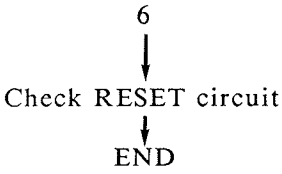

1
Is power jack in?
Is power switch ON?
No
Connect power jack
Switching on
Yes
Is there +5V to to the ICs?
No
Is there 17V from the dc adaptor?
Yes
No
Check switching power supply
Check dc adaptor
Yes
Are RAM all right?
No
Replace RAM
Yes
Are RAM address buffer and data buffer all right?
No
Replace 74HCT374 and 74HCT245
Yes
Is there Φ0 to 65C02?
No
Check crystal oscillator and related circuit
Yes
Is there SYNC and R/W on 65C02?
No
CPU is fault
Yes
Is system address bus and data bus all right?
No
Check CPU, buffers and gate array
Yes
Is RESET circuit all right?
No
Check RESET circuit
Check video circuit
END

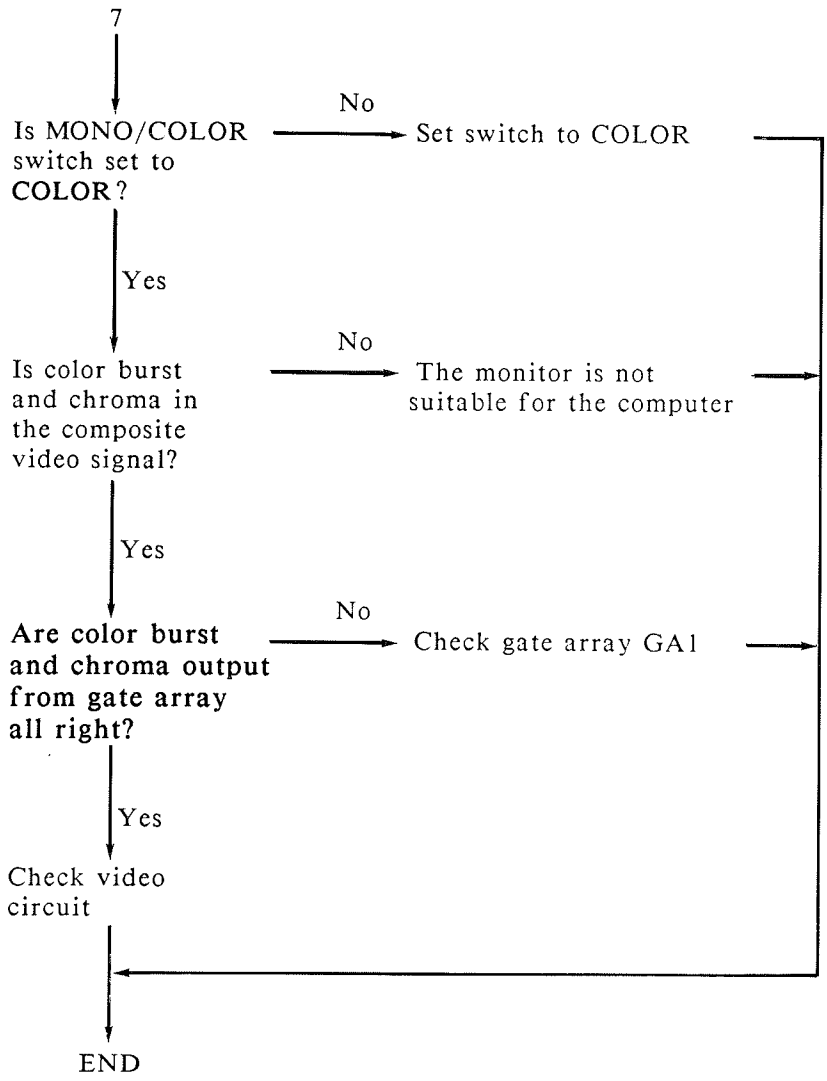

2
Is there +12V to drive?
No
Check switching power supply
Yes
Are PH0 to PH3 all right?
No
Check gate array GA2
Yes
Is $\overline{\text{ENBL1}}$ all right?
No
Yes
Is RDATA signal all right?
No
Yes
Replace disk drive
END

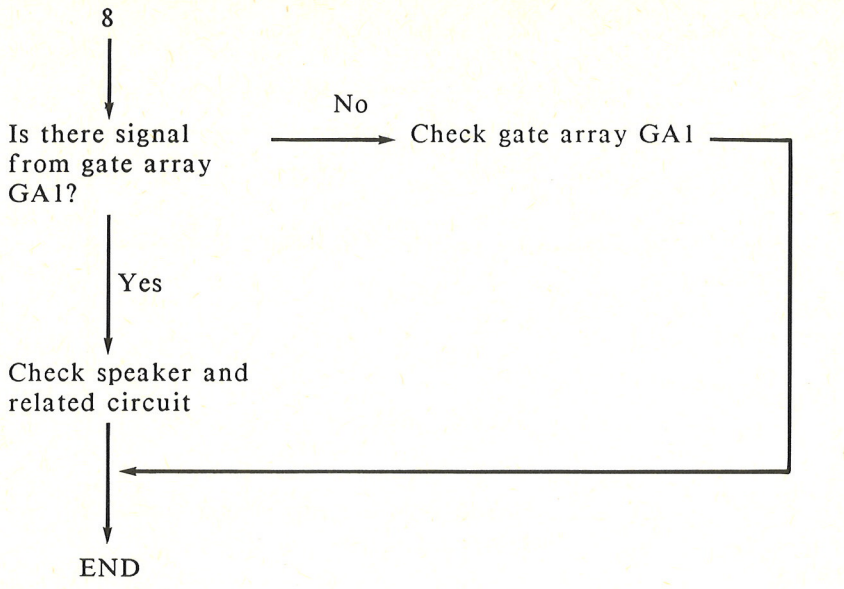

3
Are WDATA, WERQ and WPROT signals all right?
No
Check gate array GA2
Yes
Is drive speed 300 rpm?
No
Adjust drive speed
Yes
Replace disk drive
END

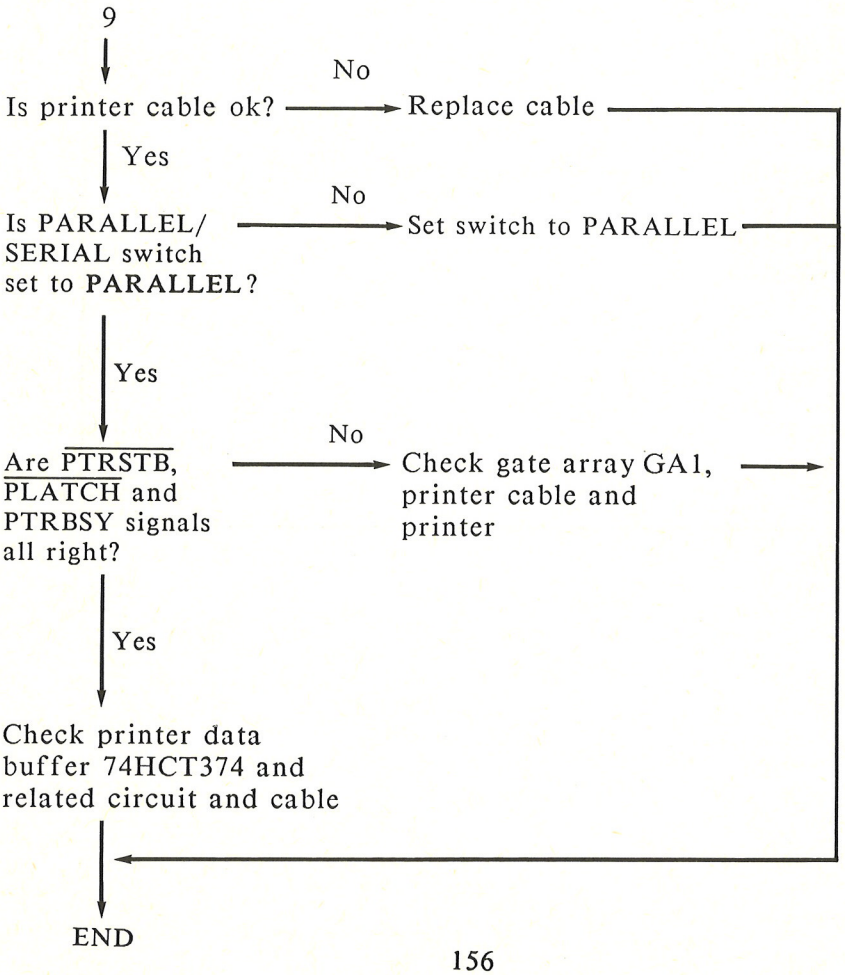

4
Are KBDSATB, AKD and KBD all right?
No
Check gate array GA1 and keyboard encoder
Yes
Is caps lock circuit all right?
No
Check caps lock circuit
Yes
Is keyboard matrix all right?
No
Check contact and jumpers
Yes
Is flat-cable jumper all right?
No
Check flat cable jumper
Yes
Check keyboard encoder and keyboard ROM
END

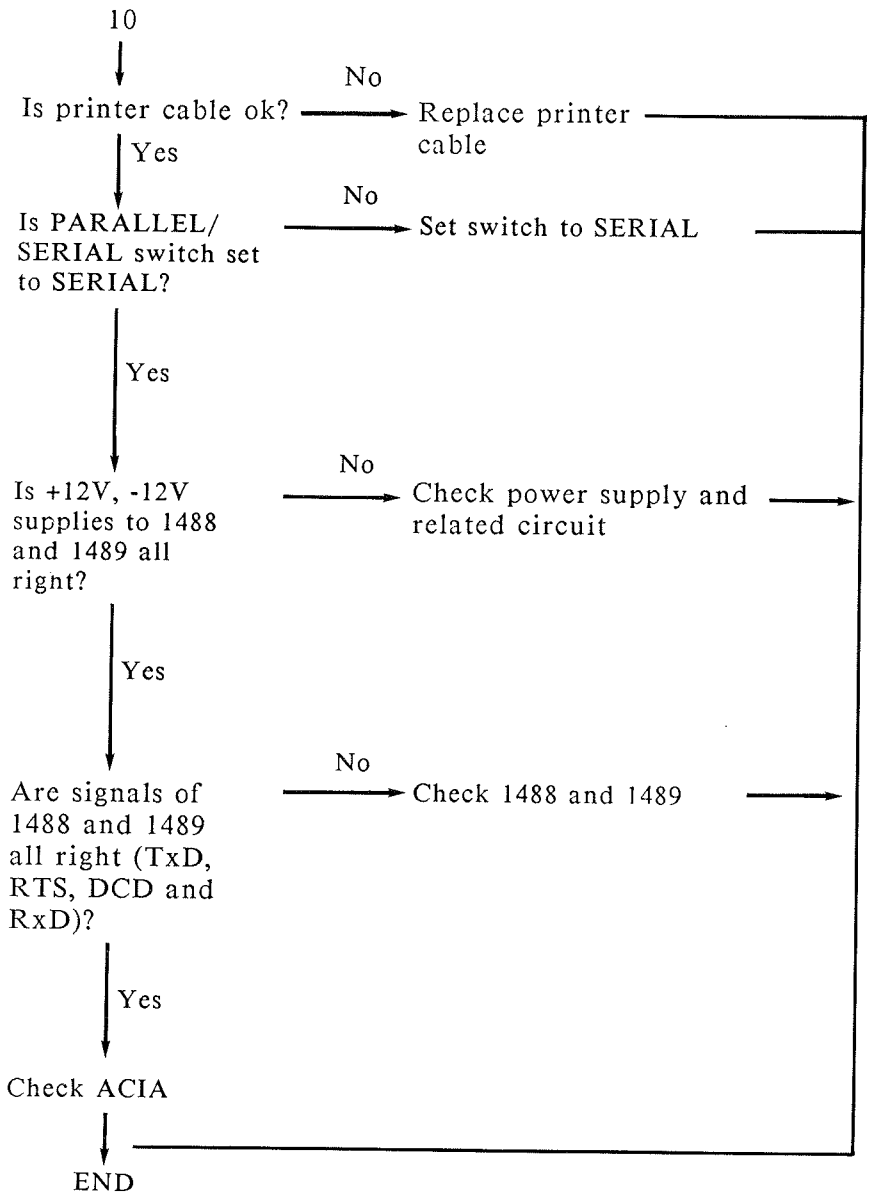

5
Is 80-column switch set to 80-column?
No
Set switch to 80-column
Yes
Is auxiliary RAM all right?
No
Check auxiliary RAM
Yes
Check gate array GA1 & video data latch
END

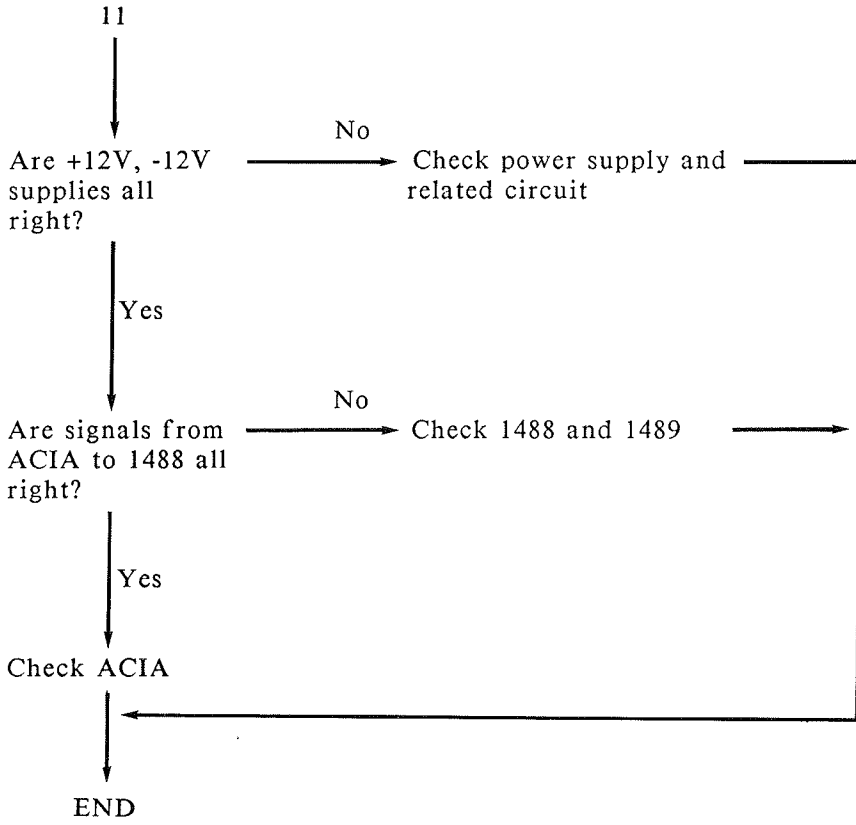

6
Check RESET circuit
END

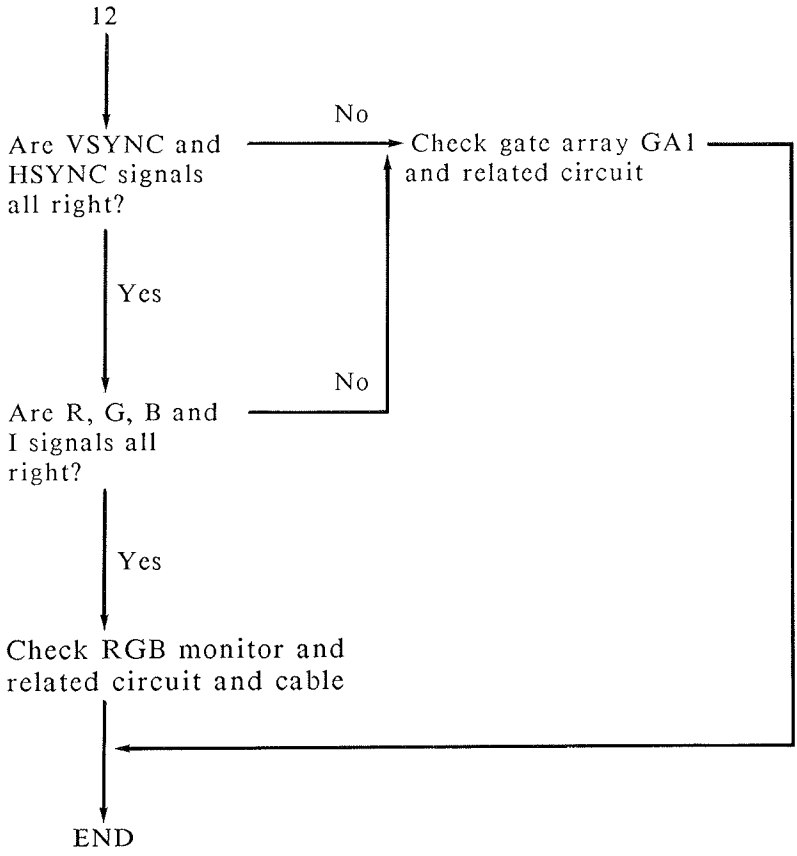

7
Is MONO/COLOR switch set to COLOR?
No
Set switch to COLOR
Yes
Is color burst and chroma in the composite video signal?
No
The monitor is not suitable for the computer
Yes
Are color burst and chroma output from gate array all right?
No
Check gate array GA1
Yes
Check video circuit
END

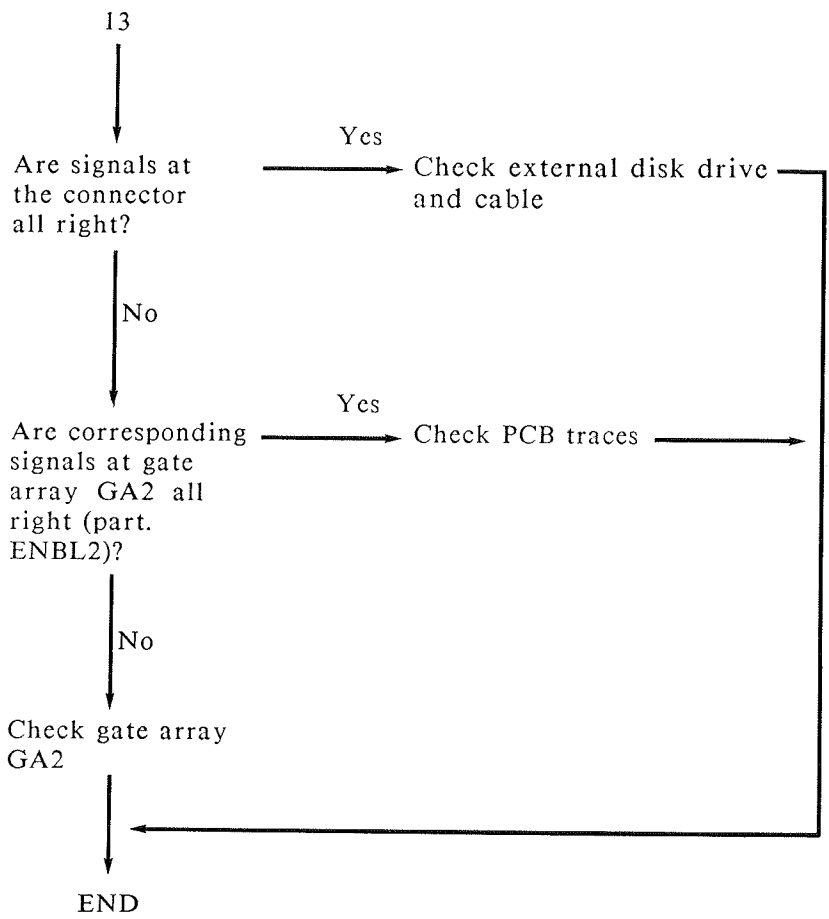

8
Is there signal from gate array GA1?
No
Check gate array GA1
Yes
Check speaker and related circuit
END

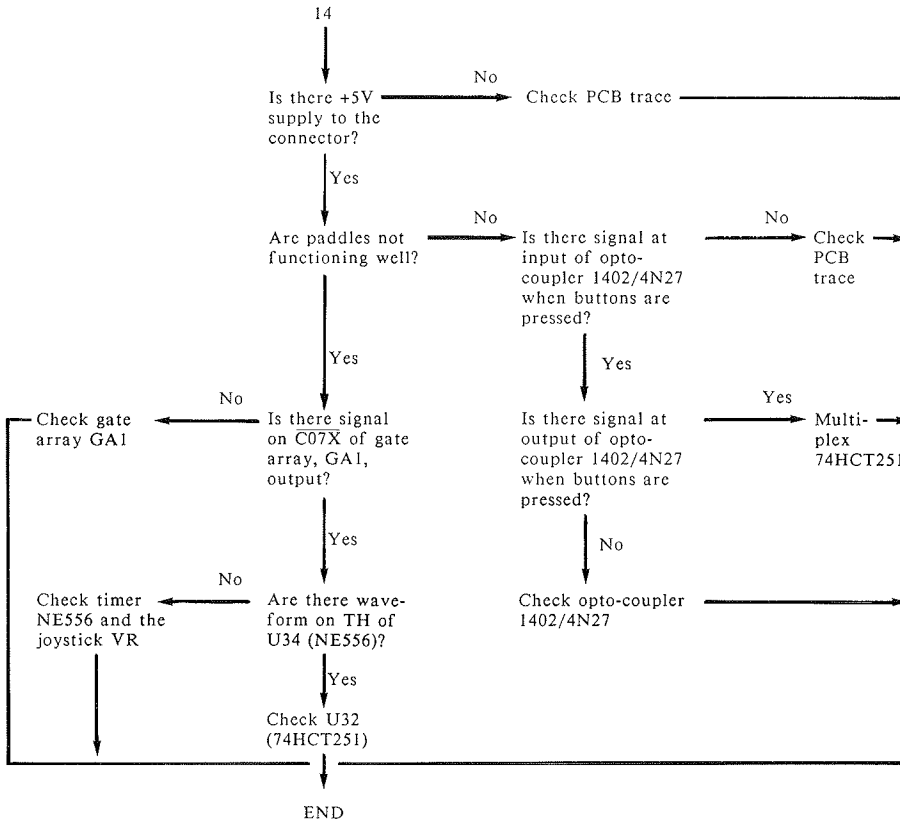

9
Is printer cable ok?
No
Replace cable
Yes
Is PARALLEL/ SERIAL switch set to PARALLEL?
No
Set switch to PARALLEL
Yes
Are PTRSTB, PLATCH and PTRBSY signals all right?
No
Check gate array GA1, printer cable and printer
Yes
Check printer data buffer 74HCT374 and related circuit and cable
END

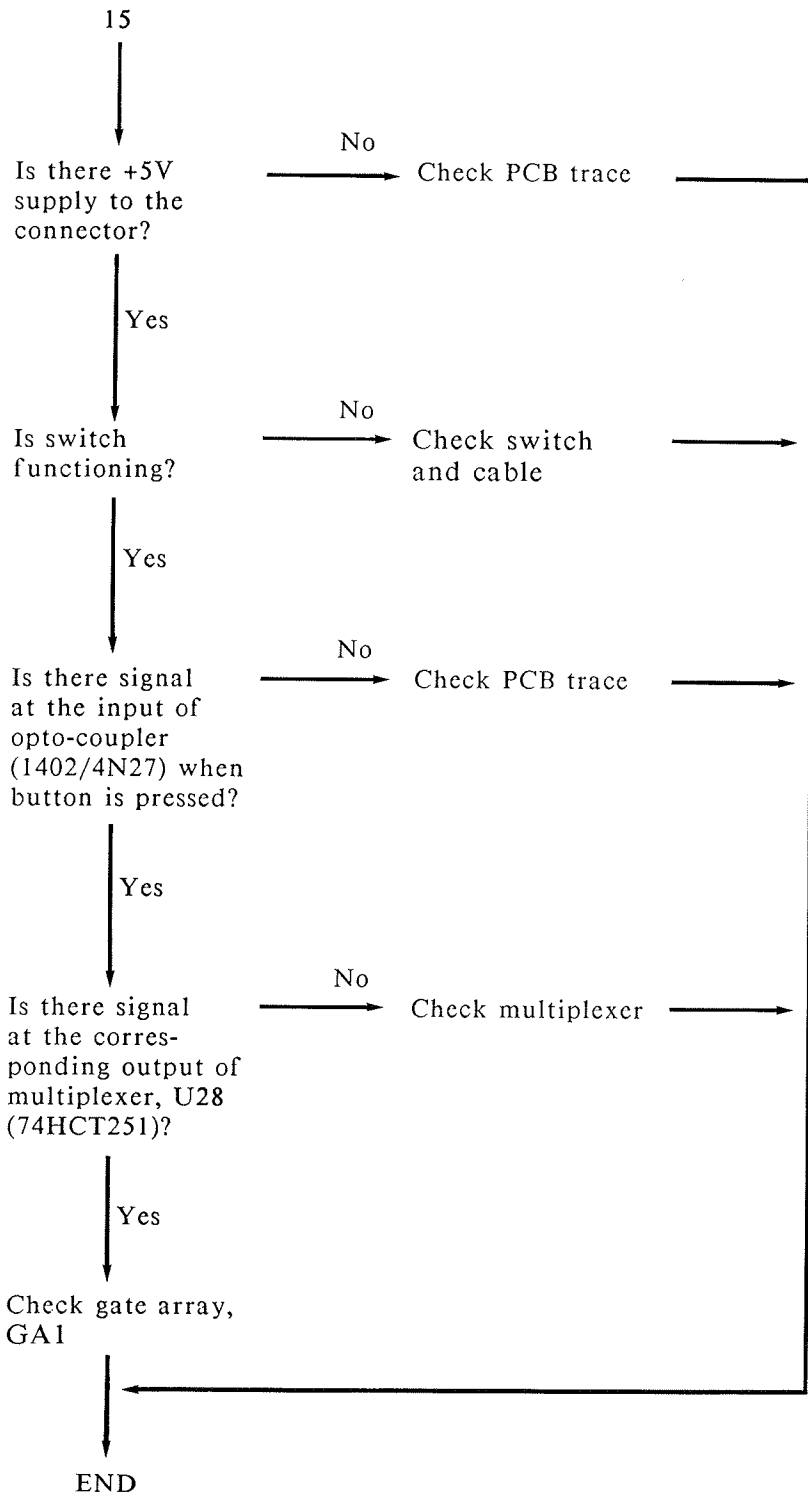

10
Is printer cable ok?
No
Replace printer cable
Yes
Is PARALLEL/ SERIAL switch set to SERIAL?
No
Set switch to SERIAL
Yes
Is +12V, -12V supplies to 1488 and 1489 all right?
No
Check power supply and related circuit
Yes
Are signals of 1488 and 1489 all right (TxD, RTS, DCD and RxD)?
No
Check 1488 and 1489
Yes
Check ACIA
END

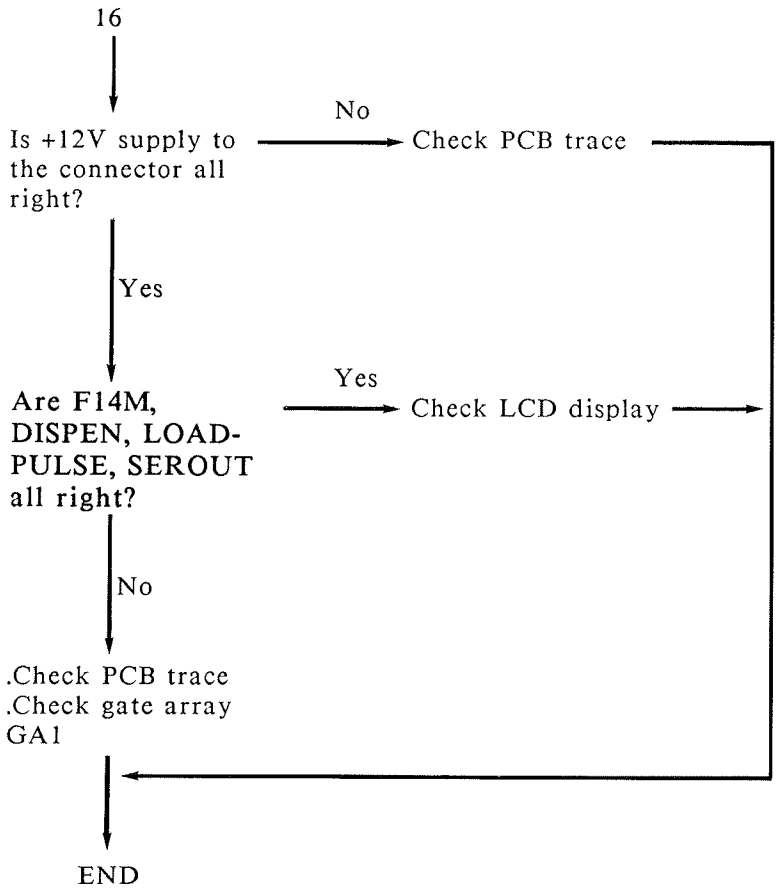

11
Are +12V, -12V supplies all right?
No
Check power supply and related circuit
Yes
Are signals from ACIA to 1488 all right?
No
Check 1488 and 1489
Yes
Check ACIA
END


12
Are VSYNC and HSYNC signals all right?
No
Check gate array GA1 and related circuit
Yes
Are R, G, B and I signals all right?
No
Yes
Check RGB monitor and related circuit and cable
END


13
Are signals at the connector all right?
Yes
Check external disk drive and cable
No
Are corresponding signals at gate array GA2 all right (part. ENBL2)?
Yes
Check PCB traces
No
Check gate array GA2
END


14
Is there +5V supply to the connector?
No
Check PCB trace
Yes
Are paddles not functioning well?
No
Is there signal at input of opto-coupler 1402/4N27 when buttons are pressed?
No
Check PCB trace
Yes
Yes
Is there signal on C07X of gate array, GA1, output?
No
Check gate array GA1
Is there signal at output of opto-coupler 1402/4N27 when buttons are pressed?
Yes
Multi-plex 74HCT251
Yes
No
Are there wave-form on TH of U34 (NE556)?
No
Check timer NE556 and the joystick VR
Check opto-coupler 1402/4N27
Yes
Check U32 (74HCT251)
END


15
Is there +5V supply to the connector?
No
Check PCB trace
Yes
Is switch functioning?
No
Check switch and cable
Yes
Is there signal at the input of opto-coupler (1402/4N27) when button is pressed?
No
Check PCB trace
Yes
Is there signal at the corres-ponding output of multiplexer, U28 (74HCT251)?
No
Check multiplexer
Yes
Check gate array, GA1
END


16
Is +12V supply to the connector all right?
No
Check PCB trace
Yes
Are F14M, DISPEN, LOAD-PULSE, SEROUT all right?
Yes
Check LCD display
No
.Check PCB trace
.Check gate array GA1
END

APPENDIX A

65C02 PROGRAMMING SPECIFICATION (The followings are extracted from GTE. G65SCXXX data sheets)

Addressing Modes

Fifteen addressing modes are available to the user of the GTE G65SCXXX family of microprocessors. The addressing modes are described in the following paragraphs:

Implied Addressing

In the implied addressing mode, the address containing the operand is implicitly stated in the operation code of the instruction.

Accumulator Addressing

This form of addressing is represented with a one byte instruction and implies an operation on the accumulator.

Immediate Addressing

With immediate addressing, the operand is contained in the second byte of the instruction; no further memory addressing is required.

Absolute Addressing

For absolute addressing, the second byte of the instruction specifies the eight low order bits of the effective address while the third byte specifies the eight high order bits. Therefore, this addressing mode allows access to the total 65K bytes of addressable memory.

Zero Page Addressing

Zero page addressing allows shorter code and execution times by only fetching the second byte of the instruction and assuming a zero high address byte. The careful use of zero page addressing can result in significant increase in code efficiency.

Ackn: GTE Microcircuits®

Absolute Indexed Addressing

Absolute indexed addressing is used in conjunction with X and Y index register and is referred to as "Absoulte, X", and "Absolute, Y". The effective address is formed by adding the contents of X and Y to the address contained in the second and third bytes of the instruction. This mode allows the index register to contain the index or count value and the instruction to contain the base address. This type of indexing allows any location referencing and the index to modify multiple fields resulting in reduced coding and execution time.

Zero Page Indexed Addressing

Zero page absolute addressing is used in conjunction with the index register and is referred to as "Zero Page, X" or "Zero Page, Y." The effective address is calculated by adding the second byte to the contents of the index register. Since this is a form of "Zero Page" addressing, the content of the second byte references a location in page zero. Additionally, due to the "Zero Page" addressing nature of this mode, no carrry is added to the high order eight bits of memory and crossing of page boundaries does not occur.

Relative Addressing

Relative addressing is used only with branch instruction; it establishes a destination for the conditional branch.

Zero Page Indexed Indirect Addressing

With zero page indexed indirect addressing (usually referrred to as Indirect X) the second byte of the instruction is added to the contents of the X index register; the carry is discarded. the result of this addition points to a memory location on page zero whose contents is the low order eight bits of the effective address. The next memroy location in page zero contains the high order eight bits of the effective address. Both memory locations specifying the high and low order bytes of the effective address must be in page zero.

Absolute Indexed Indirect Addressing (Jump Instruction Only)

With absolute indexed indirect addressing, the contents of the second and third instruction bytes are added to the X register. The result of this additon points to a memory location containing the

lower-order eight bits of the effective address. The next memory location contains the high-order eight bits of the effective address.

Indirect Indexed Addressing

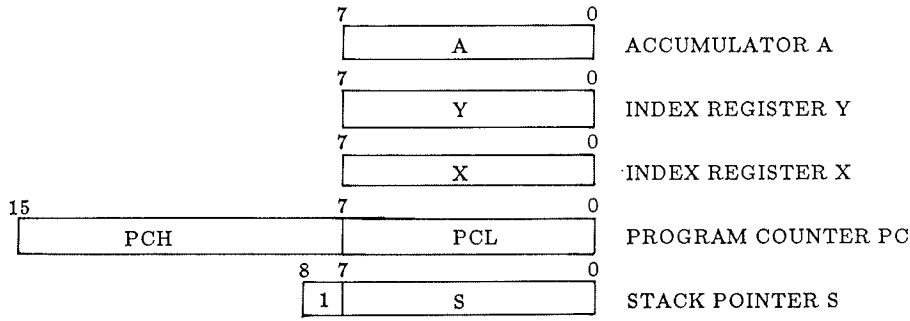
This form of addressing is usually referred to as Indirect, Y. The second byte of the instruction points to a memory location in page zero. The contents of this memory location is added to the contents of the Y index register, the result being the low order eight bits of the effective adress. The carry from this addition is added to the contents of the next page zero memory location, the result being the high order eight bits of the effective address.

Zero Page Indirect Addressing

In this form of addressing, the second byte of the instruction contains the low order eight bits of a memory location. The high order eight bits is always zero. The contents of the fully specified memory location is the low order byte of the effective address. The next memory location contains the high order byte of the effective address.

Absolute Indirect Addressing (Jump Instruction Only)

The second byte of the instruction contains the low order eight bits of a memory location. The high order eight bits of that memory location is contained in the third byte of the instruction. The contents of the fully specified memory location is the low order byte of the effective address. The next memory location contains the high order byte of the effective address which is loaded into the 16 bits of the program counter.



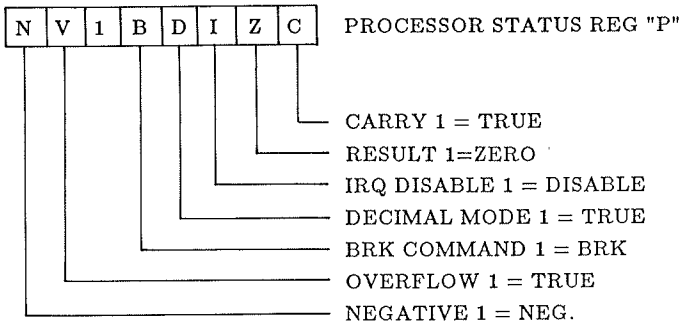


Figure A1 Microprocessor Programming Model

Table A1 Instruction Set-Alphabetical Sequence

ADC Add memory to Accumulator with Carray
AND "AND" Memory with Accumulator
ASL Shift One Bit Left
BCC Branch on Carry Clear
BCS Branch on Carry Set
BEQ Branch on Result Zero
BIT Test memory Bits with Accumulator
BMI Branch on Result Minus
BNE Branch on Result Not Zero
BPL Branch on Result Plus
•BRA Branch Always
BRK Force Break
BVC Branch on Overflow Clear
BVS Branch on Overflow Set
CLC Clear Carry Flag
CLD CLear Decimal Mode
CLI Clear Interrupt Disable Bit
CLV Clear Overflow Flag
CMP Compare Memory and Accumulator
CPX Compare memory
CPY Compare Memory and Index Y
DEC Decrement by One
DEX Decrement Index X by One
DEY Decrement Index Y by One
EOR "Exclusive-or" Memory with Accumulator
INC Increment by One
INX Increment Index X by One
INY Increment Index Y by One
JMP Jump to New Location

JSR	Jump to New Location Saving Return Address
LDA	Load Accumulator with Memory
LDX	Load Index X with Memory
LDY	Load Index Y with Memory
LSR	Shift One Bit Right
NOP	No Operation
ORA	"OR" Memory with Accumulator
PHA	Push Accumulator on Stack
PHP	Push Processor Status on Stack
•PHX	Push Index X on Stack
•PHY	Push Index Y on Stack
PLA	Pull Accumulator from Stack
PLP	Pull Processor Status from Stack
•PLX	Pull Index X from Stack
•PLY	Pull Index Y from Stack
ROL	Rotate One Bit Left
ROR	Rotate One Bit Right
RTI	Return from Interrupt
RTS	Retrun form Subroutine
SBC	Subtract Memory from Accumulator with Borrow
SEC	Set Carry Flag
SED	Set Decimal Mode
SEI	Set Interrupt Disable Bit
STA	Store Accumulator in memory
STX	Store Index X in Memory
STY	Store Index Y in Memory
•STZ	Store Zero in Memory
TAX	Transfer Accumulator to Index X
TAY	Transfer Accumulator to Index Y
•TRB	Test and Reset memory Bits with Accumulator
•TSB	Test and Set memory Bits with Accumulator
TSX	Transfer Stack Pointer to Index X
TXA	Transfer Index X to Accumulator
TXS	Transfer Index Y to Stack Pointer
TYA	Transfer Index Y to Accumulator

Note: • = New Instruction

LSD / MSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	BRK	ORA ind, X			TSB zpg	ORA zpg	ASL zpg		PHP	ORA imm	ASL A		TSB abs	ORA abs	ASL abs		0
1	BPL rel	ORA ind, Y	ORA ind		TRB zpg	ORA zpg, X	ASL zpg, X		CLC	ORA abs, Y	INC A		TRB abs	ORA abs, X	ASL abs, X		1
2	JSR abs	AND ind, X			BIT zpg	AND zpg	ROL zpg		PLP	AND imm	ROL A		BIT abs	AND abs	ROL abs		2
3	BMI rel	AND ind, Y	AND ind		BIT zpg X	AND zpg, X	ROL zpg, X		SEC	AND abs, Y	DEC A		BIT abs,X	AND abs, X	ROL abs, X		3
4	RTI	EOR ind, X				EOR zpg	LSR zpg		PHA	EOR imm	LSR A		JMP abs	EOR abs	LSR abs		4
5	BVC rel	EOR ind, Y	EOR ind			EOR zpg, X	LSR zpg, X		CLI	EOR abs, Y	PHY			EOR abs, X	LSR abs, X		5
6	RTS	ADC ind, X			STZ zpg	ADC zpg	ROR zpg		PLA	ADC imm	ROR A		JMP ind	ADC abs	ROR abs		6
7	BVS rel	ADC ind, Y	ADC ind		STZ zpg,X	ADC zpg, X	ROR zpg, X		SEI	ADC abs, Y	PLY		JMP ind,X	ADC abs, X	ROR abs, X		7
8	BRA rel	STA ind, X			STY zpg	STA zpg	STX zpg		DEY	BIT imm	TXA		STY abs	STA abs	STX abs		8
9	BCC rel	STA ind, Y	STA ind		STY zpg, X	STA zpg, X	STX zpg, Y		TYA	STA abs, Y	TXS		STZ abs	STA abs, X	STZ abs,X		9
A	LDY imm	LDA ind, X	LDX imm		LDY zpg	LDA zpg	LDX zpg		TAY	LDA imm	TAX		LDY abs	LDA abs	LDX abs		A
B	BCS rel	LDA ind, Y	LDA ind		LDY zpg, X	LDA zpg, X	LDX zpg, Y		CLV	LDA abs, Y	TSX		LDY abs, X	LDA abs, X	LDX abs, Y		B
C	CPY imm	CMP ind, X			CPY zpg	CMP zpg	DEC zpg		INY	CMP imm	DEX		CPY abs	CMP abs	DEC abs		C
D	BNE rel	CMP ind, Y	CMP ind			CMP zpg, X	DEC zpg, X		CLD	CMP abs, Y	PHX			CMP abs, X	DEC abs, X		D
E	CPX imm	SBC ind, X			CPX zpg	SBC zpg	INC zpg		INX	SBC imm	NOP		CPX abs	SBC abs	INC abs		E
F	BEQ rel	SBC ind, Y	SBC ind			SBC zpg, X	INC zpg, X		SED	SBC abs, Y	PLX			SBC abs, X	INC abs, X		F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Note: ▭ = New Op Codes

Figure A2 Microprocessor Op Code Table

MNE-MONIC	OPERATION	IMME-DIATE OP	n	#	ABSO-LUTE OP	n	#	ZERO PAGE OP	n	#	(4) IMPLIED OP	n	#	(IND, X) OP	n	#	(1) (IND) Y OP	n	#	ZPG,X OP	n	#	(1) ABS,X OP	n	#	(1) ABS,Y OP	n	#	RELA-TIVE (2) OP	n	#	INDI-RECT OP	n	#	ZPG,Y OP	n	#	PROCESSOR STATUS CODE 7 6 4 3 2 1 0 N V B D I Z C	MNE-MONIC
ADC	A + M + C → A (3)	69	2	2	6D	4	3	65	3	2				61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3				72	5	2				N V • • • Z C	ADC
AND	A∧M → A	29	2	2	2D	4	3	25	3	2				21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3				32	5	2				N • • • • Z •	AND
ASL	C ← [7 0] ← 0				0E	6	3	06	5	2	0A	2	1							16	6	2	1E	6	3													N • • • • Z C	ASL
BCC	BRANCH IF C=0																												90	2	2							• • • • • • •	BCC
BCS	BRANCH IF C=1																												B0	2	2							• • • • • • •	BCS
BEQ	BRANCH IF Z=1																												F0	2	2							• • • • • • •	BEQ
BIT	A∧M (5)	89	2	2	2C	4	3	24	3	2										34	4	2	3C	4	3													M M • • • Z •	BIT
BMI	BRANCH IF N=1																												30	2	2							• • • • • • •	BMI
BNE	BRANCH IF Z=0																												D0	2	2							• • • • • • •	BNE
BPL	BRANCH IF N=0																												10	2	2							• • • • • • •	BPL
BRA	BRANCH ALWAYS																												80	2	2							• • • • • • •	BRA
BRK	BREAK										00	7	1																									• • 1 0 1 • •	BRK
BVC	BRANCH IF V=0																												50	2	2							• • • • • • •	BVC
BVS	BRANCH IF V=1																												70	2	2							• • • • • • •	BVS
CLC	0 → C										18	2	1																									• • • • • • 0	CLC
CLD	0 → D										D8	2	1																									• • • 0 • • •	CLD
CLI	0 → I										58	2	1																									• • • • 0 • •	CLI
CLV	0 → V										B8	2	1																									• 0 • • • • •	CLV
CMP	A-M	C9	2	2	CD	4	3	C5	3	2				C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3				D2	5	2				N • • • • Z C	CMP
CPX	X-M	E0	2	2	EC	4	3	E4	3	2																												N • • • • Z C	CPX
CPY	Y-M	C0	2	2	CC	4	3	C4	3	2																												N • • • • Z C	CPY
DEC	DECREMENT				CE	6	3	C6	5	2	3A	2	1							D6	6	2	DE	6	3													N • • • • Z •	DEC
DEX	X-1 → X										CA	2	1																									N • • • • Z •	DEX
DEY	Y-1 → Y										88	2	1																									N • • • • Z •	DEY
EOR	A ∨ M → A	49	2	2	4D	4	3	45	3	2				41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3				52	5	2				N • • • • Z •	EOR
INC	INCREMENT				EE	6	3	E6	5	2	1A	2	1							F6	6	2	FE	6	3													N • • • • Z •	INC
INX	X + 1 → X										E8	2	1																									N • • • • Z •	INX
INY	Y + 1 → Y										C8	2	1																									N • • • • Z •	INY
JMP	JUMP TO NEW LOC				4C	3	3							7C	6	3																						• • • • • • •	JMP
JSR	JUMP SUB				20	6	3																															• • • • • • •	JSR
LDA	M → A	A9	2	2	AD	4	3	A5	3	2				A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3				B2	5	2				N • • • • Z •	LDA
LDX	M → X	A2	2	2	AE	4	3	A6	3	2																BE	4	3							B6	4	2	N • • • • Z •	LDX
LDY	M → Y	A0	2	2	AC	4	3	A4	3	2										B4	4	2	BC	4	3													N • • • • Z •	LDY
LSR	0 → [7 0] → C				4E	6	3	46	5	2	4A	2	1							56	6	2	5E	6	3													0 • • • • Z C	LSR
NOP	NO OPERATION										EA	2	1																									• • • • • • •	NOP
ORA	AVM → A	09	2	2	0D	4	3	05	3	2				01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3				12	5	2				N • • • • Z •	ORA
PHA	A →Ms S-1→S										48	3	1																									• • • • • • •	PHA
PHP	P →Ms S-1→S										08	3	1																									• • • • • • •	PHP
PHX	X →Ms S-1→S										DA	3	1																									• • • • • • •	PHX
PHY	Y →Ms S-1→S										5A	3	1																									• • • • • • •	PHY
PLA	S+1→S Ms→A										68	4	1																									N • • • • Z •	PLA
PLP	S+1→S Ms→P										28	4	1																									N V 1 D I Z C	PLP
PLX	S+1→S Ms→X										FA	4	1																									N • • • • Z •	PLX
PLY	S+1→S Ms→Y										7A	4	1																									N • • • • Z •	PLY
ROL	[7 0] ← C ←				2E	6	3	26	5	2	2A	2	1							36	6	2	3E	6	3													N • • • • Z C	ROL
ROR	C → [7 0] →				6E	6	3	66	5	2	6A	2	1							76	6	2	7E	6	3													N • • • • Z C	ROR
RTI	RTRN INT										40	6	1																									N V 1 D I Z C	RTI
RTS	RTRN SUB										60	6	1																									• • • • • • •	RTS
SBC	A-M-C̄ →A (3)	E9	2	2	ED	4	3	E5	3	2				E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3				F2	5	2				N V • • • Z C	SBC
SEC	1 → C										38	2	1																									• • • • • • 1	SEC
SED	1 → D										F8	2	1																									• • • 1 • • •	SED
SEI	1 → I										78	2	1																									• • • • 1 • •	SEI
STA	A → M				8D	4	3	85	3	2				81	6	2	91	6	2	95	4	2	9D	5	3	99	5	3				92	5	2				• • • • • • •	STA
STX	X → M				8E	4	3	86	3	2																											96	4	2	• • • • • • •	STX
STY	Y → M				8C	4	3	84	3	2										94	4	2																	• • • • • • •	STY
STZ	00 → M				9C	4	3	64	3	2										74	4	2	9E	5	3													• • • • • • •	STZ
TAX	A → X										AA	2	1																									N • • • • Z •	TAX
TAY	A → Y										A8	2	1																									N • • • • Z •	TAY
TRB	A∧M → M (6)				1C	6	3	14	5	2																													• • • • • Z •	TRB
TSB	A∨M → M (6)				0C	6	3	04	5	2																													• • • • • Z •	TSB
TSX	S → X										BA	2	1																									N • • • • Z •	TSX
TXA	X → A										8A	2	1																									N • • • • Z •	TXA
TXS	X → S										9A	2	1																									• • • • • • •	TXS
TYA	Y → A										98	2	1																									N • • • • Z •	TYA

Table A2 Operational Codes, Execution Time, and Memory Requirements

Notes:

1. Add 1 to "n" if page boundary is crossed, except STA and STZ.
2. Add 1 to "n" if branch occurs to same page. Add 2 to "n" if branch occurs to different page.
3. Add 1 to "n" if decimal mode.
4. Accumulator address is included in Implied address.
5. "N" and "V" flags are unchanged in immediate mode.
6. "Z" flag indicates A^M result (Same as BIT instruction).

Y	Index Y
A	Accumulator
M	Memory per effective address
Ms	Memory per stack pointer
+	Add
-	Subtract
^	And
V	Or
V	Exclusive or
n	No. Cycles
#	No. Bytes
M_6	Memory Bit #6
M_7	Memory Bit #7

APPENDIX B

CUSTOM IC PIN ASSIGNMENT

The computer includes two custom-made integrated circuit for addressing, decoding, memory management, video signal processing, input/output port control and other logic control. The pin assignment and descriptions are listed in TABLE B1 and B2.

Pin no.	Signal name	I/O	Description
1	SPKR	O	toggles the speaker circuit to generate a "click" sound
2	XINT	I	x-direction interrupt of the mouse
3	YINT	I	y-direction interrupt of the mouse
4	$\overline{INH}$	I	active low to disable internal ROM/EPROM(s) and RAM.
5	HSYNC	O	horizontal sync of the video circuit
6	$\overline{IOSTB}$	O	active low when I/O locations $C800 to $CFFF are accessed
7	$\overline{COFX}$	O	active low when I/O locations $C0F0-$C0FF are accessed
8	$\overline{C7XX}$	O	active low when I/O locations $C700 to $C7FF are accessed
9	$\overline{C0DX}$	O	active low when I/O locations $C0D0 to $C0DF are accessed
10	$\overline{C5XX}$	O	active low when I/O locations $C500 to $C5FF are accessed
11	FSEL	I	active high to select NTSC video output
12	PPTR	I	active high to indicate a parallel printer
13	COLOR	I	active high to enable color output in video signal
14	F7M	O	system 7MHz clock
15	GND1		
16	$\overline{\mu PRST}$	I	connected directly to 65C02's $\overline{RESET}$
17	R/$\overline{W}$	I	system Read/$\overline{Write}$
18	Φ0	O	65C02's phase 0 clock
19	$\overline{RAS}$	O	common $\overline{RAS}$ signal of the RAMs
20	$\overline{MCAS}$	O	$\overline{CAS}$ signal of the main RAM

21--28	RA0--RA7	O	multiplexed address bus of RAM
29--40	A15--A4	I	system address bus
41	VCC1		+5V
42--45	A3--A0	I	system address bus
46	D7	I/O	Data bus
47--49	D6--D4	I	Data bus
50,51	VID7, VID6	I	8-th and 7-th bit of the video data
52	$\overline{\text{MAINOE}}$	O	active low to enable main RAM output buffer
53	$\overline{\text{AUXOE}}$	O	active low to enable auxiliary RAM output buffer
54	$\overline{\text{ACAS}}$	O	$\overline{\text{CAS}}$ of auxiliary RAM
55	$\overline{\text{374OE}}$	O	enable signal of the video data buffer
56--58	VROMA0--VROMA2	O	video ROM address A0 to A2
59,60	VROMA9--VROMA10	O	video ROM address A9, A10
61	F14M	I	14MHz system clock
62	XTAL	I	video clock from oscillator
63	CHROMA	O	chroma of the video signal
64	BURST	O	color burst of the video signal
65	GND2		
66	GR	O	active high to indicate graphics mode
67	VDATA	I	video data
68	$\overline{\text{LDPS}}$	O	active low to load the shift register, U22
69	VID7M	O	7MHz video clock to gate with F14M to shift the shift register
70	$\overline{\text{DISPEN}}$	O	active low to enable external display module
71	PTRBSY	I	active high to indicate the parallel printer not ready to accept data
72	$\overline{\text{PRSTB}}$	O	active low to strobe data into the parallel printer
73	$\overline{\text{PLATCH}}$	O	active low to latch data into the parallel printer data buffer
74	KBDSTB	I	active high input to GA1 to indicate a key has been pressed
75	$\overline{\text{KBD}}$	O	active low to enable output of keyboard ROM/EPROM
76	$\overline{\text{GC06X}}$	O	active low to enable multiplexer output of the hand controls/mouse

77	$\overline{C07X}$	O	active low when I/O locations $C070 to $C07F are accessed
78--81	C0--C3	O	color information of the video signal
82	$\overline{CSYNC}$	O	composite sync of the video signal
83	$\overline{ACIACS1}$	O	active low to enable ACIAs
84	ACIA1CS0	O	active high to enable ACIA of port 1
85	ACIA2CS0	O	active high to enable ACIA of port 2
86	$\overline{IRQ}$	O	connected directly to 65C02's $\overline{IRQ}$
87--89	ROMA12--ROMA14	O	high order address lines of system ROM/EPROM
90	$\overline{ROMOE}$	O	active low to enable system ROM/EPROM
91	VCC2		+5V
92	ZCLK	O	clock for internal Z80 CPU
93	$\overline{INTMOUSE}$	I	active low to activate internal mouse
94	Q3	O	asymmetrical 2MHz clock
95	F358M	O	system 3MHz clock
96	$\overline{373LE}$	O	active low to enable output of data buffer of internal Z80 module
97	VSYNC	O	vertical sync of the video signal
98	$\overline{RESET}$	I	connected to $\overline{GARST}$
99	AKD	I	active high to enable the keyboard encoder
100	CXXX	O	active high when I/O locations $C000 to $CFFF are accessed

Table B1 Pin assignment and signal description of custom made IC1.

Pin no	**Signal name**	**I/O**	**Description**
1	GND1		
2	$\overline{ZDAT}$	O	active low to enable data buffer of internal Z80 module
3	$\overline{DMA}$	I	connected directly to system $\overline{DMA}$
4	RDY	O	connected directly to system RDY
5	$\overline{ZWAIT}$	O	connected directly to Z80's $\overline{WAIT}$
6	$\overline{RFSH}$	I	connected directly to Z80's $\overline{RFSH}$

7	$\overline{ZWR}$	I	connected directly to Z80's $\overline{WR}$
8	N.C.		
9	ZCLK	O	clock for Z80 CPU
10	VSYNC	I	vertical sync of the video signal
11	Q3	I	asymmetrical 2MHz clock
12	$\overline{\Phi 0}$	I	65C02's phase 0 clock
13	PPTR	I	PPTR=1; parallel printer PPTR=0; serial printer
14	$\overline{TCTRL}$	I	active low to enable R/$\overline{W}$ buffer of 65C02
15	TIN	I	connected directly to 65C02's Read/$\overline{Write}$
16	TOUT	O	connected directly to system Read/$\overline{Write}$
17	R/$\overline{W}$	I	system Read/$\overline{Write}$
18	CXXX	I	active high when I/O locations $C000 to $CFFF are accessed
19--24	A11--A6	I	system address bus
25	N.C.		
26--31	A5--A0	I	system address bus
32	GND2		
33	$\overline{RESET}$	I	connected to $\overline{\mu PRST}$
34--39	D0--D5	I/O	system data bus
40	N.C.		
41,42	D6,D7	I/O	system data bus
43--46	ZA15--ZA12	I	high order 4 bits address bus of Z80 CPU
47--50	TA15--TA12	I	high order 4 bits address bus from GA2 after transformation of ZA15 through ZA12
51	$\overline{ZADR}$	I	active low to enable address buffer of the internal Z80 module
52	RDATA	I	serial data input from disk drive
53	WPROT	O	active high to indicate current diskette being write-protected
54	$\overline{WREQ}$	O	active low to write data onto current diskette
55	WDATA	O	serial data output to disk drive
56	$\overline{ENBL2}$	O	active low to enable external disk drive
57	N.C.		
58	$\overline{ENBL1}$	O	active low to enable internal disk drive
59	SIDE1	O	active high to access side 1 of the diskette

60--63	PH0--PH3	O	active high to turn on corresponding phases of the stepper motor of the current drive
64	VCC		+5V

Table B2 Pin assignment and signal description of custom made IC2

APPENDIX C

UM 6551
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER

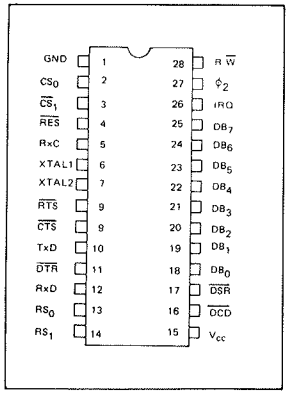
FEATURES

- On-chip baud rate generator: 15 programmable baud rates derived from a standard 1.8432 MHz external crystal (50 to 19,200 baud).
- Programmable interrupt and status register to simplify software design.
- Single + 5 volt power supply.
- Serial echo mode.
- False start bit detection.
- 8-bit bi-directional data bus for direct communication with the microprocessor.
- External 16x clock input for non-standard baud rates (up to 125 k baud).
- Programmable: word lengths; number of stop bits; and parity bit generation and detection.
- Data set and modem control signals provided.
- Parity: (odd, even, none, mark, space).
- Full-duplex or half-duplex operation.
- 5,6,7,8, and 9 bit transmission.

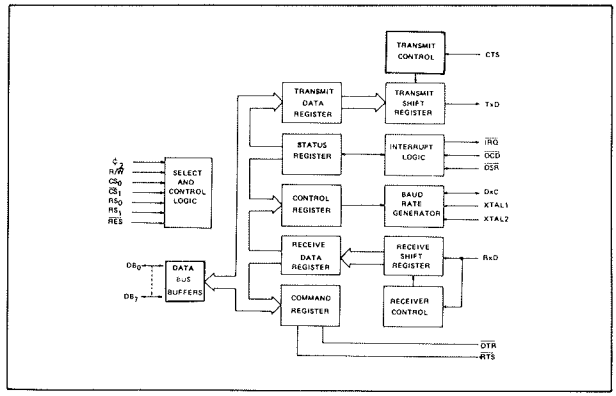
GENERAL DESCRIPTION

The UM6551 is an Asynchronous Communication Adapter (ACIA) intended to provide for interfacing the 6500/6800 microprocessor families to serial communication data sets and modems. A unique feature is the inclusion of an on-chip programmable baud rate generator, with a crystal being the only external component required.

PIN CONFIGURATION



BLOCK DIAGRAM



Ackn: United Microelectronics Corp..

ABSOLUTE MAXIMUM RATINGS

RATING	SYMBOL	ALLOWABLE RANGE
Supply Voltage	V_{CC}	-0.3V to + 7.0V
Input/Output Voltage	V_{IN}	-0.3V to + 7.0V
Operating Temperature	T_{OP}	0 °C to 70°C
Storage Temperature	T_{STG}	-55° C to 150° C

COMMENT

Stresses above those listed under "Absolute Maximum Ratings", may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

All inputs contain protection circuitry to prevent damage to high static charges. Care should be exercised to prevent unnecessary application of voltages in excess of the allowable limits.

D. C. CHARACTERISTICS (V_{CC} = 5.0V ± 5%, T_A = 0– 70°C, unless otherwise noted)

CHARACTERISTIC	SYMBOL	MIN	TYP	MAX	UNIT
Input High Voltage	V_{IH}	2.0	–	V_{CC}	V
Input Low Voltage	V_{IL}	–0.3	–	0.8	V
Input Leakage Current: V_{IN} = 0 to 5V ($\phi 2$, R/$\overline{W}$, $\overline{RES}$, CS_0, $\overline{CS_1}$, RS_0, RS_1, $\overline{CTS}$, RxD, $\overline{DCD}$, $\overline{DSR}$)	I_{IN}	–	±1.0	±2.5	µA
Input Leakage Current for High Impedance State (Three State)	I_{TSI}	–	±2.0	±10.0	µA
Output High Voltage: I_{LOAD} = -100µA (DB_0 - DB_7, TxD, RxC, $\overline{RTS}$, $\overline{DTR}$)	V_{OH}	2.4	–	–	V
Output Low Voltage: I_{LOAD} = 1.6mA (DB_0 - DB_7, TxD, RxC, $\overline{RTS}$, $\overline{DTR}$, $\overline{IRQ}$)	V_{OL}	–	–	0.4	V
Output High Current (Sourcing): V_{OH} = 2.4V (DB_0 - DB_7, TxD, RxC, $\overline{RTS}$, $\overline{DTR}$)	I_{OH}	–100	–	–	µA
Output Low Current (Sinking): V_{OL} = 0.4V (DB_0 - DB_7, TxD, RxC, $\overline{RTS}$, $\overline{DTR}$, $\overline{IRQ}$)	I_{OL}	1.6	–	–	mA
Output Leakage Current (Off State): V_{OUT} = 5V ($\overline{IRQ}$)	I_{OFF}	–	1.0	10.0	µA
Clock Capacitance ($\phi 2$)	C_{CLK}	–	–	20	pF
Input Capacitance (Except XTAL 1 and XTAL2)	C_{IN}	–	–	10	pF
Output Capacitance	C_{OUT}	–	–	10	pF
Power Dissipation (See Graph)(T_A = 0°C) V_{CC} = 5.25V	P_D	–	170	300	mW

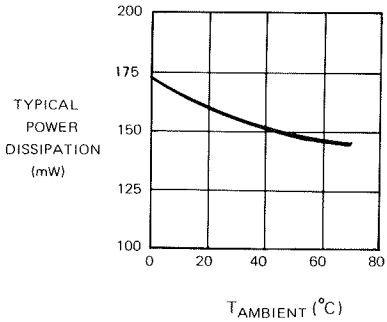


Figure C1 Power Dissipation vs. Temperature

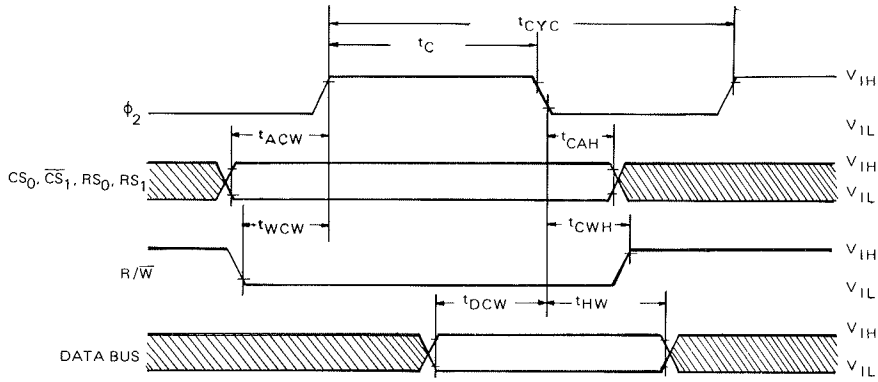


Figure C2 Write Timing Characteristics

WRITE CYCLE (V_{CC} = 5.0V ± 5%, T_A = 0 to 70°C, unless otherwise noted)

CHARACTERISTIC	SYMBOL	UM6551		UM6551A		UNIT
		MIN	MAX	MIN	MAX	
Cycle Time	t_{CYC}	1.0	-	0.5	-	µs
ϕ2 Pulse Width	t_C	400	-	200	-	ns
Address Set-Up Time	t_{ACW}	120	-	70	-	ns
Address Hold Time	t_{CAH}	0	-	0	-	ns
R/$\overline{W}$ Set-Up Time	t_{WCW}	120	-	70	-	ns
R/$\overline{W}$ Hold Time	t_{CWH}	0	-	0	-	ns
Data Bus Set-Up Time	t_{DCW}	150	-	60	-	ns
Data Bus Hold Time	t_{HW}	20	-	20	-	ns

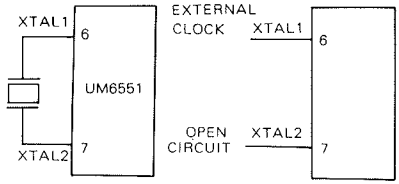
(t_r and t_f = 10 to 30 ns)

CRYSTAL SPECIFICATION

1. Temperature stability ± o.o1% (0° to 70° C)
2. Characteristics at 25° C ± 2° C
   a. Frequency (MHz) 1.8432
   b. Frequency tolerance (±%) 0.02
   c. Resonance mode Series
   d. Equivalent resistance (ohm) 400 max.
   e. Drive level mW 2
   f. Shunt capacitance pF 7 max.
   g. Oscillation mode Fundamental

No other external components should be in the crystal circuit

CLOCK GENERATION



INTERNAL CLOCK **EXTERNAL CLOCK**

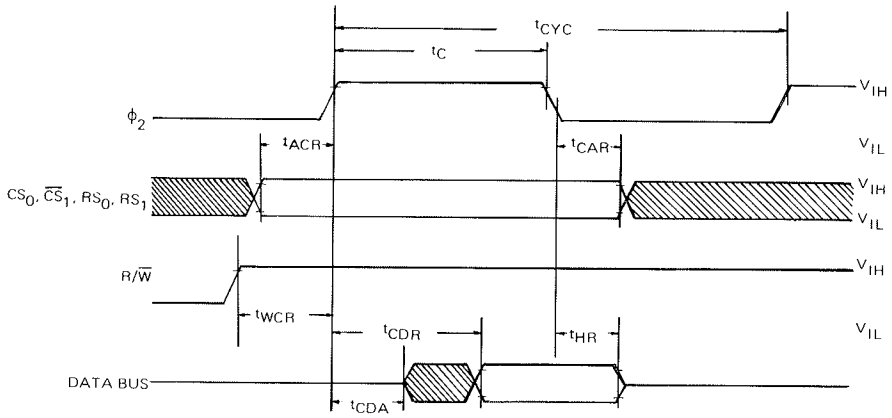
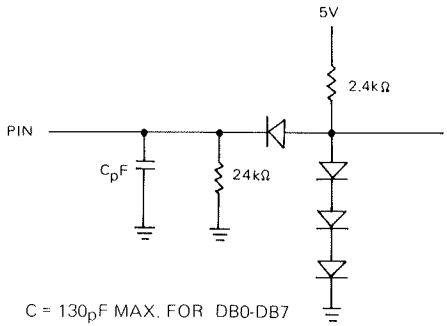


Figure C3 Read Timing Characteristics

READ CYCLE (V_{CC} = 5.0V ± 5%, T_A = 0 to 70°C, unless otherwise noted)

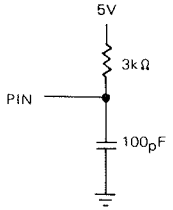
CHARACTERISTIC	SYMBOL	UM6551		UM6551A		UNIT
		MIN	MAX	MIN	MAX	
Cycle Time	t_{CYC}	1.0	–	0.5	–	µs
Pulse Width (ϕ2)	t_C	400	–	200	–	ns
Address Set-Up Time	t_{ACR}	120	–	70	–	ns
Address Hold Time	t_{CAR}	0	–	0	–	ns
R/W Set-Up Time	t_{WCR}	120	–	70	–	ns
Read Access Time (Valid Data)	t_{CDR}	–	200	–	150	ns
Read Data Hold Time	t_{HR}	20	–	20	–	ns
Bus Active Time (Invalid Data)	t_{CDA}	40	–	40	–	ns

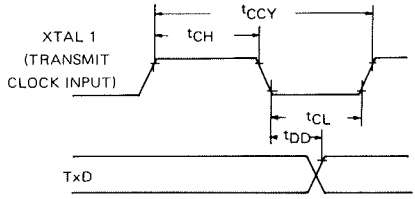
TEST LOAD



C = 130pF MAX. FOR DB0-DB7
C = 30pF MAX. FOR ALL OTHER OUTPUTS

OPEN COLLECTOR
OUTPUT TEST LOAD





NOTE: TxD rate is 1/16 TxC rate

Figure C4(a) Transmit Timing with External Clock

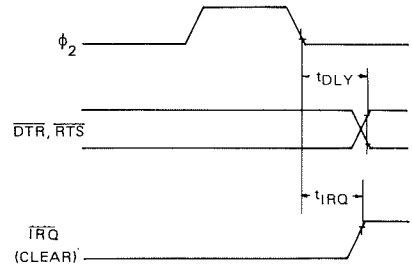
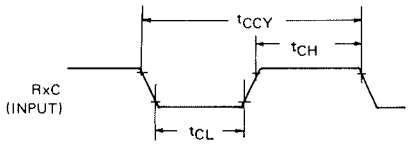


Figure C4(b) Interrupt and Output Timing



NOTE: RxD rate is 1/16 RxC rate.

Figure C4(c) Receive External clock Timing

TRANSMIT/RECEIVE CHARACTERISTICS

CHARACTERISTIC	SYMBOL	UM6551		UM6551A		UNIT
		MIN	MAX	MIN	MAX	
Transmit/Receive Clock Rate	t_{CCY}	400*	–	400*	–	ns
Transmit/Receive Clock High Time	t_{CH}	175	–	175	–	ns
Transmit/Receive Clock Low Time	t_{CL}	175	–	175	–	ns
XTAL1 to TxD Propagation Delay	t_{DD}	–	500	–	500	ns
Propagation Delay ($\overline{RTS}$, $\overline{DTR}$).	t_{DLY}	–	500	–	500	ns
$\overline{IRQ}$ Propagation Delay (Clear)	t_{IRQ}	–	500	–	500	ns

(t_r, t_f = 10 to 30 ns input clocks only)

* The baud rate with external clocking is: Baud Rate = $\frac{1}{16 \times T_{CCY}}$

INTERFACE SIGNAL DESCRIPTION

$\overline{RES}$ (Reset) — During system initialization a low on the $\overline{RES}$ input will cause internal registers to be cleared.

ϕ2 (Input Clock) — The input clock is the system ϕ_2 clock and is used to trigger all data transfers between the system microprocessor and the UM6551.

R/$\overline{W}$ (Read/Write) — The R/$\overline{W}$ is generated by the microprocessor and is used to control the direction of data transfers. A high on the R/$\overline{W}$ pin allows the processor to read the data supplied by the UM6551. A low on the R/$\overline{W}$ pin allows a write to the UM6551.

$\overline{IRQ}$ (Interrupt Request) — The $\overline{IRQ}$ pin is an interrupt signal from the interrupt control logic. It is an open drain output, permitting several devices to be connected to the common $\overline{IRQ}$ microprocessor input. Normally a high level, $\overline{IRQ}$ goes low when an interrupt occurs.

DB_0 - DB_7 (Data Bus)

The DB_0 -DB_7 pins are the eight data lines used for transfer of data between the processor and the UM6551. These lines, are bi-directional and are normally high-impedance except during Read cycles when selected.

CS_0,CS_1 (Chip Selects)

The two chip select inputs are normally connected to the processor address lines either directly or through decoders. The UM6551 is selected when CS_0 is high and $\overline{CS}_1$ is low.

RS_ϕ, RS_1 (Register Selects)

The two register select lines are normally connected to the processor address lines to allow the processor to select the various UM6551 internal registers. The following table indicates the internal register select coding.

RS_1	RS_0	WRITE	READ
0	0	Transmit Data Register	Receiver Data Register
0	1	Programmed Reset (Data is "Don't Care")	Status Register
1	0	Command Register	
1	1	Control Register	

The table shows that only the Command and Control registers are read/write. The Programmed Reset operation does not cause any data transfer, but is used to clear the UM6551 registers. The Programmed Reset is slightly different from the Hardware Reset ($\overline{RES}$) and these differences are described in the individual register definitions.

ACIA/MODEM INTERFACE SIGNAL DESCRIPTION

XTAL1, XTAL2 (Crystal Pins)

These pins are normally directly connected to the external crystal (1.8432 MHz) used to derive the various baud rates. Alternatively, an externally generated clock may be used to drive the XTAL1 pin, in which case the XTAL2 pin must float.

TxD (Transmit Data)

The TxD output line is used to transfer serial NRZ (nonreturn-to-zero) data to the modem. The LSB (least significant bit) of the Transmit Data Register is the first data bit transmitted and the rate of data transmission is determined by the baud rate selected.

RxD (Receive Data)

The RxD input line is used to transfer serial NRZ data into the ACIA from the modem, LSB first. The receiver data rate is either the programmed baud rate or the rate of an externally generated receiver clock. This selection is made by programming the Control Register.

RxC (Receive Clock)

The RxC is a bi-directional pin which serves as either the receiver 16x clock input or the receiver 16x clock output. The latter mode results if the internal baud rate generator is selected for receiver data clocking.

$\overline{RTS}$ (Request to Send)

The $\overline{RTS}$ output pin is used to control the modem from the processor. The state of the $\overline{RTS}$ pin is determined by the contents of the Command Register.

$\overline{CTS}$ (Clear to Send)

The $\overline{CTS}$ input pin is used to control the transmitter operation. The enable state is with $\overline{CTS}$ low. The transmitter is automatically disabled if $\overline{CTS}$ is high.

$\overline{DTR}$ (Data Terminal Ready)

This output pin is used to indicate the status of the UM6551 to the modem. A low on $\overline{DTR}$ indicates the UM6551 is enabled and a high indicates it is disabled. The processor controls this pin via bit 0 of the Command Register.

$\overline{DSR}$ (Data Set Ready)

The $\overline{DSR}$ input pin is used to indicate to the UM6551 the status of the modem. A low indicates the "ready" state and a high, "not-ready." $\overline{DSR}$ is a high-impedance input and must not be a no-connect. If unused, it should be driven high or low, but not switched.

Note: If Command Register Bit 0=1 and a change of state on $\overline{DSR}$ occurs, $\overline{IRQ}$ will be set, and Status Register Bit 6 will reflect the new level. The state of $\overline{DSR}$ does not affect either Transmitter or Receiver operation.

$\overline{DCD}$ (Data Carrier Detect)

The $\overline{DCD}$ input pin is used to indicate to the UM6551 the status of the carrier-detect output of the modem. A low indicates that the modem carrier signal is present and a high, that it is not. $\overline{DCD}$, like $\overline{DSR}$, is a high-impedance input and must not be a no-connect.

Note: If Command Register Bit 0=1 and a change of state on $\overline{DCD}$ occurs, $\overline{IRQ}$ will be set, and Status Register Bit 5 will reflect the new level. The state of $\overline{DCD}$ does not affect Transmitter operation, but must be low for the Receiver to operate.

INTERNAL ORGANIZATION

The Transmitter/Receiver sections of the UM6551 are depicted by the block diagram in Figure 5.

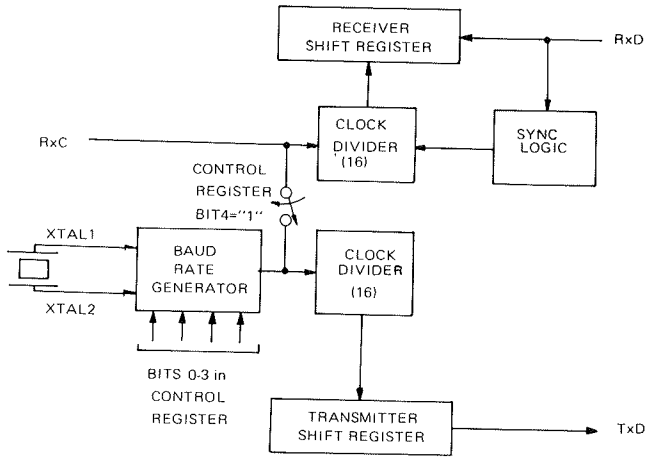


Figure C5 Transmitter/Receiver clock circuits

Bits 0-3 of the Control Register select the divisor used to generate the baud rate for the Transmitter. If the Receiver clock is to use the same baud rate as the Transmitter, then RxC becomes an output pin and can be used to slave other circuits to the UM6551.

CONTROL REGISTER

The Control Register is used to select the desired mode for the UM6551. The word length, number of stop bits, and clock controls are all determined by the Control Register, which is depicted in Figure 6.

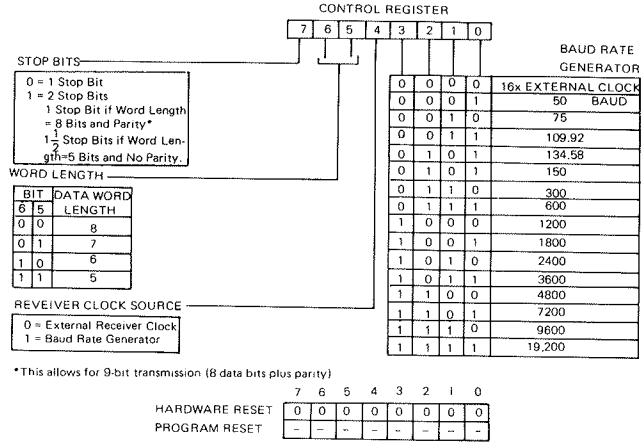


Figure C6 Control Register Format

COMMAND REGISTER

The Command Register is used to control Specific Transmit/Receive functions and is show in Figure 7.

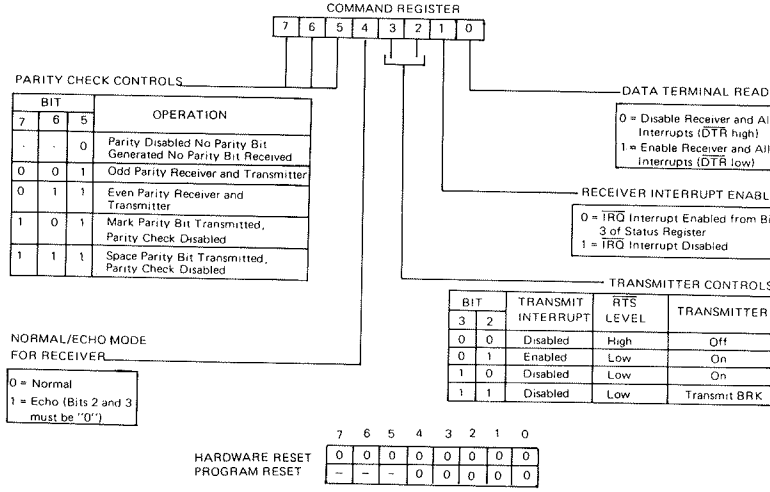
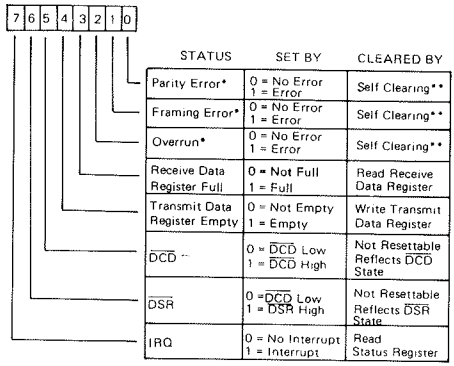


Figure C7 Command Register Format

STATUS REGISTER

The Status Register is used to indicate to the processor the status of various UM6551 functions and is outlined in Figure 8.



STATUS	SET BY	CLEARED BY
Parity Error*	0 = No Error 1 = Error	Self Clearing**
Framing Error*	0 = No Error 1 = Error	Self Clearing**
Overrun*	0 = No Error 1 = Error	Self Clearing**
Receive Data Register Full	0 = Not Full 1 = Full	Read Receive Data Register
Transmit Data Register Empty	0 = Not Empty 1 = Empty	Write Transmit Data Register
$\overline{DCD}$	0 = $\overline{DCD}$ Low 1 = $\overline{DCD}$ High	Not Resettable Reflects $\overline{DCD}$ State
$\overline{DSR}$	0 = $\overline{DCD}$ Low 1 = $\overline{DSR}$ High	Not Resettable Reflects $\overline{DSR}$ State
IRQ	0 = No Interrupt 1 = Interrupt	Read Status Register

*NO INTERRUPT GENERATED FOR THESE CONDITIONS.
**CLEARED AUTOMATICALLY AFTER A READ OF RDR AND THE NEXT ERROR FREE RECEIPT OF DATA.

	7	6	5	4	3	2	1	0
HARDWARE RESET	0	–	–	1	0	0	0	0
PROGRAM RESET	–	–	–	–	–	0	–	–

Figure C8 Status Register Format

TRANSMIT AND RECEIVE DATA REGISTERS

These registers are used as temporary data storage for the UM6551 Transmit and Receive circuits. The Transmit Data Register is characterized as follows:

- Bit 0 is the leading bit to be transmitted.
- Unused data bits are the high-order bits and are "don't care" for transmission.

The Receive Data Register is characterized in a similar fashion:

- Bit 0 is the leading bit received.
- Unused data bits are the high-order bits and are "0" for the receiver.
- Parity bits are not contained in the Receive Data Register, but are stripped-off after being used for external parity checking. Parity and all unused high-order bits are "0".

Figure 9 illustrates a single transmitted or received data word, for the example of 8 data bits, parity, and 1 stop bit.

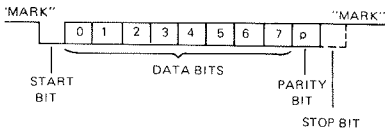


Figure C9 Serial Data Stream Example

ORDERING INFORMATION

PART NO.	PACKAGE	CLOCK RATE
UM6551	Plastic	1 MHz
UM6551A	Plastic	2 MHz

APPENDIX D

KEYBOARD LAYOUTS, KEYS AND THEIR ASSOCIATED ASCII CODES

Figure D1 to Figure D5 are different keyboard layouts for different versions of the computer.

TABLE D1 to TABLE D5 show the ASCII code of different versions of keyboard.

In TABLE D2, the column KEY refer to alphanumeric letters on the keytops as in Fig D1.

Figure D1 USA standard keyboard layout

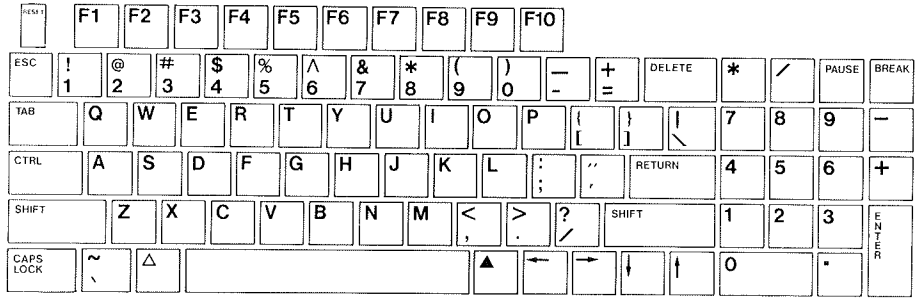


Table D1 the ASCII code of standard USA keyboard

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
F1	00	NUL	00	NUL	00	NUL	00	NUL
F2	01	SOH	01	SOH	01	SOH	01	SOH
F3	02	STX	02	STX	02	STX	02	STX
F4	03	ETX	03	ETX	03	ETX	03	ETX
F5	04	EOT	04	EOT	04	EOT	04	EOT
F6	05	ENQ	05	ENQ	05	ENQ	05	ENQ
F7	06	ACK	06	ACK	06	ACK	06	ACK
F8	07	BEL	07	BEL	07	BEL	07	BEL
F9	0C	FF	0C	FF	0C	FF	0C	FF
F10	18	CAN	18	CAN	18	CAN	18	CAN
0)	30	0	30	0	29	)	29	)
1 !	31	1	31	1	21	!	21	!
2 @	32	2	00	NUL	40	@	00	NUL
3 #	33	3	33	3	23	#	23	#
4 $	34	4	34	4	24	$	24	$
5 %	35	5	35	5	25	%	25	%
6 ^	36	6	1E	RS	5E	^	1E	RS
7 &	37	7	37	7	26	&	26	&
8 *	38	8	38	8	2A	*	2A	*
9 (	39	9	39	9	28	(	28	(
A	61	a	01	SOH	41	A	01	SOH
B	62	b	02	STX	42	B	02	STX
C	63	c	03	ETX	43	C	03	ETX
D	64	d	04	EOT	44	D	04	EOT
E	65	e	05	ENQ	45	E	05	ENQ
F	66	f	06	ACK	46	F	06	ACK
G	67	g	07	BEL	47	G	07	BEL
H	68	h	08	BS	48	H	08	BS
I	69	i	09	HT	49	I	09	HT
J	6A	j	0A	LF	4A	J	0A	LF
K	6B	k	0B	VT	4B	K	0B	VT
L	6C	l	0C	FF	4C	L	0C	FF
M	6D	m	0D	CR	4D	M	0D	CR
N	6E	n	0E	SO	4E	N	0E	SO
O	6F	o	0F	SI	4F	O	0F	SI

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
P	70	p	10	DLE	50	P	10	DLE
Q	71	q	11	DC1	51	Q	11	DC1
R	72	r	12	DC2	52	R	12	DC2
S	73	s	13	DC3	53	S	13	DC3
T	74	t	14	DC4	54	T	14	DC4
U	75	u	15	NAK	55	U	15	NAK
V	76	v	16	SYN	56	V	16	SYN
W	77	w	17	ETB	57	W	17	ETB
X	78	x	18	CAN	58	X	18	CAN
Y	79	y	19	EM	59	Y	19	EM
Z	7A	z	1A	SUB	5A	Z	1A	SUB
- —	2D	-	1F	US	5F	—	1F	US
= +	3D	=	3D	=	2B	+	2B	+
[{	5B	[	1B	ESC	7B	{	1B	ESC
] }	5D	]	1D	GS	7D	}	1D	GS
\ |	5C	\	1C	FS	7C	|	1C	FS
; :	3B	;	3B	;	3A	:	3A	:
' "	27	'	27	'	22	"	22	"
, <	2C	,	2C	,	3C	<	3C	<
. >	2E	.	2E	.	3E	>	3E	>
/ ?	2F	/	2F	/	3F	?	3F	?
` ~	60	`	60	`	7E	~	7E	~
SPACE	20	SP	20	SP	20	SP	20	SP
←	08	BS	08	BS	08	BS	08	BS
→	15	NAK	15	NAK	15	NAK	15	NAK
↓	0A	LF	0A	LF	0A	LF	0A	LF
↑	0B	VT	0B	VT	0B	VT	0B	VT
0	30	0	30	0	30	0	30	0
1	31	1	31	1	31	1	31	1
2	32	2	32	2	32	2	32	2
3	33	3	33	3	33	3	33	3
4	34	4	34	4	34	4	34	4
5	35	5	35	5	35	5	35	5
6	36	6	36	6	36	6	36	6
7	37	7	37	7	37	7	37	7
8	38	8	38	8	38	8	38	8
9	39	9	39	9	39	9	39	9
*	2A	*	2A	*	2A	*	2A	*
/	2F	/	2F	/	2F	/	2F	/
-	2D	-	2D	-	2D	-	2D	-
+	2B	+	2B	+	2B	+	2B	+
.	2E	.	2E	.	2E	.	2E	.
ESC	1B	ESC	1B	ESC	1B	ESC	1B	ESC
TAB	09	HT	09	HT	09	HT	09	HT
DELETE	7F	DEL	7F	DEL	7F	DEL	7F	DEL
RETURN	0D	CR	0D	CR	0D	CR	0D	CR
PAUSE	13	DC3	13	DC3	13	DC3	13	DC3
BREAK	03	ETX	03	ETX	03	ETX	03	ETX
ENTER	0D	CR	0D	CR	0D	CR	0D	CR

Figure D2 Simplified (DVORAK) keyboard layout

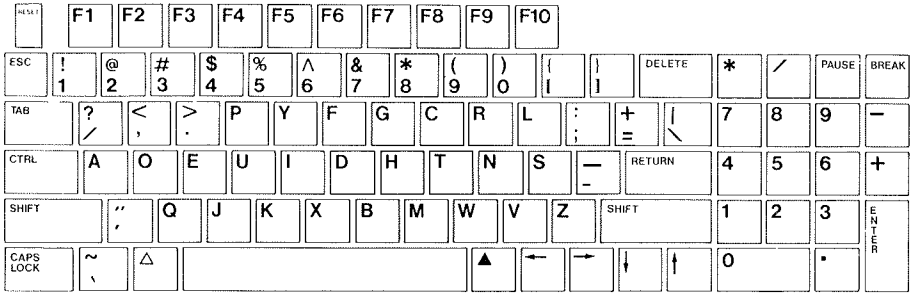


Table D2 the ASCII code of simplified (DVORAK) keyboard

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
F1	00	NUL	00	NUL	00	NUL	00	NUL
F2	01	SOH	01	SOH	01	SOH	01	SOH
F3	02	STX	02	STX	02	STX	02	STX
F4	03	ETX	03	ETX	03	ETX	03	ETX
F5	04	EOT	04	EOT	04	EOT	04	EOT
F6	05	ENQ	05	ENQ	05	ENQ	05	ENQ
F7	06	ACK	06	ACK	06	ACK	06	ACK
F8	07	BEL	07	BEL	07	BEL	07	BEL
F9	0C	FF	0C	FF	OC	FF	0C	FF
F10	18	CAN	18	CAN	18	CAN	18	CAN
0)	30	0	30	0	29	)	29	)
1 !	31	1	31	1	21	!	21	!
2 @	32	2	00	NUL	40	@	00	NUL
3 #	33	3	33	3	23	#	23	#
4 $	34	4	34	4	24	$	24	$
5 %	35	5	35	5	25	%	25	%
6 ^	36	6	1E	RS	5E	^	1E	RS
7 &	37	7	37	7	26	&	26	&
8 *	38	8	38	8	2A	*	2A	*
9 (	39	9	39	9	28	(	28	(
A	61	a	01	SOH	41	A	01	SOH
B	78	x	18	CAN	58	X	18	CAN
C	6A	j	0A	LF	4A	J	0A	LF
D	65	e	05	ENQ	45	E	05	ENQ
E	2E	.	2E	.	3E	>	3E	>
F	75	u	15	NAK	55	U	15	NAK
G	69	i	09	HT	49	I	09	HT
H	64	d	04	EOT	44	D	04	EOT
I	63	c	03	ETX	43	C	03	ETX
J	68	h	08	BS	48	H	08	BS
K	74	t	14	DC4	54	T	14	DC4
L	6E	n	0E	SO	4E	N	0E	SO
M	6D	m	0D	CR	4D	M	0D	CR
N	62	b	02	STX	42	B	02	STX
O	72	r	12	DC2	52	R	12	DC2

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
P	6C	l	0C	FF	4C	L	0C	FF
Q	2F	/	2F	/	3F	?	3F	?
R	70	p	10	DLE	50	P	10	DLE
S	6F	o	0F	SI	4F	O	0F	SI
T	79	y	19	EM	59	Y	19	EM
U	67	g	07	BEL	47	G	07	BEL
V	6B	k	0B	VT	4B	K	0B	VT
W	2C	,	2C	,	3C	<	3C	<
X	71	q	11	DC1	51	Q	11	DC1
Y	66	f	06	ACK	46	F	06	ACK
Z	27	'	27	'	22	"	22	"
- —	5B	[	1B	ESC	7B	{	1B	ESC
= +	5D	]	1D	GS	7D	}	1D	GS
[{	3B	;	3B	;	3A	:	3A	:
] }	3D	=	3D	=	2B	+	2B	+
\ |	5C	\	1C	FS	7C	|	1C	FS
; :	73	s	13	DC3	53	S	13	DC3
' "	2D	-	1F	US	5F	—	1F	US
, <	77	W	17	ETB	57	W	17	ETB
. >	76	v	16	SYN	56	V	16	SYN
/ ?	7A	z	1A	SUB	5A	Z	1A	SUB
` ~	60	`	60	`	7E	~	7E	~
SPACE	20	SP	20	SP	20	SP	20	SP
←	08	BS	08	BS	08	BS	08	BS
→	15	NAK	15	NAK	15	NAK	15	NAK
↑	0A	LF	0A	LF	0A	LF	0A	LF
↓	0B	VT	0B	VT	0B	VT	0B	VT
0	30	0	30	0	30	0	30	0
1	31	1	31	1	31	1	31	1
2	32	2	32	2	32	2	32	2
3	33	3	33	3	33	3	33	3
4	34	4	34	4	34	4	34	4
5	35	5	35	5	35	5	35	5
6	36	6	36	6	36	6	36	6
7	37	7	37	7	37	7	37	7
8	38	8	38	8	38	8	38	8
9	39	9	39	9	39	9	39	9
*	2A	*	2A	*	2A	*	2A	*
/	2F	/	2F	/	2F	/	2F	/
-	2D	-	2D	-	2D	-	2D	-
+	2B	+	2B	+	2B	+	2B	+
.	2E	.	2E	.	2E	.	2E	.
ESC	1B	ESC	1B	ESC	1B	ESC	1B	ESC
TAB	09	HT	09	HT	09	HT	09	HT
DELETE	7F	DEL	7F	DEL	7F	DEL	7F	DEL
RETURN	0D	CR	0D	CR	OD	CR	0D	CR
PAUSE	13	DC3	13	DC3	13	DC3	13	DC3
BREAK	03	ETX	03	ETX	03	ETX	03	ETX
ENTER	0D	CR	0D	CR	0D	CR	0D	CR

Figure D3 Italian version keyboard layout

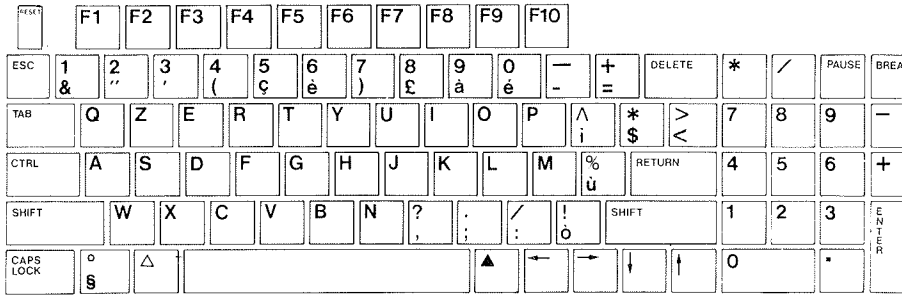


Table D3 the ASCII code of Italian version keyboard

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
F1	00	NUL	00	NUL	00	NUL	00	NUL
F2	01	SOH	01	SOH	01	SOH	01	SOH
F3	02	STX	02	STX	02	STX	02	STX
F4	03	ETX	03	ETX	03	ETX	03	ETX
F5	04	EOT	04	EOT	04	EOT	04	EOT
F6	05	ENQ	05	ENQ	05	ENQ	05	ENQ
F7	06	ACK	06	ACK	06	ACK	06	ACK
F8	07	BEL	07	BEL	07	BEL	07	BEL
F9	0C	FF	0C	FF	0C	FF	0C	FF
F10	18	CAN	18	CAN	18	CAN	18	CAN
é 0	5D	é	1D	GS	30	0	1D	GS
& 1	26	&	26	&	31	1	31	1
" 2	22	"	22	"	32	2	32	2
' 3	27	'	27	'	33	3	33	3
(4	28	(	28	(	34	4	34	4
ç 5	5C	ç	1C	FS	35	5	35	5
è 6	7D	è	7D	è	36	6	36	6
) 7	29	)	29	)	37	7	37	7
£ 8	23	£	23	£	38	8	38	8
à 9	7B	à	7B	à	39	9	39	9
A	61	a	01	SOH	41	A	01	SOH
B	62	b	02	STX	42	B	02	STX
C	63	c	03	ETX	43	C	03	ETX
D	64	d	04	EOT	44	D	04	EOT
E	65	e	05	ENQ	45	E	05	ENQ
F	66	f	06	ACK	46	F	06	ACK
G	67	g	07	BEL	47	G	07	BEL
H	68	h	08	BS	48	H	08	BS
I	69	i	09	HT	49	I	09	HT
J	6A	j	0A	LF	4A	J	0A	LF
K	6B	k	0B	VT	4B	K	0B	VT
L	6C	l	0C	FF	4C	L	0C	FF
M	6D	m	0D	CR	4D	M	0D	CR
N	6E	n	0E	SO	4E	N	0E	SO

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
O	6F	o	0F	SI	4F	O	0F	SI
P	70	p	10	DLE	50	P	10	DLE
Q	71	q	11	DC1	51	Q	11	DC1
R	72	r	12	DC2	52	R	12	DC2
S	73	s	13	DC3	53	S	13	DC3
T	74	t	14	DC4	54	T	14	DC4
U	75	u	15	NAK	55	U	15	NAK
V	76	v	16	SYN	56	V	16	SYN
W	77	w	17	ETB	57	W	17	ETB
X	78	x	18	CAN	58	X	18	CAN
Y	79	y	19	EM	59	Y	19	EM
Z	7A	z	1A	SUB	5A	Z	1A	SUB
- —	2D	-	1F	US	5F	—	1F	US
= +	3D	=	3D	=	2B	+	2B	+
ì ^	7E	ì	1E	RS	5E	^	1E	RS
$ *	24	$	24	$	2A	*	2A	*
< >	3C	<	3C	<	3E	>	3E	>
, ?	2C	,	2C	,	3F	?	3F	?
ù %	60	ù	60	ù	25	%	25	%
; .	3B	;	3B	;	2E	.	2E	.
: /	3A	:	3A	:	2F	/	2F	/
Ò !	7C	Ò	7C	Ò	21	!	21	!
§ ∘	40	§	00	NUL	5B	∘	1B	ESC
SPACE	20	SP	20	SP	20	SP	20	SP
←	08	BS	08	BS	08	BS	08	BS
→	15	NAK	15	NAK	15	NAK	15	NAK
↑	0A	LF	0A	LF	0A	LF	0A	LF
↓	0B	VT	0B	VT	0B	VT	0B	VT
0	30	0	30	0	30	0	30	0
1	31	1	31	1	31	1	31	1
2	32	2	32	2	32	2	32	2
3	33	3	33	3	33	3	33	3
4	34	4	34	4	34	4	34	4
5	35	5	35	5	35	5	35	5
6	36	6	36	6	36	6	36	6
7	37	7	37	7	37	7	37	7
8	38	8	38	8	38	8	38	8
9	39	9	39	9	39	9	39	9
*	2A	*	2A	*	2A	*	2A	*
/	2F	/	2F	/	2F	/	2F	/
-	2D	-	2D	-	2D	-	2D	-
+	2B	+	2B	+	2B	+	2B	+
.	2E	.	2E	.	2E	.	2E	.
ESC	1B	ESC	1B	ESC	1B	ESC	1B	ESC
TAB	09	HT	09	HT	09	HT	09	HT
DELETE	7F	DEL	7F	DEL	7F	DEL	7F	DEL
RETURN	0D	CR	0D	CR	0D	CR	0D	CR
PAUSE	13	DC3	13	DC3	13	DC3	13	DC3
BREAK	03	ETX	03	ETX	03	ETX	03	ETX
ENTER	0D	CR	0D	CR	0D	CR	0D	CR

Figure D4 French version keyboard layout

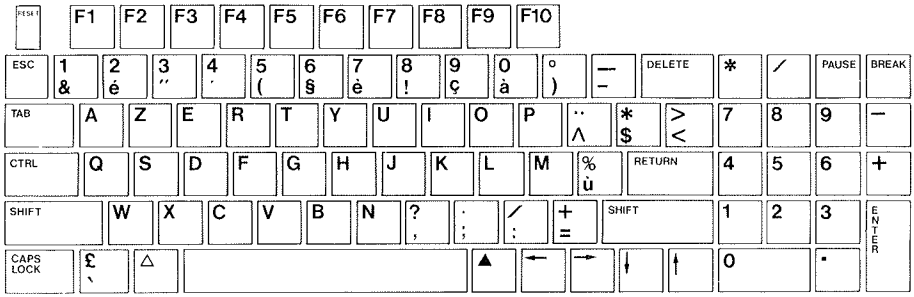


Table D4 the ASCII code of French version keyboard

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
F1	00	NUL	00	NUL	00	NUL	00	NUL
F2	01	SOH	01	SOH	01	SOH	01	SOH
F3	02	STX	02	STX	02	STX	02	STX
F4	03	ETX	03	ETX	03	ETX	03	ETX
F5	04	EOT	04	EOT	04	EOT	04	EOT
F6	05	ENQ	05	ENQ	05	ENQ	05	ENQ
F7	06	ACK	06	ACK	06	ACK	06	ACK
F8	07	BEL	07	BEL	07	BEL	07	BEL
F9	0C	FF	0C	FF	0C	FF	0C	FF
F10	18	CAN	18	CAN	18	CAN	18	CAN
à 0	40	à	00	NUL	30	0	00	NUL
&1	26	&	26	&	31	1	31	1
é 2	7B	é	7B	é	32	2	32	2
" 3	22	"	22	"	33	3	33	3
' 4	27	'	27	'	34	4	34	4
(5	28	(	28	(	35	5	35	5
§ 6	5D	§	1D	GS	36	6	1D	GS
è 7	7D	è	7D	è	37	7	37	7
! 8	21	!	21	!	38	8	38	8
ç 9	5C	ç	1C	FS	39	9	1C	FS
A	61	a	01	SOH	41	A	01	SOH
B	62	b	02	STX	42	B	02	STX
C	63	c	03	ETX	43	C	03	ETX
D	64	d	04	EOT	44	D	04	EOT
E	65	e	05	ENQ	45	E	05	ENQ
F	66	f	06	ACK	46	F	06	ACK
G	67	g	07	BEL	47	G	07	BEL
H	68	h	08	BS	48	H	08	BS
I	69	i	09	HT	49	I	09	HT
J	6A	j	0A	LF	4A	J	0A	LF
K	6B	k	0B	VT	4B	K	0B	VT
L	6C	l	0C	FF	4C	L	0C	FF
M	6D	m	0D	CR	4D	M	0D	CR
N	6E	n	0E	SO	4E	N	0E	SO

KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
O	6F	o	0F	SI	4F	O	0F	SI
P	70	p	10	DLE	50	P	10	DLE
Q	71	q	11	DC1	51	Q	11	DC1
R	72	r	12	DC2	52	R	12	DC2
S	73	s	13	DC3	53	S	13	DC3
T	74	t	14	DC4	54	T	14	DC4
U	75	u	15	NAK	55	U	15	NAK
V	76	v	16	SYN	56	V	16	SYN
W	77	w	17	ETB	57	W	17	ETB
X	78	x	18	CAN	58	X	18	CAN
Y	79	y	19	EM	59	Y	19	EM
Z	7A	z	1A	SUB	5A	Z	1A	SUB
) °	29	)	1B	ESC	5B	°	1B	ESC
- —	2D	-	1F	US	5F	—	1F	US
^ ..	5E	^	1E	RS	7E	..	1E	RS
$ *	24	$	24	$	2A	*	2A	*
< >	3C	<	3C	<	3E	>	3E	>
, ?	2C	,	2C	,	3F	?	3F	?
ù %	7C	ù	7C	ù	25	%	25	%
; .	3B	;	3B	;	2E	.	2E	.
: /	3A	:	3A	:	2F	/	2F	/
= +	3D	=	3D	=	2B	+	2B	+
` £	60	`	60	`	23	£	23	£
SPACE	20	SP	20	SP	20	SP	20	SP
←	08	BS	08	BS	08	BS	08	BS
→	15	NAK	15	NAK	15	NAK	15	NAK
↑	0A	LF	0A	LF	0A	LF	0A	LF
↓	0B	VT	0B	VT	0B	VT	0B	VT
0	30	0	30	0	30	0	30	0
1	31	1	31	1	31	1	31	1
2	32	2	32	2	32	2	32	2
3	33	3	33	3	33	3	33	3
4	34	4	34	4	34	4	34	4
5	35	5	35	5	35	5	35	5
6	36	6	36	6	36	6	36	6
7	37	7	37	7	37	7	37	7
8	38	8	38	8	38	8	38	8
9	39	9	39	9	39	9	39	9
*	2A	*	2A	*	2A	*	2A	*
/	2F	/	2F	/	2F	/	2F	/
-	2D	-	2D	-	2D	-	2D	-
+	2B	+	2B	+	2B	+	2B	+
.	2E	.	2E	.	2E	.	2E	.
ESC	1B	ESC	1B	ESC	1B	ESC	1B	ESC
TAB	09	HT	09	HT	09	HT	09	HT
DELETE	7F	DEL	7F	DEL	7F	DEL	7F	DEL
RETURN	0D	CR	0D	CR	0D	CR	0D	CR
PAUSE	13	DC3	13	DC3	13	DC3	13	DC3
BREAK	03	ETX	03	ETX	03	ETX	03	ETX
ENTER	0D	CR	0D	CR	0D	CR	0D	CR

Figure D5 German version keyboard layout

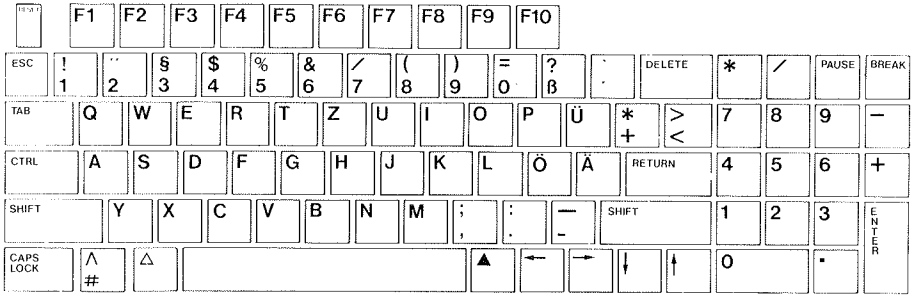
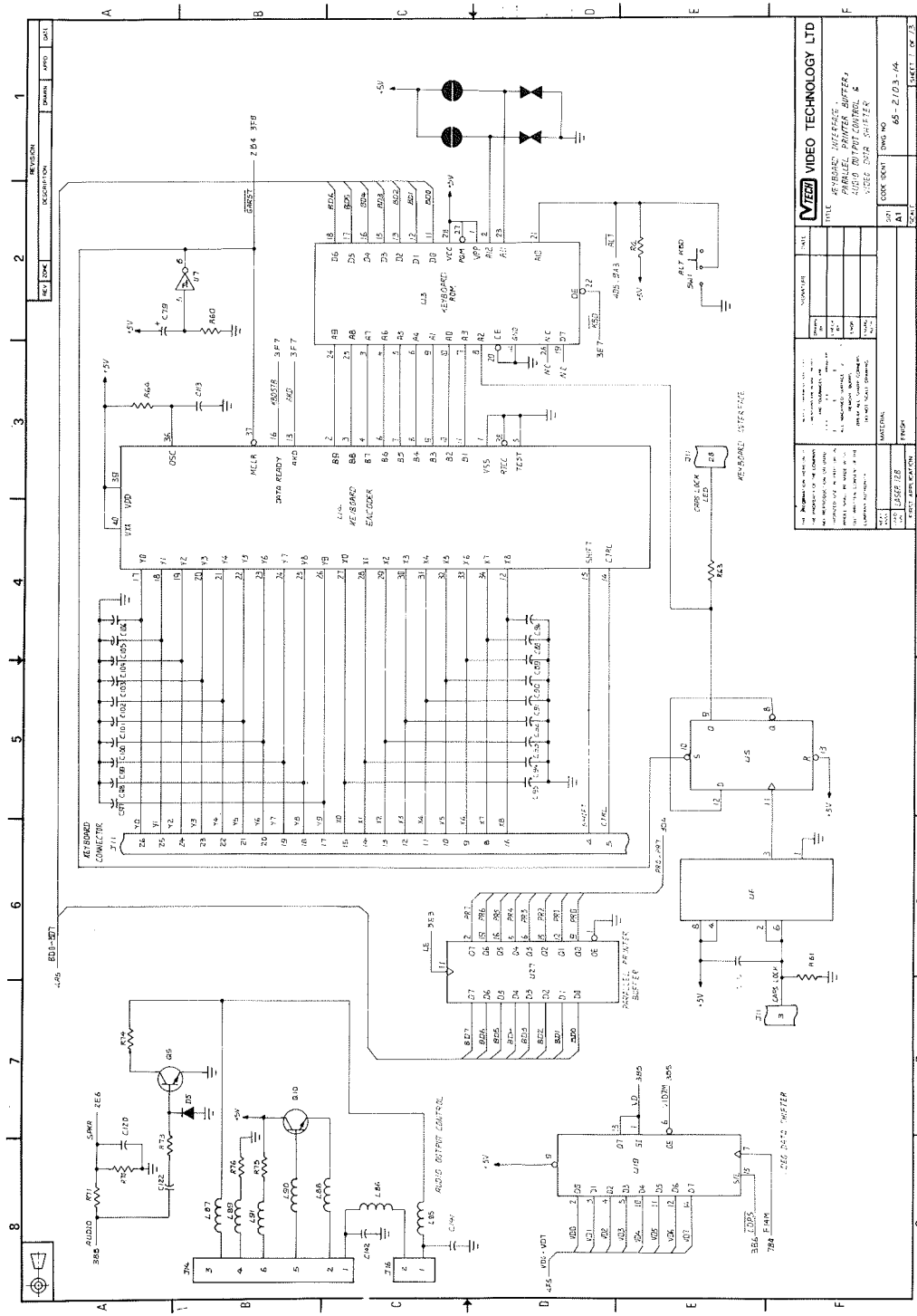


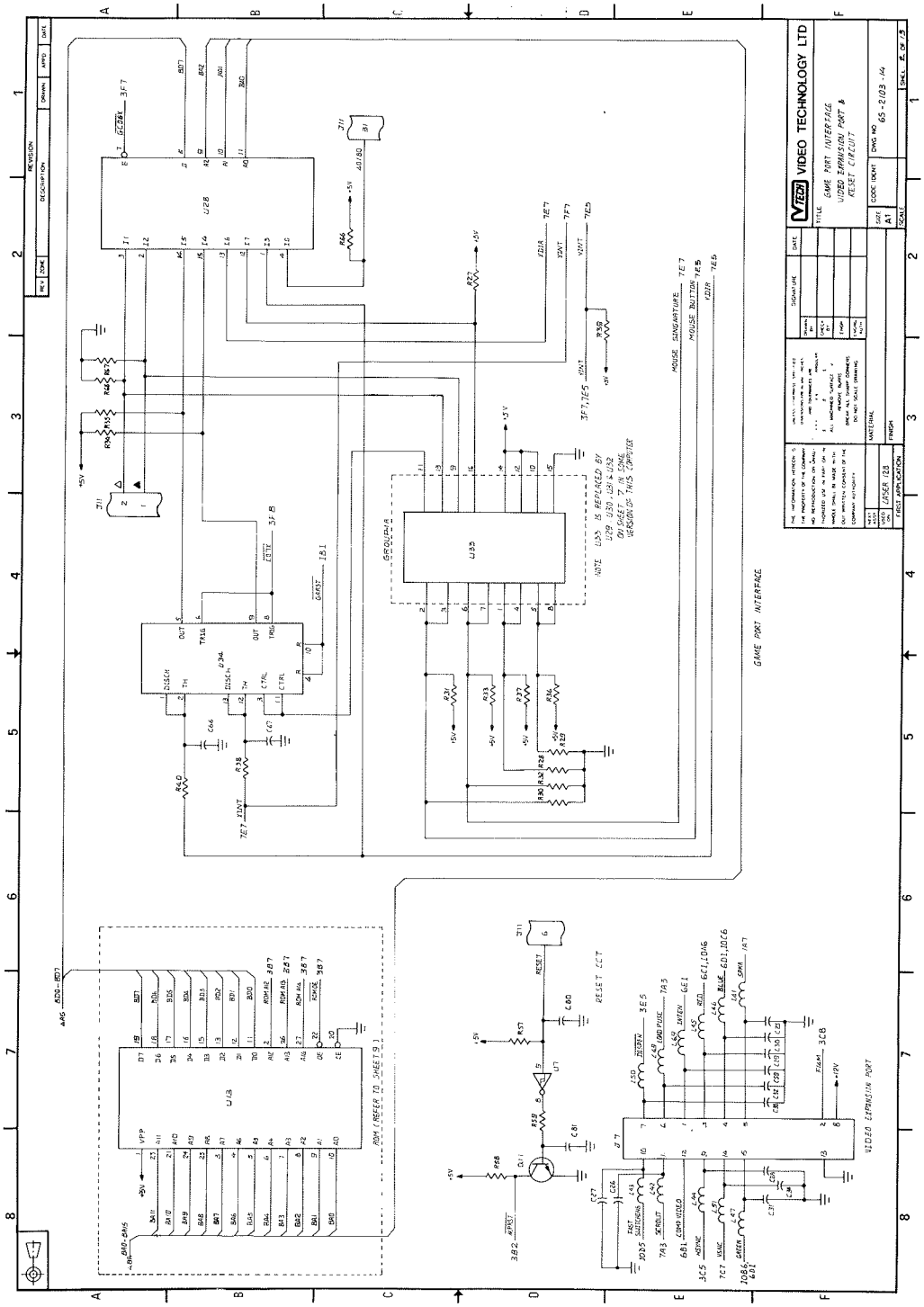
Table D5 The ASCII code of German version keyboard

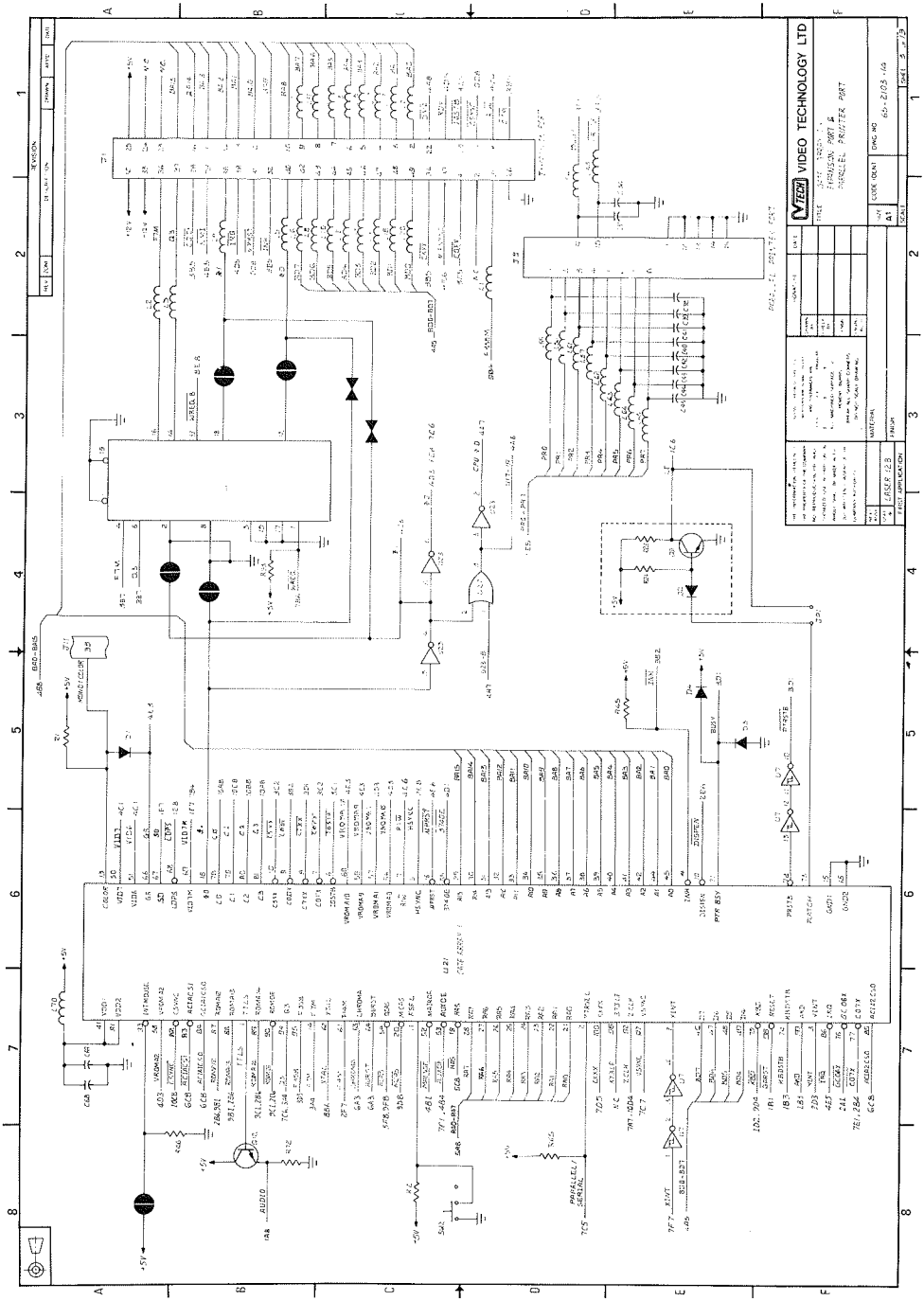
KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
F1	00	NUL	00	NUL	00	NUL	00	NUL
F2	01	SOH	01	SOH	01	SOH	01	SOH
F3	02	STX	02	STX	02	STX	02	STX
F4	03	ETX	03	ETX	03	ETX	03	ETX
F5	04	EOT	04	EOT	04	EOT	04	EOT
F6	05	ENQ	05	ENQ	05	ENQ	05	ENQ
F7	06	ACK	06	ACK	06	ACK	06	ACK
F8	07	BEL	07	BEL	07	BEL	07	BEL
F9	0C	FF	0C	FF	0C	FF	0C	FF
F10	18	CAN	18	CAN	18	CAN	18	CAN
0 =	30	0	30	0	3D	=	3D	=
1 !	31	1	31	1	21	!	21	!
2 "	32	2	32	2	22	"	22	"
3 §	33	3	00	NUL	40	§	00	NUL
4 $	34	4	34	4	24	$	24	$
5 %	35	5	35	5	25	%	25	%
6 &	36	6	36	6	26	&	26	&
7 /	37	7	37	7	2F	/	2F	/
8 (	38	8	38	8	28	(	28	(
9)	39	9	39	9	29	)	29	)
A	61	a	01	SOH	41	A	01	SOH
B	62	b	02	STX	42	B	02	STX
C	63	c	03	ETX	43	C	03	ETX
D	64	d	04	EOT	44	D	04	EOT
E	65	e	05	ENQ	45	E	05	ENQ
F	66	f	06	ACK	46	F	06	ACK
G	67	g	07	BEL	47	G	07	BEL
H	68	h	08	BS	48	H	08	BS
I	69	i	09	HT	49	I	09	HT
J	6A	j	0A	LF	4A	J	0A	LF
K	6B	k	0B	VT	4B	K	0B	VT
L	6C	l	0C	FF	4C	L	0C	FF
M	6D	m	0D	CR	4D	M	0D	CR
N	6E	n	0E	SO	4E	N	0E	SO

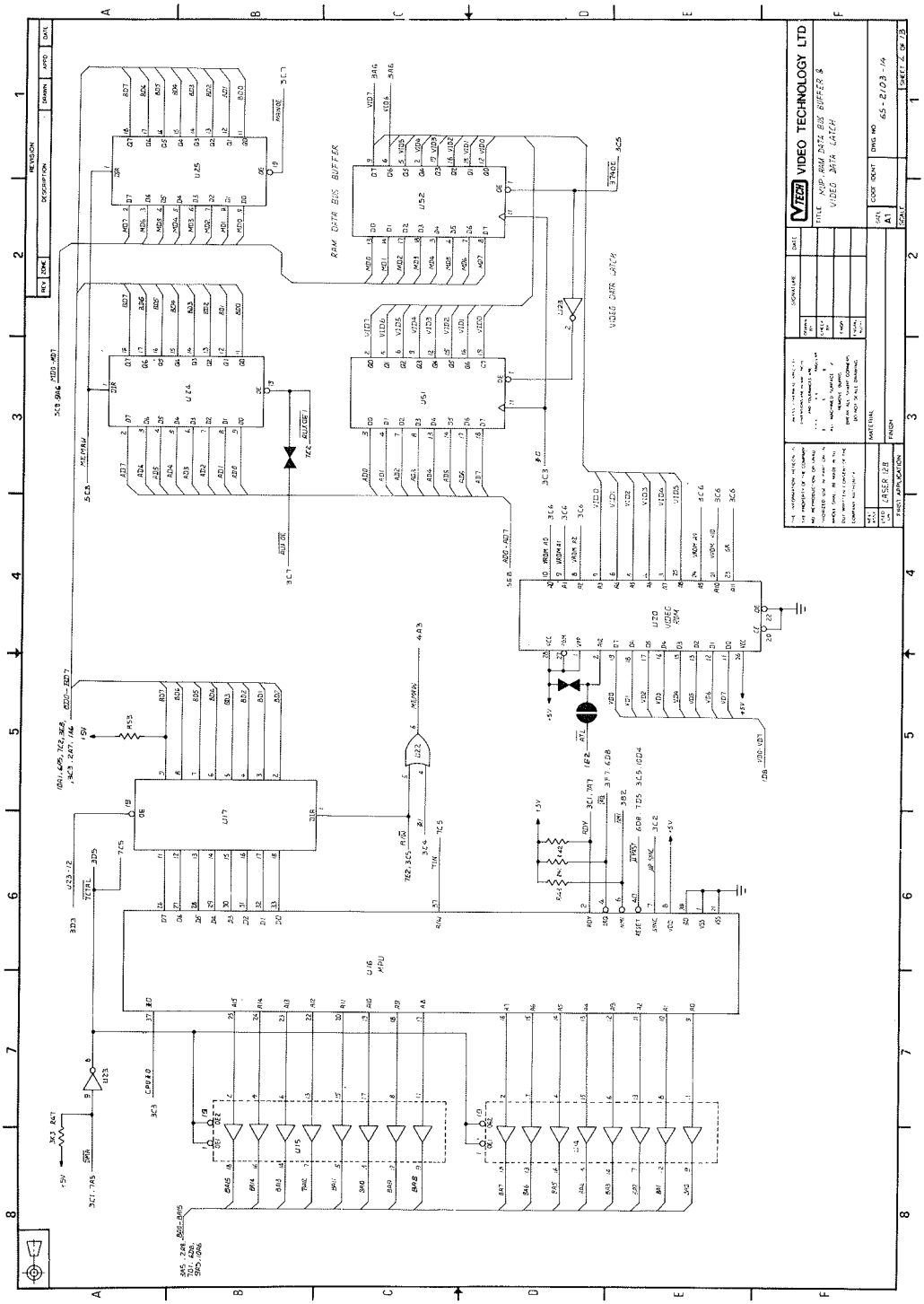
KEY	NORM	CHAR	CTRL	CHAR	SHIFT	CHAR	BOTH	CHAR
O	6F	o	0F	SI	4F	O	0F	SI
P	70	p	10	DLE	50	P	10	DLE
Q	71	q	11	DC1	51	Q	11	DC1
R	72	r	12	DC2	52	R	12	DC2
S	73	s	13	DC3	53	S	13	DC3
T	74	t	14	DC4	54	T	14	DC4
U	75	u	15	NAK	55	U	15	NAK
V	76	v	16	SYN	56	V	16	SYN
W	77	w	17	ETB	57	W	17	ETB
X	78	x	18	CAN	58	X	18	CAN
Y	79	y	19	SUB	59	Y	19	SUB
Z	7A	z	1A	US	5A	Z	1A	US
ß?	7E	ß	7E	ß	3F	?	3F	?
´ `	27	´	27	´	60	`	60	`
Ü	7D	Ü	1D	GS	5D	Ü	1D	GS
+ *	2B	+	2B	+	2A	*	2A	*
< >	3C	<	3C	<	3E	>	3E	>
Ö	7C	Ö	1C	FS	5C	Ö	1C	FS
Ä	7B	Ä	1B	ESC	5B	Ä	1B	ESC
, ;	2C	,	2C	,	3B	;	3B	;
. :	2E	.	2E	.	3A	:	3A	:
- —	2D	-	1F	US	5F	—	1F	US
# ^	23	#	1E	RS	5E	^	1E	RS
SPACE	20	SP	20	SP	20	SP	20	SP
←	08	BS	08	BS	08	BS	08	BS
→	15	NAK	15	NAK	15	NAK	15	NAK
↓	0A	LF	0A	LF	0A	LF	0A	LF
↑	0B	VT	0B	VT	0B	VT	0B	VT
0	30	0	30	0	30	0	30	0
1	31	1	31	1	31	1	31	1
2	32	2	32	2	32	2	32	2
3	33	3	33	3	33	3	33	3
4	34	4	34	4	34	4	34	4
5	35	5	35	5	35	5	35	5
6	36	6	36	6	36	6	36	6
7	37	7	37	7	37	7	37	7
8	38	8	38	8	38	8	38	8
9	39	9	39	9	39	9	39	9
*	2A	*	2A	*	2A	*	2A	*
/	2F	/	2F	/	2F	/	2F	/
-	2D	-	2D	-	2D	-	2D	-
+	2B	+	2B	+	2B	+	2B	+
.	2E	.	2E	.	2E	.	2E	.
ESC	1B	ESC	1B	ESC	1B	ESC	1B	ESC
TAB	09	HT	09	HT	09	HT	09	HT
DELETE	7F	DEL	7F	DEL	7F	DEL	7F	DEL
RETURN	0D	CR	0D	CR	0D	CR	0D	CR
PAUSE	13	DC3	13	DC3	13	DC3	13	DC3
BREAK	03	ETX	03	ETX	03	ETX	03	ETX
ENTER	0D	CR	0D	CR	0D	CR	0D	CR

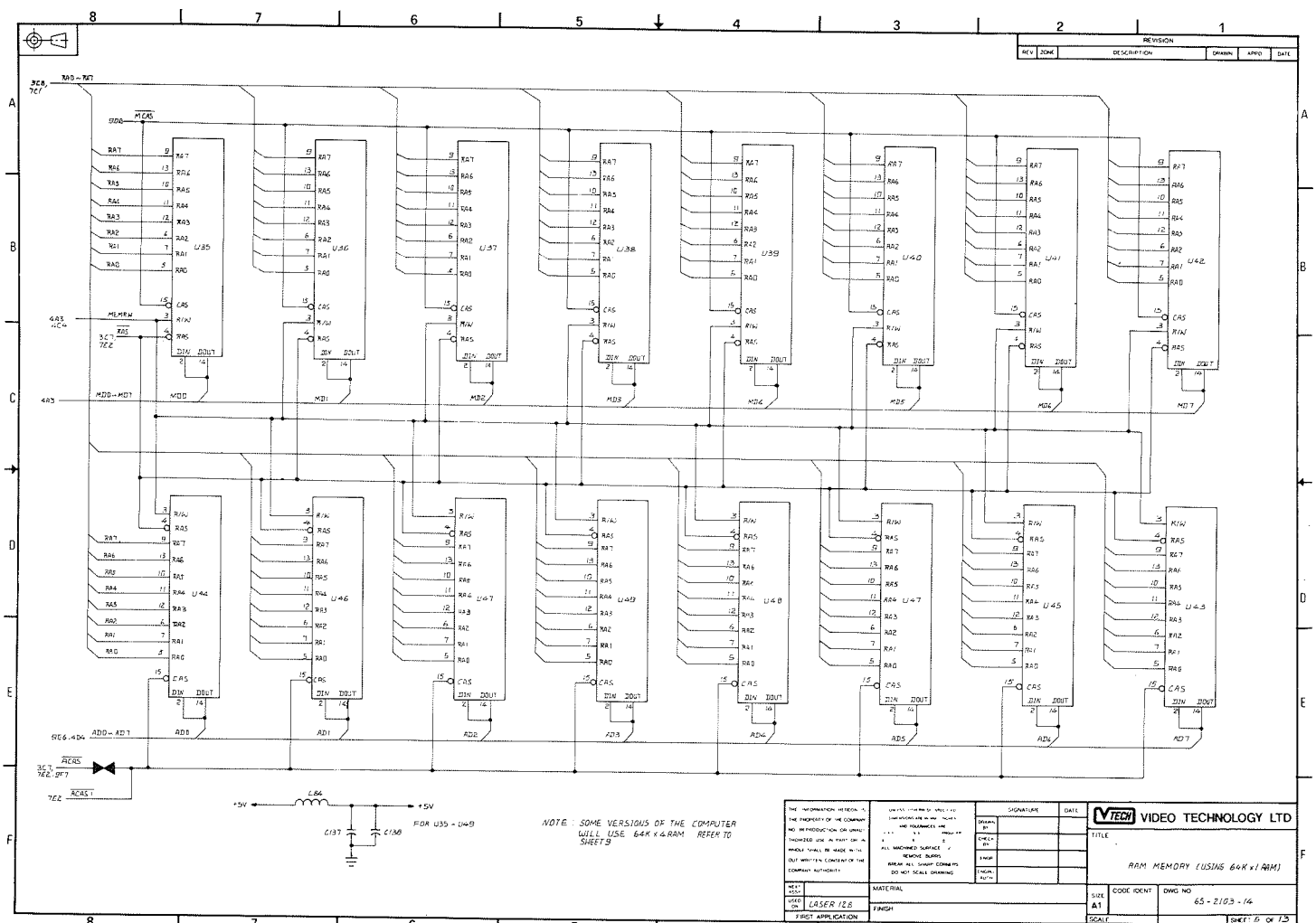
SCHEMATICS

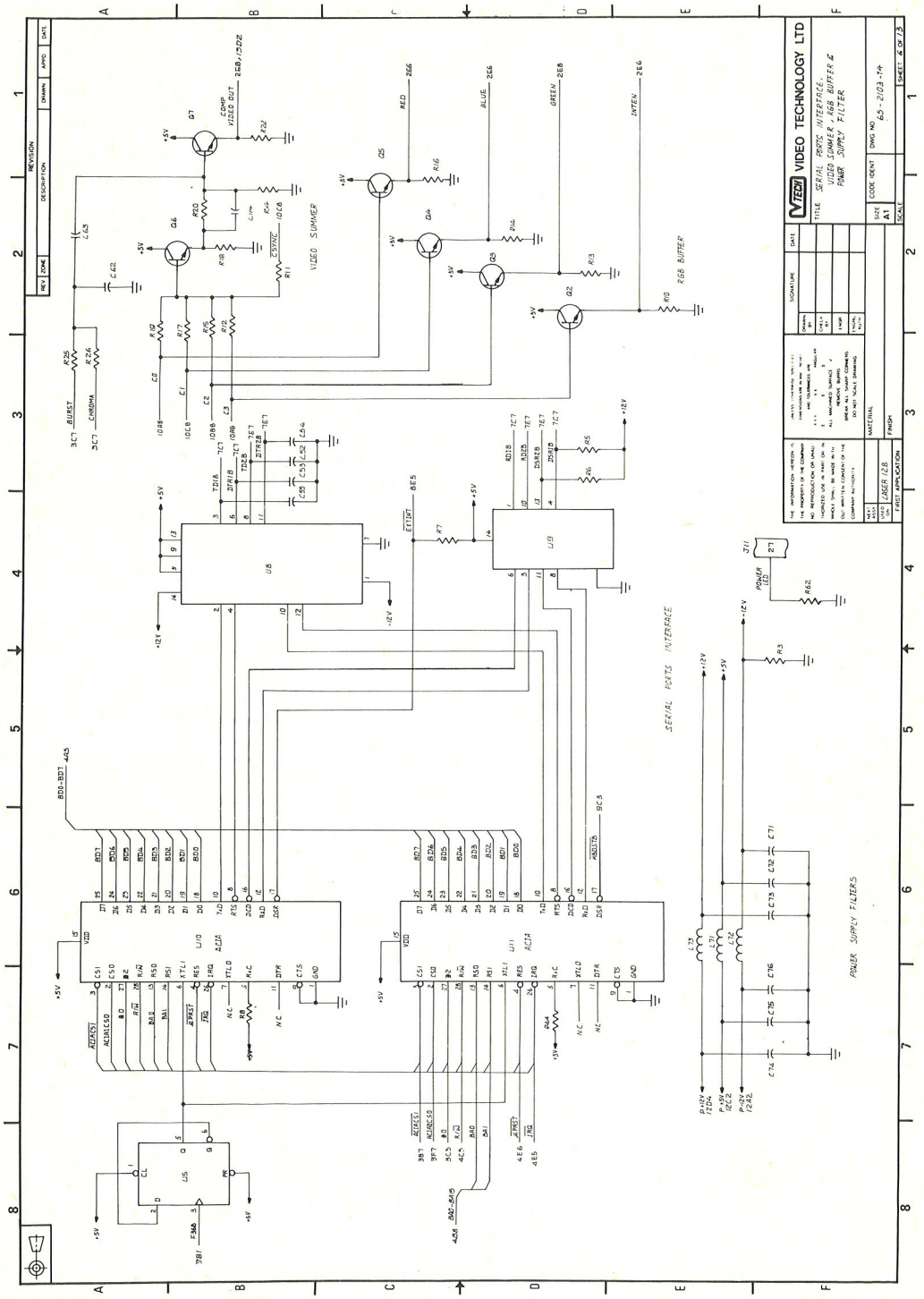


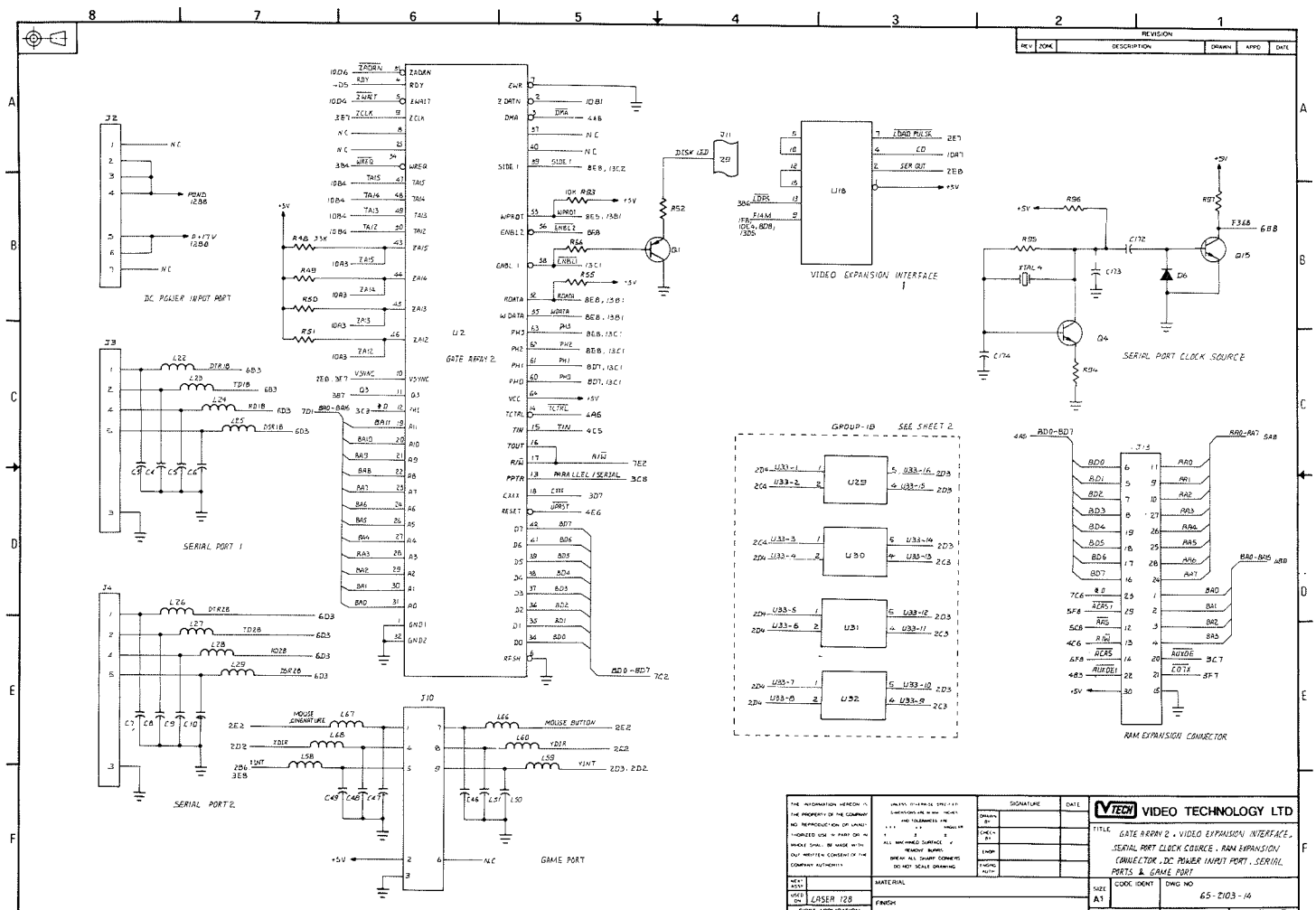

VIDEO TECHNOLOGY LTD
GAME PORT INTERFACE
VIDEO EXPANSION PORT B
RESET CIRCUIT
65-2103-44
GAME PORT INTERFACE
VIDEO EXPANSION PORT
RESET CCT
U28
U35
U36
U33
U34
U1
U7
U11
U12
U13
U14
U15
U16
U17
U18
U19
U20
U21
U22
U23
U24
U25
U26
U27
U29
U30
U31
U32
Q11
J11
J7
J8
J9
J10
R38
R39
R40
R41
R42
R43
R44
R45
R46
R47
R48
R49
R50
R51
R52
R53
R54
R55
R56
R57
R58
R59
C60
C61
C62
C63
C64
C65
C66
C67
C68
C69
C70
C71
C72
C73
C74
C75
C76
C77
C78
C79
C80
C81
C82
C83
C84
C85
C86
C87
C88
C89
C90
C91
C92
C93
C94
C95
C96
C97
C98
C99


VIDEO TECHNOLOGY LTD
LASER 12B
FIRST APPLICATION

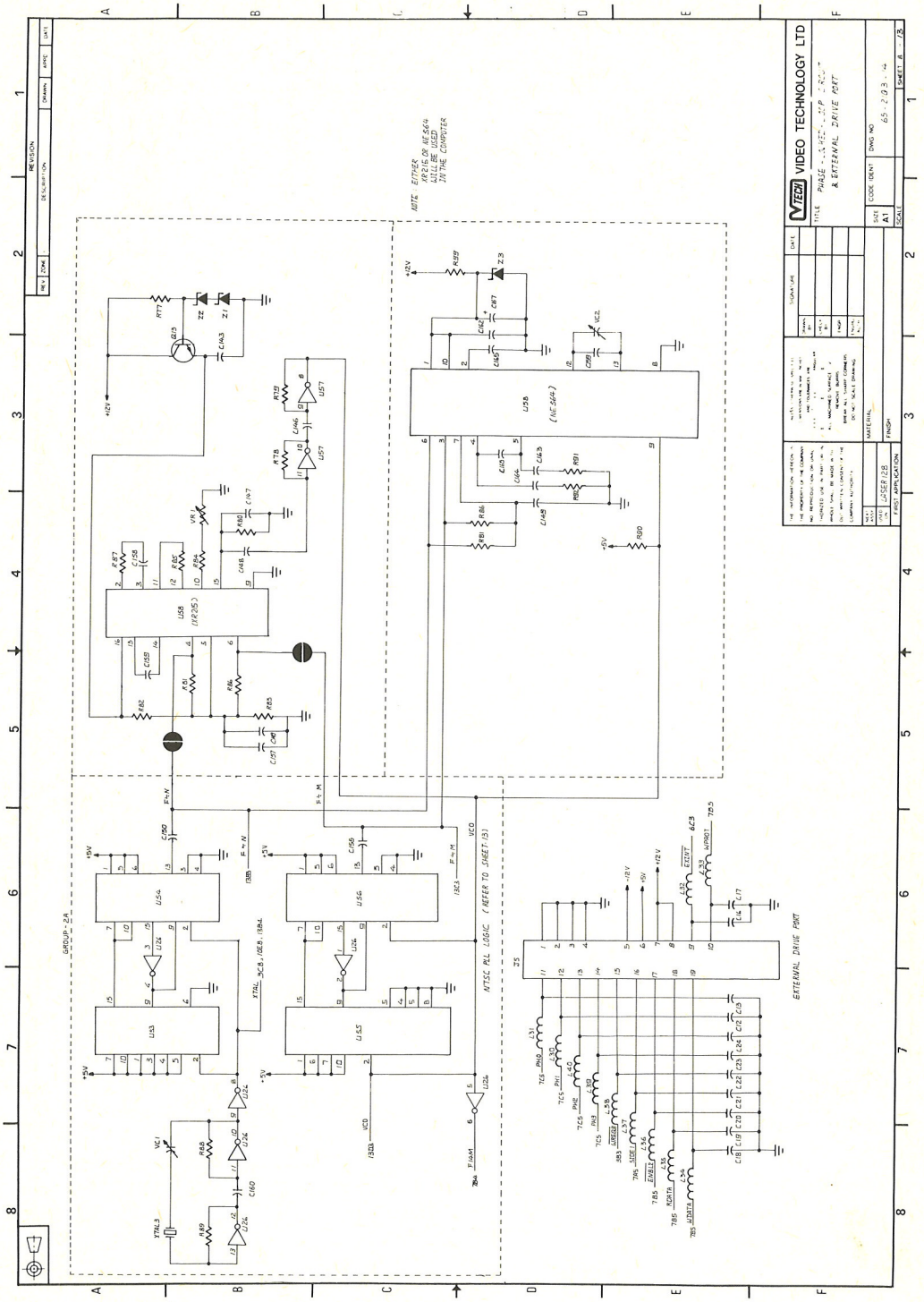

VIDEO TECHNOLOGY LTD
MUP, RAM DATA BUS BUFFER &
VIDEO DATA LATCH
RAM DATA BUS BUFFER
VIDEO DATA LATCH
U16
MPU
U17
U20
VIDEO
ROM
U24
U25
U51
U52
U15
U14
U23
U22
U23-12
REVISION
DESCRIPTION
DRAWN
APPD
DATE
SIGNATURE
MATERIAL
FINISH
FIRST APPLICATION
LASER 128
SHEET 4 OF 13
DWG NO
65-2103-14
SIZE
A1
SCALE

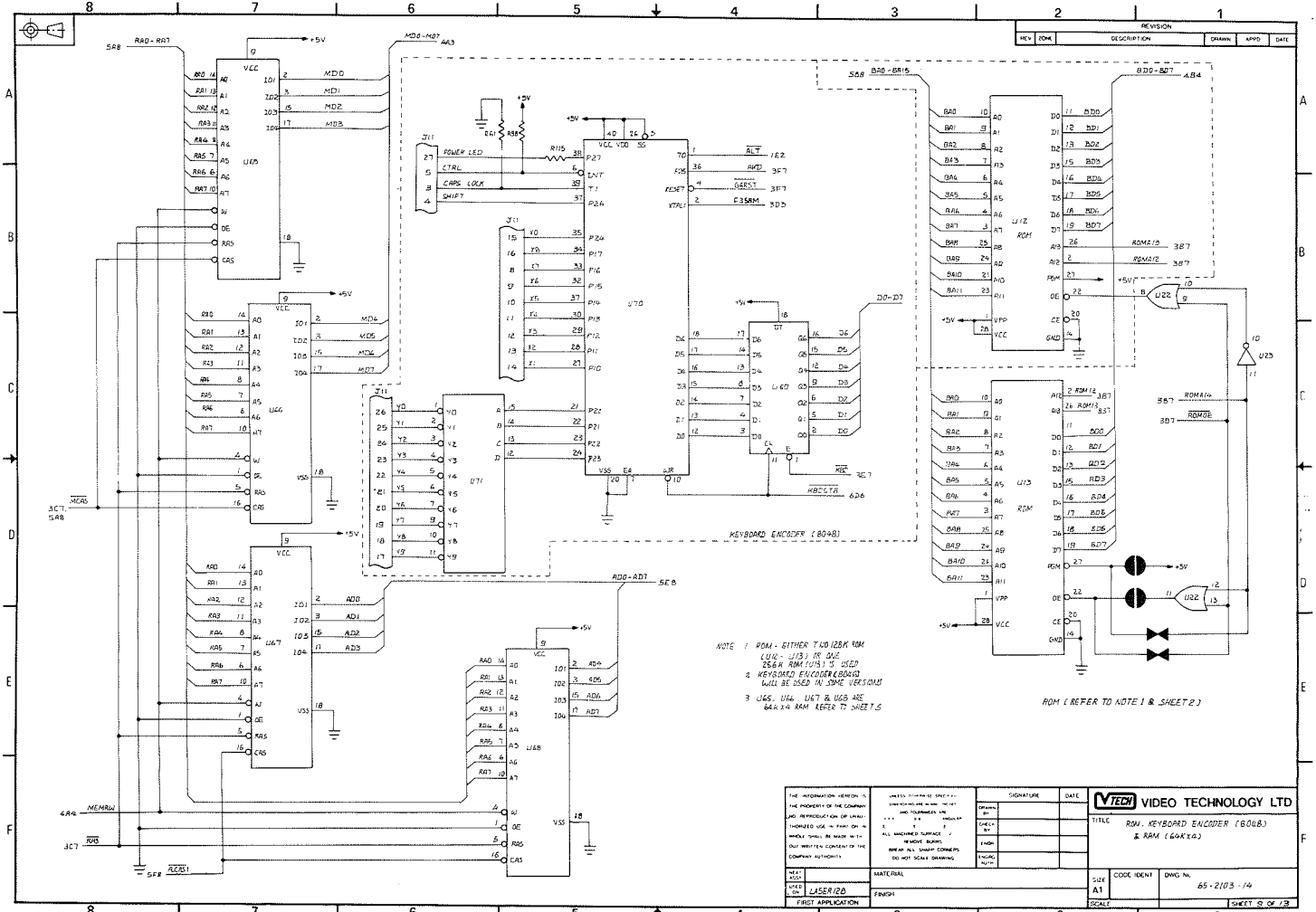

REVISION
REV
ZONE
DESCRIPTION
DRAWN
APPD
DATE
U35
U36
U37
U38
U39
U40
U41
U42
U44
U46
U47
U48
U45
U43
MEMRW
MCAS
MD0 - MD7
AD0 - AD7
ACAS
+5V
C84
C137
C138
FOR U35 - U49
NOTE : SOME VERSIONS OF THE COMPUTER WILL USE 64K x 4 RAM REFER TO SHEET 9
VIDEO TECHNOLOGY LTD
TITLE
RAM MEMORY (USING 64K x 1 RAM)
SIZE
A1
CODE IDENT
DWG NO
65 - 2103 - 14
SCALE
SHEET 6 OF 13
LASER 128
FIRST APPLICATION
MATERIAL
FINISH
SIGNATURE
DATE
DRAWN BY
CHECK BY
ENGR
ENGR. AUTH
DO NOT SCALE DRAWING

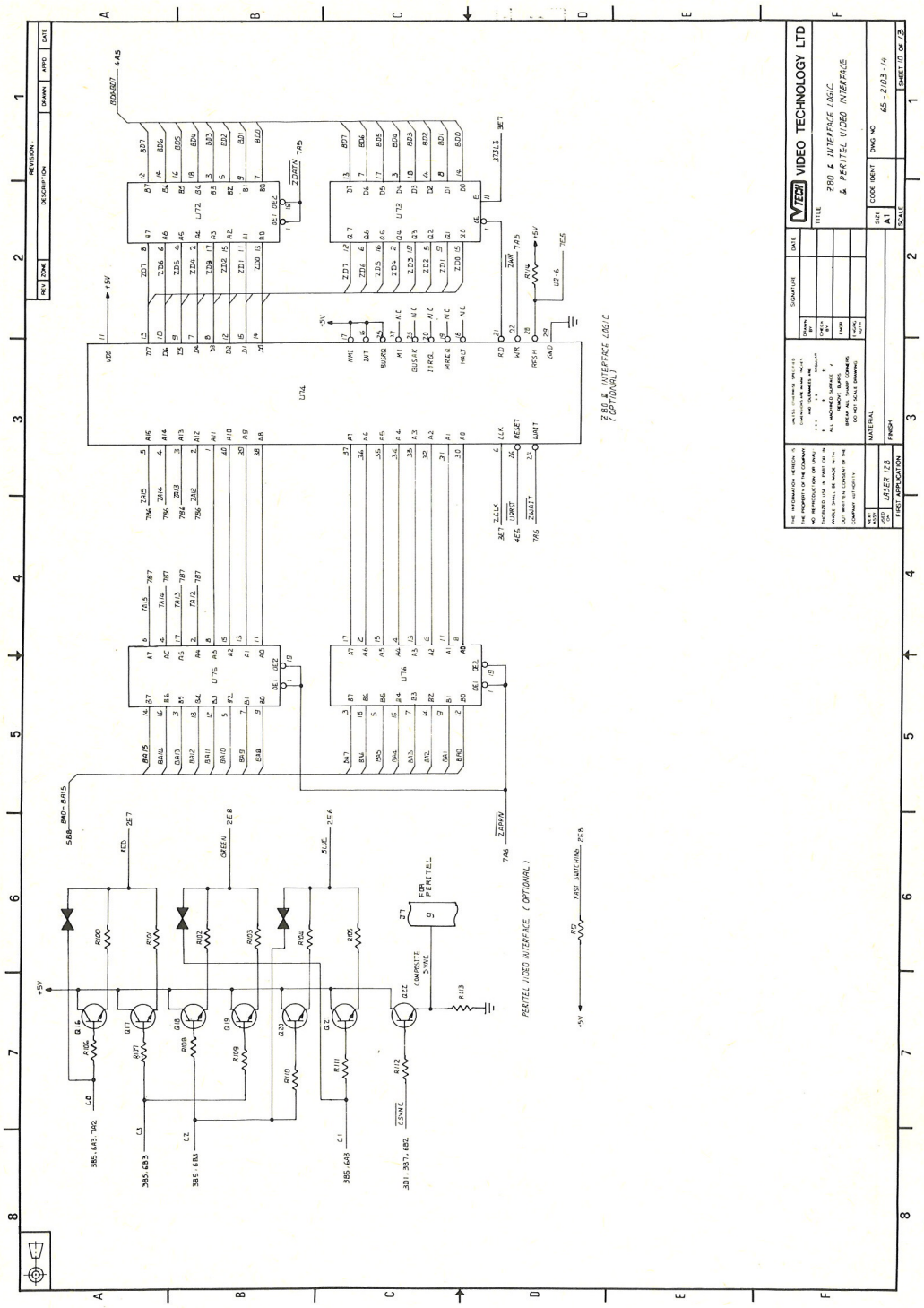

VIDEO TECHNOLOGY LTD
SERIAL PORTS INTERFACE
VIDEO SUMMER, RGB BUFFER &
POWER SUPPLY FILTER
65-2103-74
SHEET 4 OF 13
VIDEO SUMMER
RGB BUFFER
SERIAL PORTS INTERFACE
POWER SUPPLY FILTERS
FIRST APPLICATION

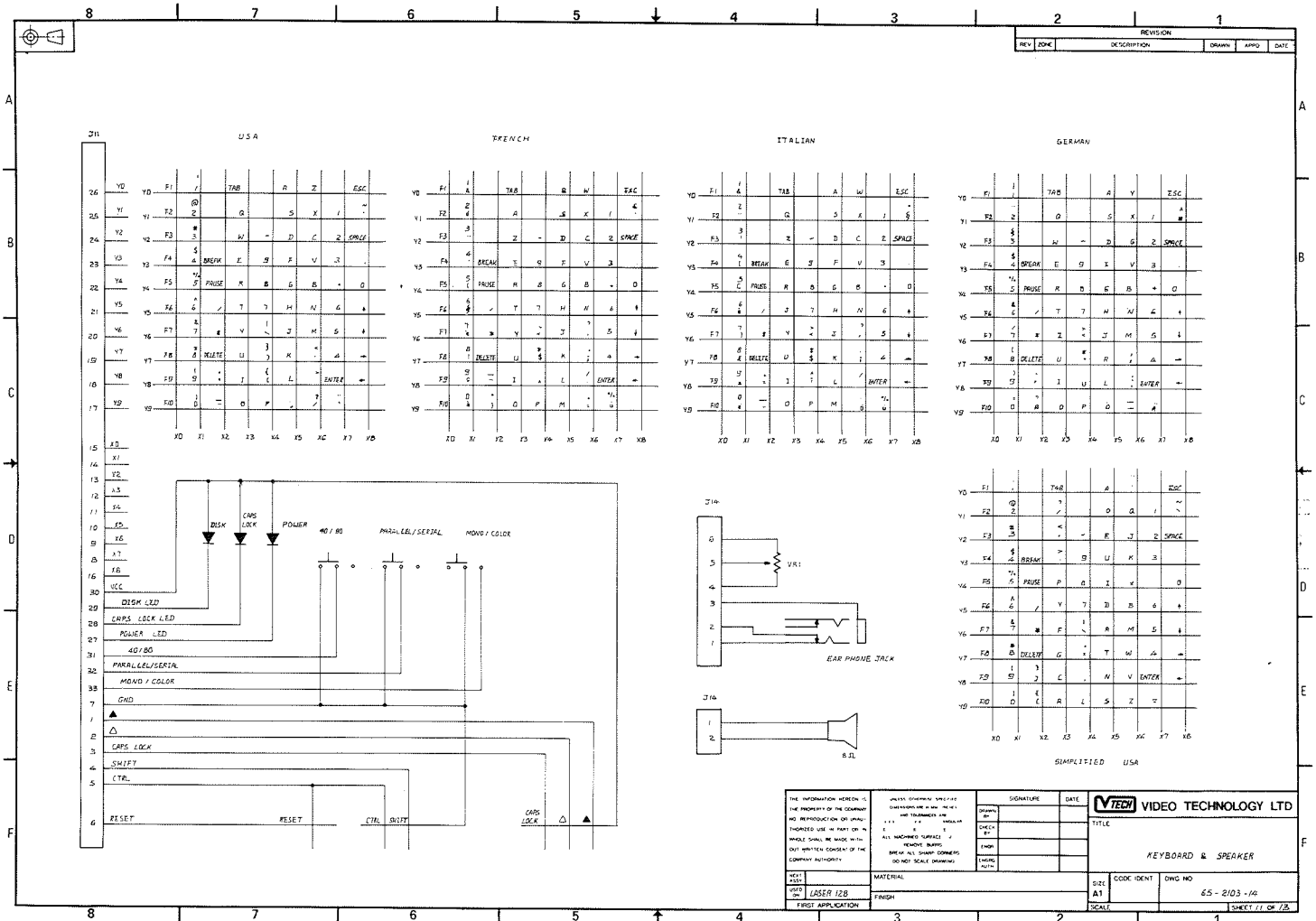

REVISION
REV ZONE DESCRIPTION DRAWN APPD DATE
J2
DC POWER INPUT PART
J3
SERIAL PORT 1
J4
SERIAL PORT 2
U2
GATE ARRAY 2
J11
DISK LED
VIDEO EXPANSION INTERFACE 1
U30
U31
U32
GROUP-1B SEE SHEET 2
SERIAL PORT CLOCK SOURCE
J13
RAM EXPANSION CONNECTOR
J10
GAME PORT
VTECH VIDEO TECHNOLOGY LTD
TITLE GATE ARRAY 2 . VIDEO EXPANSION INTERFACE . SERIAL PORT CLOCK SOURCE . RAM EXPANSION CONNECTOR . DC POWER INPUT PORT . SERIAL PORTS & GAME PORT
DWG NO 65-2103-14
SIZE A1
SHEET 7 OF 18
LASER 128
FIRST APPLICATION

E-7

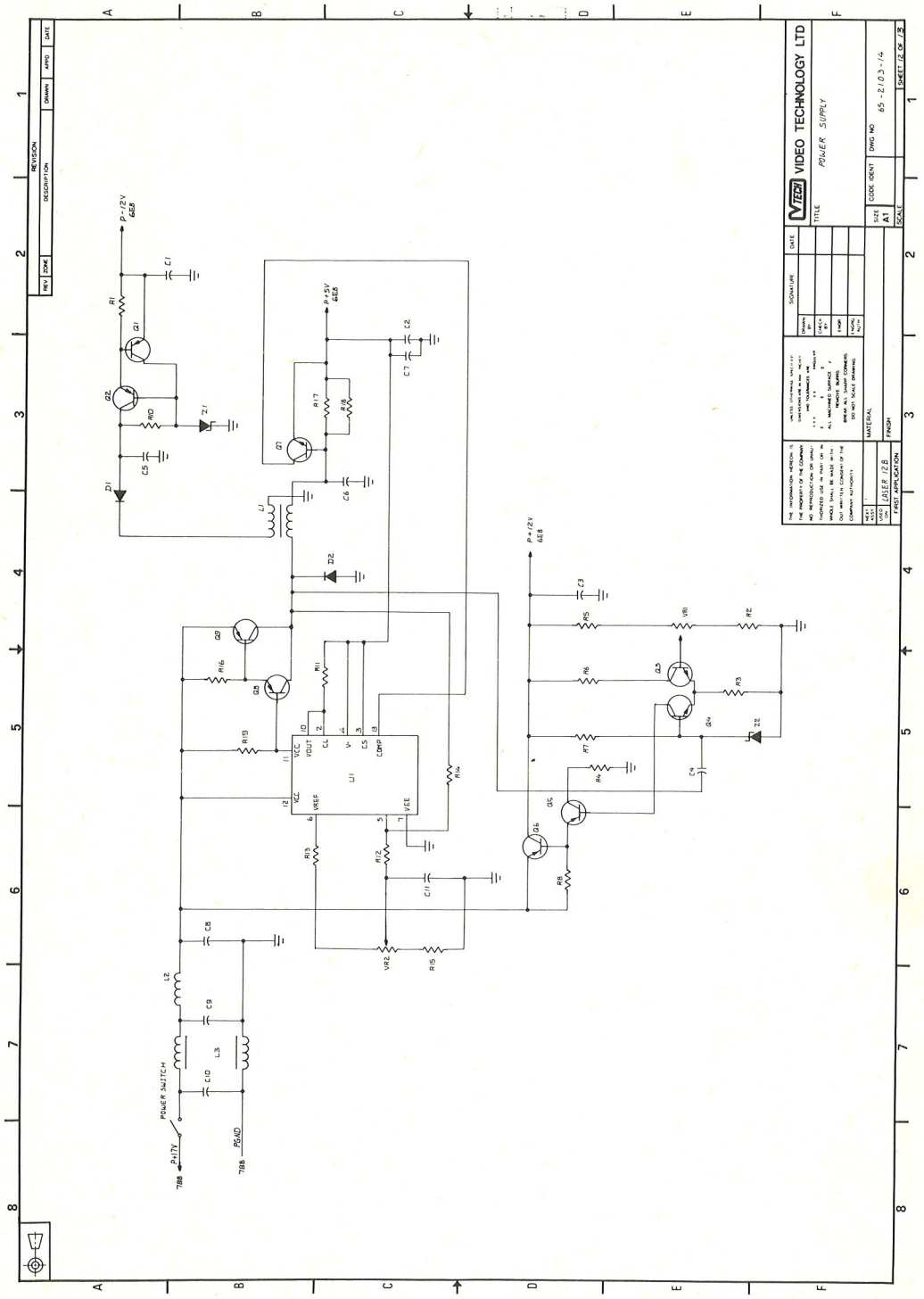

VIDEO TECHNOLOGY LTD
& EXTERNAL DRIVE PORT
NOTE: EITHER XR216 OR NE564 WILL BE USED IN THE COMPUTER
NTSC PLL LOGIC (REFER TO SHEET 13)
EXTERNAL DRIVE PORT
GROUP-2A
LASER 128
FIRST APPLICATION
SHEET 8 - 13

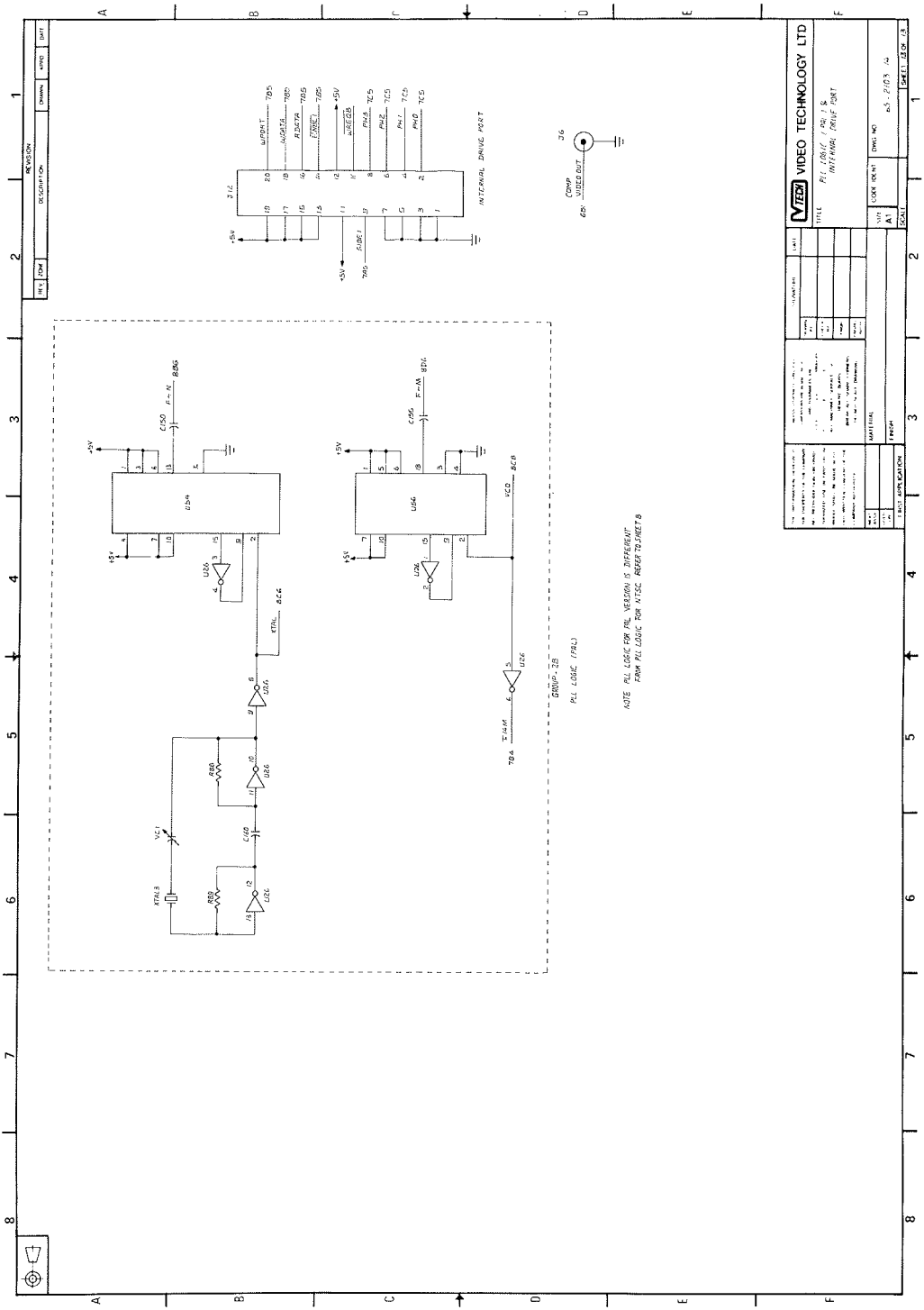

REVISION
REV ZONE DESCRIPTION DRAWN APPD DATE
KEYBOARD ENCODER (8048)
ROM (REFER TO NOTE 1 & SHEET 2)
NOTE 1 ROM - EITHER TWO 128K ROM (U12 - U13) IN ONE 256K ROM (U13) IS USED
2 KEYBOARD ENCODER (8048) WILL BE USED IN SOME VERSIONS
3 U65, U66, U67 & U68 ARE 64K x 4 RAM REFER TO SHEET 5
VIDEO TECHNOLOGY LTD
TITLE ROM, KEYBOARD ENCODER (8048) & RAM (64K x 4)
SIZE A1
CODE IDENT
DWG No. 65-2103-14
SHEET 9 OF 13
LASER128
FIRST APPLICATION
MATERIAL
FINISH
SIGNATURE
DATE
DRAWN BY
CHECK BY
ENGR
ENGRG MGR
THE INFORMATION HEREON IS THE PROPERTY OF THE COMPANY NO REPRODUCTION OR UNAUTHORIZED USE IN PART OR IN WHOLE SHALL BE MADE WITHOUT WRITTEN CONSENT OF THE COMPANY AUTHORITY
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN MM. THEREAFTER AND TOLERANCES ARE
ALL MACHINED SURFACE REMOVE BURRS
BREAK ALL SHARP CORNERS
DO NOT SCALE DRAWING
NEXT ASSY
USED ON


VIDEO TECHNOLOGY LTD
TITLE
Z80 & INTERFACE LOGIC
& PERITEL VIDEO INTERFACE
CODE IDENT
DWG NO
65-2103-14
SIZE
A1
SCALE
SHEET 10 OF 13
MATERIAL
FINISH
LASER 128
FIRST APPLICATION
Z80 & INTERFACE LOGIC
(OPTIONAL)
PERITEL VIDEO INTERFACE (OPTIONAL)
FOR PERITEL
COMPOSITE SYNC
FAST SWITCHING
REVISION
REV
ZONE
DESCRIPTION
DRAWN
APPD
DATE



REVISION
REV ZONE DESCRIPTION DRAWN APPD DATE
USA
FRENCH
ITALIAN
GERMAN
SIMPLIFIED USA
J11
DISK
CAPS LOCK
POWER
40 / 80
PARALLEL/SERIAL
MONO / COLOR
DISK LED
CAPS LOCK LED
POWER LED
40/80
PARALLEL/SERIAL
MONO / COLOR
GND
CAPS LOCK
SHIFT
CTRL
RESET
J14
VR1
EAR PHONE JACK
8 Ω
THE INFORMATION HEREON IS THE PROPERTY OF THE COMPANY NO REPRODUCTION OR UNAUTHORIZED USE IN PART OR IN WHOLE SHALL BE MADE WITHOUT WRITTEN CONSENT OF THE COMPANY AUTHORITY
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN MM. THREE AND TOLERANCES ARE
ALL MACHINED SURFACE REMOVE BURRS
BREAK ALL SHARP CORNERS
DO NOT SCALE DRAWING
SIGNATURE
DATE
DRAWN BY
CHECK BY
ENGR
ENGRG AUTH
VTECH VIDEO TECHNOLOGY LTD
TITLE
KEYBOARD & SPEAKER
NEXT ASSY
USED ON
LASER 128
FIRST APPLICATION
MATERIAL
FINISH
SIZE A1
CODE IDENT
DWG NO
65 - 2103 - 14
SCALE
SHEET 11 OF 13


VIDEO TECHNOLOGY LTD
POWER SUPPLY
65-2103-14
LASER 12B
FIRST APPLICATION
POWER SWITCH
P+17V
PGND
P-12V 6E8
P+5V 6E8
P+12V 6E8


VIDEO TECHNOLOGY LTD
INTERNAL DRIVE PORT
COMP VIDEO OUT
PLL LOGIC (PAL)
NOTE PLL LOGIC FOR PAL VERSION IS DIFFERENT
FROM PLL LOGIC FOR NTSC REFER TO SHEET 8

APPENDIX F

PARTS LIST OF THE COMPUTER

LIST OF ICs:

U1	74HCT244
U2	GA2
U3	2764 (KBD)
U4	KB-3600-PRO
U5	74HCT74
U6	NE555
U7	74HCT14
U8	1488
U9	1489
U10	6551
U11	6551
U12	ROM/27128
U13	ROM/27128/27256
U14	74HCT244
U15	74HCT244
U16	65C02
U17	74HCT245
U18	74HCT175
U19	74HCT166
U20	2764 (CG)
U21	GA1
U22	74HCT32
U23	74HCT04
U24	74HCT245
U25	74HCT245
U26	74S04
U27	74HCT374/74HCT373
U28	74HCT251
U29	4N27
U30	4N27
U31	4N27
U32	4N27
U33	1402
U34	NE556
U51	74HCT374
U52	74HCT374
U53	74HCT163
U54	74HCT163
U55	74HCT163

U56	74HCT163
U59	NE564
U65	50464
U66	50464
U67	50464
U68	50464
U77	74HCT244

LIST OF RESISTORS:
(UNIT IN OHM)

R1	3.3K
R2	3.3K
R3	10K
R4	3.3K
R5	15K
R6	15K
R7	3.3K
R8	3.3K
R9	220
R10	56
R11	470
R12	3.3K
R13	56
R14	56
R15	3.3K
R16	56
R17	3.3K
R18	1K
R19	3.3K
R20	3.3K
R21	3.3K
R22	100
R23	1K
R24	3.3K
R25	2.2K
R26	1K
R27	22K
R28	180
R29	430
R30	270
R31	270
R32	270
R33	270
R34	1K
R35	1K

R36	430
R37	180
R38	100
R39	10K
R40	100
R41	3.3K
R42	3.3K
R43	3.3K
R44	3.3K
R45	3.3K
R46	3.3K
R47	3.3K
R48	3.3K
R49	3.3K
R50	3.3K
R51	3.3K
R52	390
R53	10K
R54	10K
R55	10K
R56	3.3K
R57	4.7K
R58	390
R59	3.3K
R60	1K
R61	1M
R62	330
R63	330
R64	12K
R65	3.3K
R66	3.3K
R67	1K
R68	1K
R69	1K
R70	3.9K
R71	1K
R72	1K
R73	1K
R74	10
R75	1K
R76	560
R81	1K
R86	1K
R88	220
R89	220
R90	1K

R91	470
R92	220
R93	10K
R94	820
R95	100K
R96	2.2K
R97	5.6K
R99	330
R115	220
R116	470
R117	330

LIST OF CAPACITORS:

C1	220µF 16V
C2	0.04µF
C3	0.001µF
C4	0.001µF
C5	0.001µF
C6	0.001µF
C7	0.001µF
C8	0.001µF
C9	0.001µF
C10	0.001µF
C11	10µF 16V
C12	0.001µF
C13	0.001µF
C14	10µF 16V
C15	10µF 16V
C16	0.001µF
C17	0.001µF
C18	100pF
C19	100pF
C20	0.001µF
C21	0.001µF
C22	0.001µF
C23	0.001µF
C24	0.001µF
C25	0.001µF
C26	0.001µF
C27	0.001µF
C28	0.001µF
C29	0.001µF
C30	0.001µF
C31	0.001µF

C32	0.001μF
C33	0.001μF
C34	0.001μF
C35	0.001μF
C36	0.001μF
C37	0.001μF
C38	0.001μF
C39	0.001μF
C40	0.001μF
C41	0.001μF
C42	0.001μF
C43	0.001μF
C44	0.001μF
C45	0.001μF
C46	0.001μF
C47	0.001μF
C48	0.001μF
C49	0.001μF
C50	0.001μF
C51	0.001μF
C52	330pF
C53	330pF
C54	330pF
C55	330pF
C56	0.04μF
C58	100pF
C59	100pF
C60	0.04μF
C61	0.04μF
C62	47pF
C63	10pF
C64	0.04μF
C65	0.04μF
C66	0.027μF
C67	0.027μF
C68	100μF 10V
C69	0.04μF
C70	0.04μF
C71	100μF 16V
C72	100μF 16V
C73	100μF 16V
C74	0.04μF
C75	0.04μF
C76	0.04μF
C77	0.04μF
C78	0.04μF

C79	4.7μF 25V
C80	4.7μF 25V
C81	0.01μF
C82	0.04μF
C83	0.04μF
C84	0.04μF
C85	0.04μF
C86	0.04μF
C87	0.04μF
C88	0.04μF
C89	0.04μF
C90	0.04μF
C91	0.04μF
C92	0.04μF
C93	0.04μF
C94	0.04μF
C95	0.04μF
C96	0.04μF
C97	0.04μF
C98	0.04μF
C99	0.04μF
C100	0.04μF
C101	0.04μF
C102	0.04μF
C103	0.04μF
C104	0.04μF
C105	0.04μF
C106	0.04μF
C107	0.04μF
C108	0.04μF
C109	0.04μF
C110	0.04μF
C111	0.04μF
C112	0.04μF
C113	47pF
C114	47pF
C115	0.1μF
C116	0.04μF
C117	0.04μF
C118	0.04μF
C119	0.1μF
C120	0.1μF
C121	0.04μF
C122	0.1μF
C123	0.04μF
C139	0.04μF

C140	100µF 16V
C141	0.02µF
C142	0.02µF
C149	0.1µF
C150	0.1µF
C151	0.1µF
C152	0.1µF
C153	0.1µF
C154	0.1µF
C155	0.1µF
C159	15pF
C160	0.1µF
C162	0.04µF
C166	0.1µF
C167	10µF 16V
C168	0.04µF
C169	0.04µF
C170	0.04µF
C171	0.04µF
C172	0.04µF
C173	100pF
C174	15pF
C177	68pF
C178	0.04µF
C179	150pF
C180	0.04µF
C181	0.1µF
C182	0.1µF

LIST OF BEADS:

L1	3 1/2(E) OR 2 1/2(W)
L2	2 1/2(E) OR 1 1/2(W)
L3	3 1/2(E) OR 2 1/2(W)
L4	JUMPER
L5	JUMPER
L6	3 1/2(E) OR 2 1/2(W)
L7	3 1/2(E) OR 2 1/2(W)
L8	3 1/2(E) OR 2 1/2(W)
L9	3 1/2(E) OR 2 1/2(W)
L10	3 1/2(E) OR 2 1/2(W)
L11	3 1/2(E) OR 2 1/2(W)
L12	3 1/2(E) OR 2 1/2(W)
L13	3 1/2(E) OR 2 1/2(W)
L14	3 1/2(E) OR 2 1/2(W)

L15	3 1/2(E) OR 2 1/2(W)
L16	3 1/2(E) OR 2 1/2(W)
L17	3 1/2(E) OR 2 1/2(W)
L18	3 1/2(E) OR 2 1/2(W)
L19	3 1/2(E) OR 2 1/2(W)
L20	3 1/2(E) OR 2 1/2(W)
L21	3 1/2(E) OR 2 1/2(W)
L22	3 1/2(E) OR 2 1/2(W)
L23	3 1/2(E) OR 2 1/2(W)
L24	3 1/2(E) OR 2 1/2(W)
L25	3 1/2(E) OR 2 1/2(W)
L26	3 1/2(E) OR 2 1/2(W)
L27	3 1/2(E) OR 2 1/2(W)
L28	3 1/2(E) OR 2 1/2(W)
L29	3 1/2(E) OR 2 1/2(W)
L30	3 1/2(E) OR 2 1/2(W)
L31	3 1/2(E) OR 2 1/2(W)
L32	3 1/2(E) OR 2 1/2(W)
L33	3 1/2(E) OR 2 1/2(W)
L34	3 1/2(E) OR 2 1/2(W)
L35	3 1/2(E) OR 2 1/2(W)
L36	3 1/2(E) OR 2 1/2(W)
L37	3 1/2(E) OR 2 1/2(W)
L38	3 1/2(E) OR 2 1/2(W)
L39	3 1/2(E) OR 2 1/2(W)
L40	3 1/2(E) OR 2 1/2(W)
L41	3 1/2(E) OR 2 1/2(W)
L42	3 1/2(E) OR 2 1/2(W)
L43	3 1/2(E) OR 2 1/2(W)
L44	3 1/2(E) OR 2 1/2(W)
L45	JUMPER
L46	JUMPER
L47	JUMPER
L48	3 1/2(E) OR 2 1/2(W)
L49	3 1/2(E) OR 2 1/2(W)
L50	3 1/2(E) OR 2 1/2(W)
L51	3 1/2(E) OR 2 1/2(W)
L52	3 1/2(E) OR 2 1/2(W)
L53	3 1/2(E) OR 2 1/2(W)
L54	3 1/2(E) OR 2 1/2(W)
L55	3 1/2(E) OR 2 1/2(W)
L56	3 1/2(E) OR 2 1/2(W)
L57	3 1/2(E) OR 2 1/2(W)
L58	3 1/2(E) OR 2 1/2(W)
L59	3 1/2(E) OR 2 1/2(W)
L60	3 1/2(E) OR 2 1/2(W)

L61	3 1/2(E) OR 2 1/2(W)
L62	3 1/2(E) OR 2 1/2(W)
L63	3 1/2(E) OR 2 1/2(W)
L64	3 1/2(E) OR 2 1/2(W)
L65	3 1/2(E) OR 2 1/2(W)
L66	3 1/2(E) OR 2 1/2(W)
L67	3 1/2(E) OR 2 1/2(W)
L68	3 1/2(E) OR 2 1/2(W)
L69	JUMPER
L70	1mH CHOKE
L71	3 1/2 IN 9MM DIA CHOKE
L72	3 1/2 IN 9MM DIA CHOKE
L73	3 1/2 IN 9MM DIA CHOKE
L74	3 1/2(E) OR 2 1/2(W)
L75	3 1/2(E) OR 2 1/2(W)
L76	3 1/2(E) OR 2 1/2(W)
L77	3 1/2(E) OR 2 1/2(W)
L78	3 1/2(E) OR 2 1/2(W)
L79	3 1/2(E) OR 2 1/2(W)
L80	3 1/2(E) OR 2 1/2(W)
L81	3 1/2(E) OR 2 1/2(W)
L82	3 1/2(E) OR 2 1/2(W)
L83	3 1/2(E) OR 2 1/2(W)
L84	3.3uH CHOKE
L85	3 1/2(E) OR 2 1/2(W)
L86	3 1/2(E) OR 2 1/2(W)
L87	3 1/2(E) OR 2 1/2(W)
L88	3 1/2(E) OR 2 1/2(W)
L89	3 1/2(E) OR 2 1/2(W)
L90	3 1/2(E) OR 2 1/2(W)
L91	3 1/2(E) OR 2 1/2(W)

*Note:

Value preceding (E) indicates no. of turns of enamelled wire.

Value preceding (W) indicates no. of turns of wire wrapped wire.

LIST OF MISCELLANEOUS ITEMS

Q1	9012
Q2	9018
Q3	9018
Q4	9018
Q5	9018

Q6	1402
Q7	1402
Q8	9018
Q9	31X
Q10	31X
Q11	9014
Q12	9014
Q14	9018
Q15	9018

LED1--LED3

Z3	5V1

D1	IN4148
D2	1K60
D3	IN4148
D4	IN4148
D5	IN4148
D6	IN4148
D7	IN4148
D8	IN4148
D9	IN4148
D10	IN4148
D11	IN4148
D12	IN4148

XTAL2	1.8432MHz (OPTIONAL)
XTAL3	14.31818MHz/17.73447MHz
XTAL4	3.68MHz (OPTIONAL)

VC1	20pF
VC2	10pF

Note: R23, R24, Q8, D2 should be inserted if U27 is 74HCT373.

APPENDIX G

PCB AND COMPONENT LAYOUT

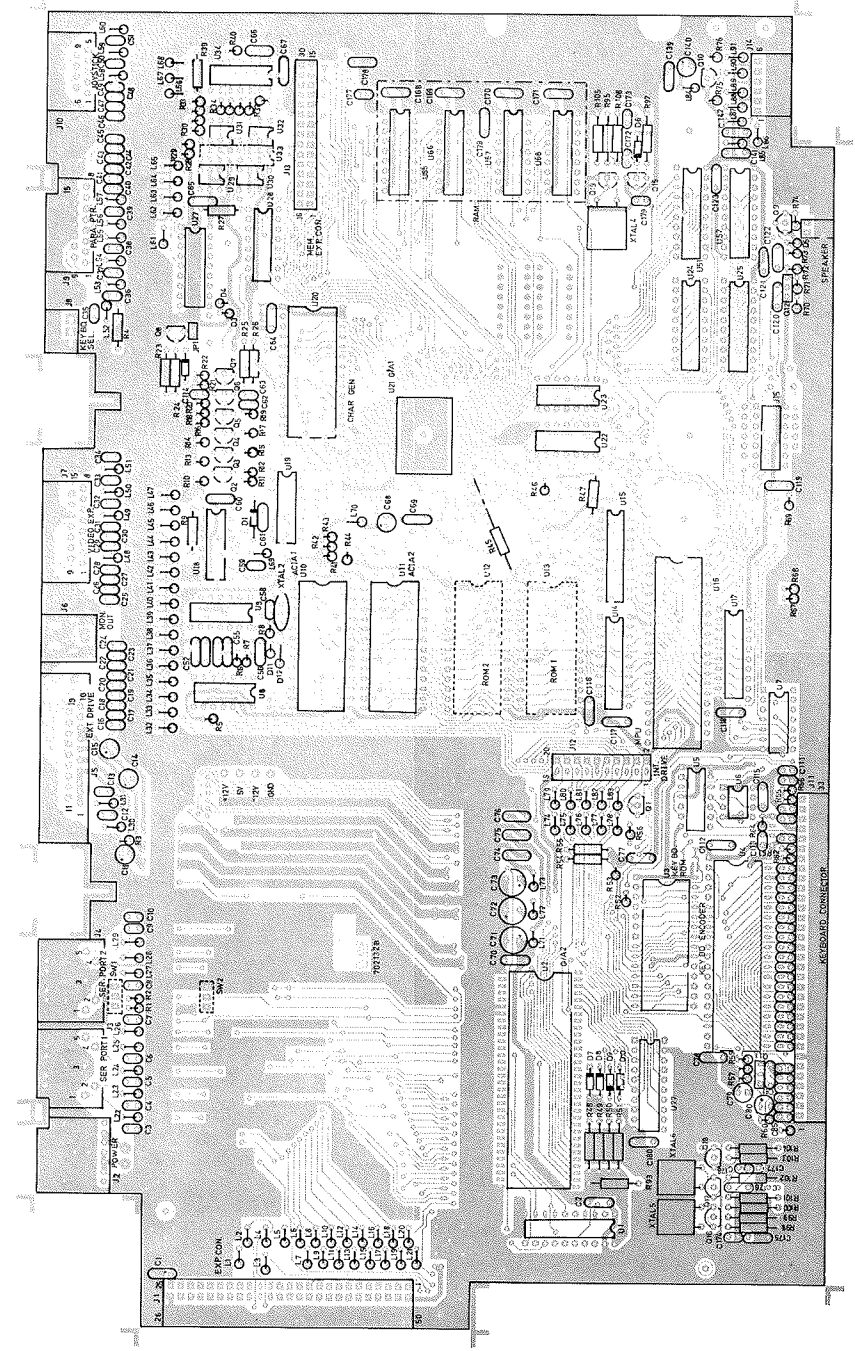


Fig. G1 PCB and component layout of component side of motherboar

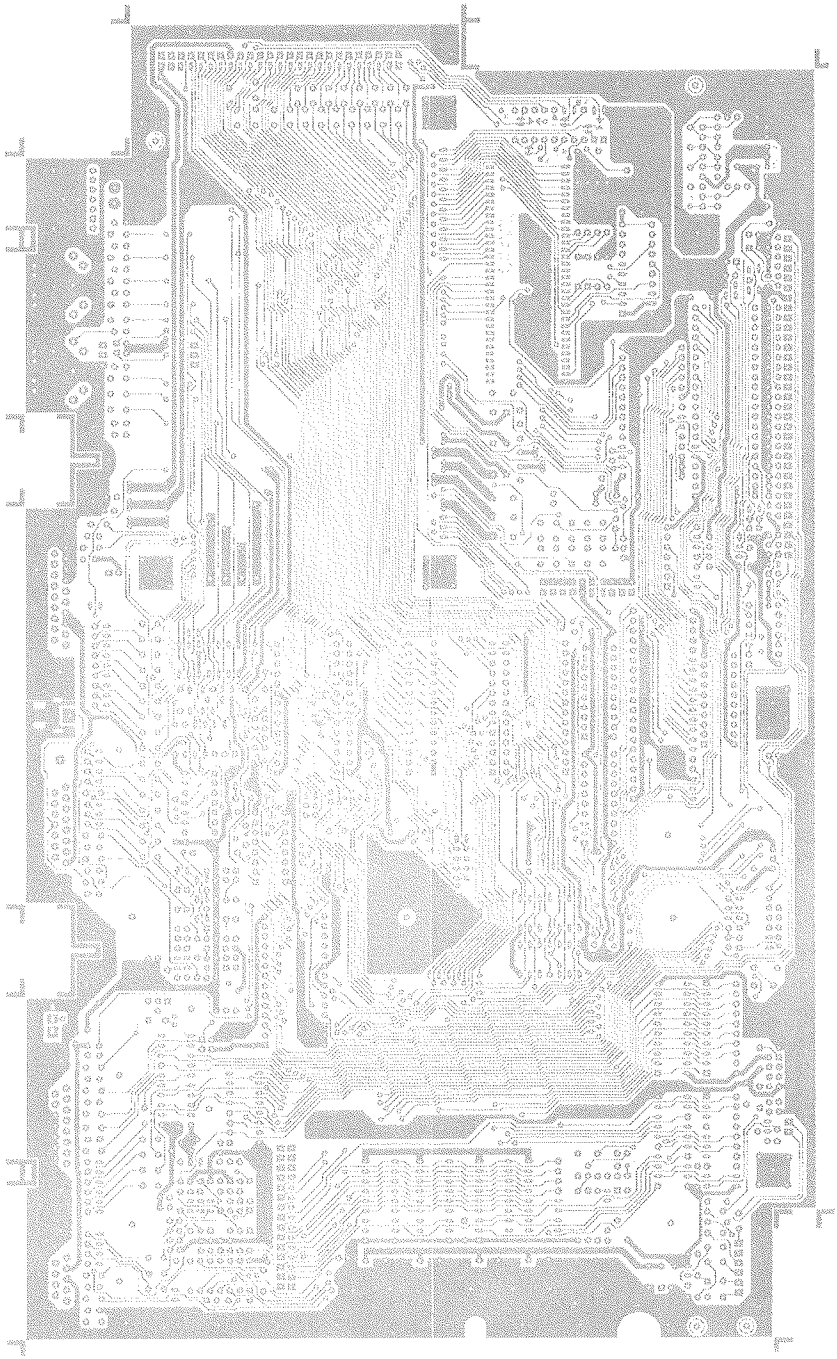


Fig. G2 PCB layout of solder side ofmotherboard

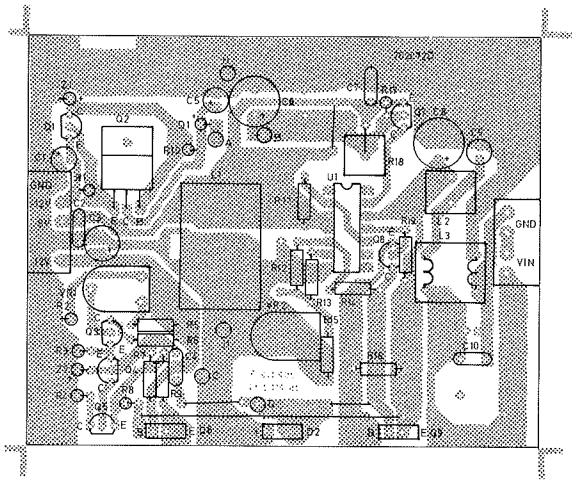


Fig. G3 PCB layout of power supply

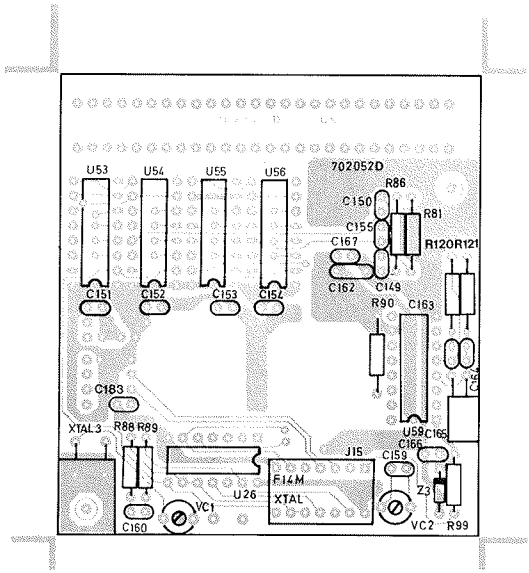


Fig. G4 PCB layout of component side of daughter board

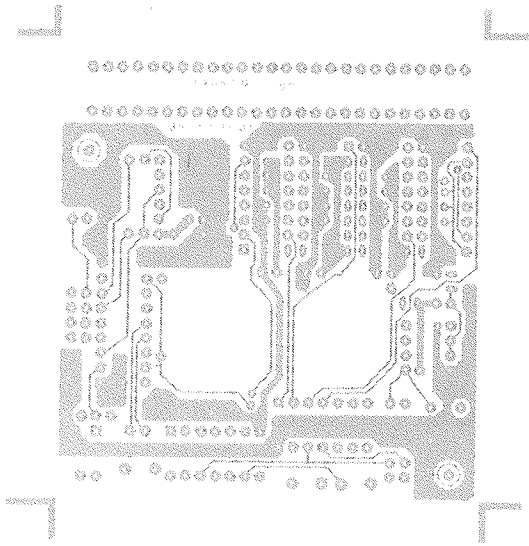


Fig. G5 PCB layout of solder side of daughter board

APPLE and APPLE IIc are registered trademarks of Apple Computer Inc..

CP/M is a registered trademark of Digital Research Inc..