

# SERVICE & TECHNICAL MANUAL (VOL. 1)

SVI-728 COMPUTER SYSTEM

SERVICE & TECHNICAL  
MANUAL (VOL. 1)  
SVI-728 COMPUTER SYSTEM



**SVI**<sup>TM</sup>  
SPECTRAVIDEO

**SVI**<sup>TM</sup>  
SPECTRAVIDEO



## COPYRIGHT

Copyright 1985 by Spectravideo International Limited (SIL). No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means. electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the prior written permission of SIL, Room 507-9, New World Office Building (West Wing), 20, Salisbury Road, Tsimshatsui, Kowloon, Hong Kong.

## DISCLAIMER

Spectravideo International Limited reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of SIL to notify any person of such revision or changes.

## TRADEMARKS

SVI is a registered trademark of Spectravideo. CP/M is a registered trademark of Digital Research. Z-80 is a trademark of Zilog, Inc.

First Printing: January, 1985

## ACKNOWLEDGEMENT

The author would like to thank the following people for their co-operation and helpful suggestion in preparing this manual.

Technical Information : Eddie Suen and Team 2 R & D Engineers

Graphic Design : Johnny Ko and Team 8 Artists

Servicing Information : All the Manufacturing Engineers

Last of all, all members of staff in Customer Service Department would highly deserve my best regard for their tremendous effort contributed in preparing this manual.

Kimba LAU  
January, 1985

## INTRODUCTION

### Scope

This manual is designed to assist the experienced and inexperienced service technician in locating and repairing problems that may occur in SVI-728 computer and its peripherals. This manual will cover all necessary information and clues for technical reference and servicing use. A spare part listing could be found in appendix for spare part order via our local agent.

## TABLE OF CONTENTS

<u>Contents</u>	<u>Page</u>
ACKNOWLEDGEMENT	ii
INTRODUCTION	iii
TABLE OF CONTENTS	iv
CHAPTER 1 SYSTEM CONFIGURATION AND ARCHITECTURE	
1.1 What is MSX?	1-1
1.2 SVI-728 MSX Computer System	1-2
1.2.1 Basic Specification	1-2
1.3 system Architecture	1-3
1.4 Z80A Microprocessor	1-4
1.5 Video Display Processor (TMS-9129 for PAL or TMS-9118 for NTSC)	1-5
1.5.1 Screen Display	1-14
1.6 Programmable Sound Generator AY-3-8910	1-15
1.7 The Programmable Peripheral Interface 8255A	1-20
1.8 Floppy Disk Controller Description	1-25
CHAPTER 2 INPUT/OUTPUT SPECIFICATION	
2.1 List of Connectors	2-1
2.2 Cassette Interface	2-2
2.3 Input/Output Ports	2-3
2.4 Printer Interface	2-4
2.5 Cartridge Slot	2-5
2.5.1 Specification of Cartridge	2-5
2.5.2 Cartridge	2-6
2.5.3 Conditional of Cartridge Connection	2-8
2.5.4 Power Capacity	2-8
2.6 Sound	2-9
2.7 SVI-728 Expansion Module Interface and Game Slot Signal Description	2-10
2.8 Keyboard	2-11

## CHAPTER 3 MEMORY SYSTEM

3.1	Memory Map	3-1
3.2	Input/Output Map	3-3
3.3	PPI Bit Assignment	3-5
3.4	PSG Bit Assignment	3-6
3.5	Input/Output Mapping of MSX Drive	3-7

## CHAPTER 4 SVI-728 PERIPHERALS

4.1	SVI-707 MSX Disk Drive (320K)	4-1
4.2	SVI-727 MSX 80 Column Video Cartridge	4-3
4.3	SVI-737 MSX 300 Baud Modem with RS-232 Interface	4-4
4.4	SVI-747 MSX 64K RAM Memory	4-6
4.5	SVI-757 MSX RS-232 Interface Cartridge	4-7
4.6	SVI-767 MSX Data Cassette	4-7
4.7	SVI-709 Network Interface Card	4-8
4.8	SVI-101 & SVI-102 MSX Joystick	4-8

## CHAPTER 5 DISASSEMBLY/ASSEMBLY AND MAINTENANCE

5.1	General Caution	5-2
5.2	Keyboard and PCB Access	5-3
5.3	Reassemble of the Console	5-4
5.4	Removal of the Heat Sink Plate	5-5
5.5	Reinstalling of the Heat Sink Plate	5-6
5.6	Removal of Video and Power Board	5-6
5.7	Reinstallation of the Video and Power Board	5-7
5.8	Removal of the Logic Board	5-7
5.9	Reinstallation of the Logic Board	5-7
5.10	Removal of Keyboard Assembly from Upper Housing	5-8
5.11	Reinstallation of Keyboard Assembly to Upper Housing	5-8
5.12	Removal of Keytops	5-9
5.13	Reinstallation of Keytops	5-9
5.14	Disassemble of Keyboard Assembly	5-10
5.15	Reassemble of Keyboard Assembly	5-11

## CHAPTER 6 TROUBLE SHOOTING

6.1	Swap Out Procedure	6-1
6.2	Symptom Checklist for SVI-728 Computer	6-1
6.3	Diagnostic Flowchart for SVI-728 Computer	6-3
6.3.1	Basic Operation	6-4
6.3.2	Power Failure	6-5
6.3.3	Cartridge Boot Up Test	6-6
6.3.4	System ROM Test	6-7
6.3.5	Video RAM Test	6-8
6.3.6	System RAM Test	6-9
6.3.7	Keyboard Test	6-10
6.3.8	I/O (Joystick) Ports Test	6-11
6.3.9	Printer Port Test	6-12
6.3.10	Cassette Port test	6-13
6.4	Diagnostic Flowcharts for SVI-707 Disk Drive	6-19

## APPENDIX A SPARE PART LISTS

### APPENDIX B CIRCUIT DIAGRAM

- SVI-728 Computer (EPROM Version)
- SVI-728 Computer (ROM Version)
- Peripheral Drawings

### APPENDIX C SOURCE CODE LISTING OF SVI-728

- CP/M 2.24 BIOS
- MSX DOS Disk Input/Output Driver

### APPENDIX D COMPONENT SPECIFICATION

### APPENDIX E SOFTWARE TESTING PROGRAMMES

(Test Cartridge for SVI-728 Computer)

## CHAPTER 1

### SYSTEM CONFIGURATIONS AND ARCHITECTURE

#### 1.1 WHAT IS MSX?

In August, 1982 SEPCTRAVIDEO INTERNATIONAL LTD. contacted Microsoft seeking to buy a ROM BASIC for SVI-318 and SVI-328 personal computer. When Nishi (representative from Microsoft) saw the specification, he was very impressed. Microsoft has long been thinking of standardizing the low end 8-bit personal computer market, so that softwares written for the standard could run on any computers which are built according to the standard. After some modifications on SVI-318/SVI-328 circuits, the standard was finally set. On 17 June, 1983 the MSX standard was finally announced in Tokyo. It is essentially a software standard requiring a software standard requiring a single board computer design consisting of the following:

- a Zilog Z80A 8-bit microprocessor
- a Texas Instruments TMS-9918A video chip or equivalent
- a General Instruments AY-3-8910 audio chip
- an updated Microsoft Basic resident in 32K bytes of ROM
- minimum 8K bytes of RAM
- 40-column display
- 16-colour capacity
- support for cassette storage
- expansion slot for software cartridge or disk add-on
- different keyboards for Japan, Korea, Europe, and the United States
- minimum 1 joystick port

By specifying this minimum hardware configuration, a software "code" standard is created, because all MSX machines are internally identical, software created for one machine will run on all machines.

The disk operating system for MSX machines is called MSX-DOS which is CP/M-80 compatible. It imitates CP/M to the extent that most CP/M programmes will run under MSX-DOS providing:

1. Software does not make use of graphics (most CP/M software does not make use of graphics anyway);
2. Software is made available on MSX 5.25, 3.25, or 3.5 inch floppies;



3. Screen and keyboard modifications are made to the programme to accommodate the MSX format.

MSX software is compatible with all MSX machines only if it comes in a ROM cartridge, this is due to the fact that some large programmes cannot be fit into those MSX machines which have only 8 or 16K user RAM. However our SVI-728 has 64K user RAM, therefore it is capable of running all MSX softwares as well as most CP/M programmes.

## 1.2 SVI-728 MSX COMPUTER SYSTEM

With the engineering experience gained in SVI-318/SVI-328 (MSX prototype machine), SVI has come up with a splendid product of innovation, SVI-728 which serves as your entrance into the expanding world of MSX software.

SVI-728 is a computer which exploded full advantage of MSX software compatibility for both becoming available today and those yet to be discovered. All MSX hardware standard mentioned in 1.1 were enveloped in SVI-728 architecture.

### 1.2.1 BASIC SPECIFICATION

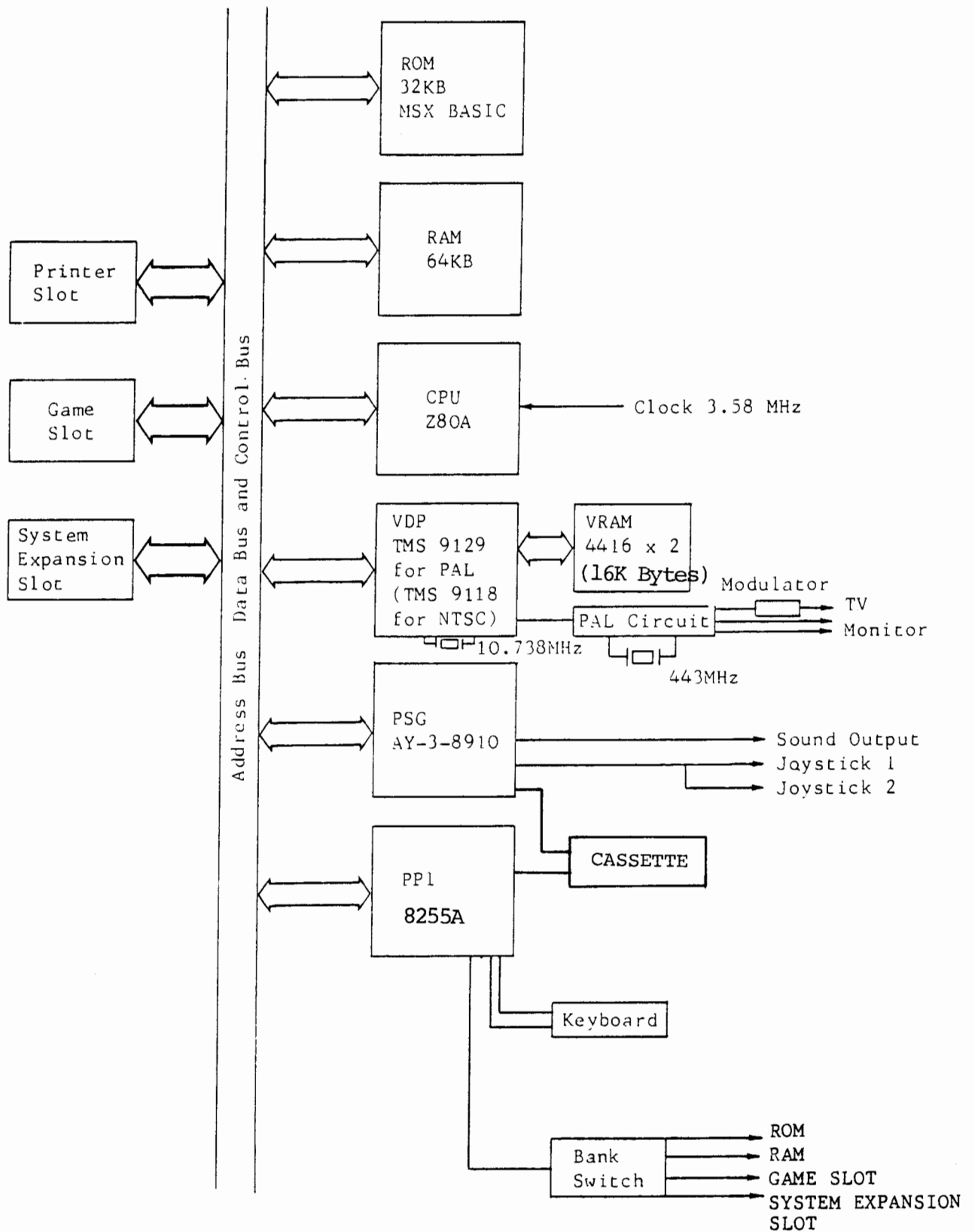
#### A. LSI

- CPU           Z80A  
                  Clock 3.58MHz (Colour Sub-carrier Frequency)  
                  1 Wait Cycle in M1
- VDP           TMS-9129 (or TMS-9118) +  
                  2 x 4416 (16K)
- PSG           AY-3-8910
- PPI           8255A

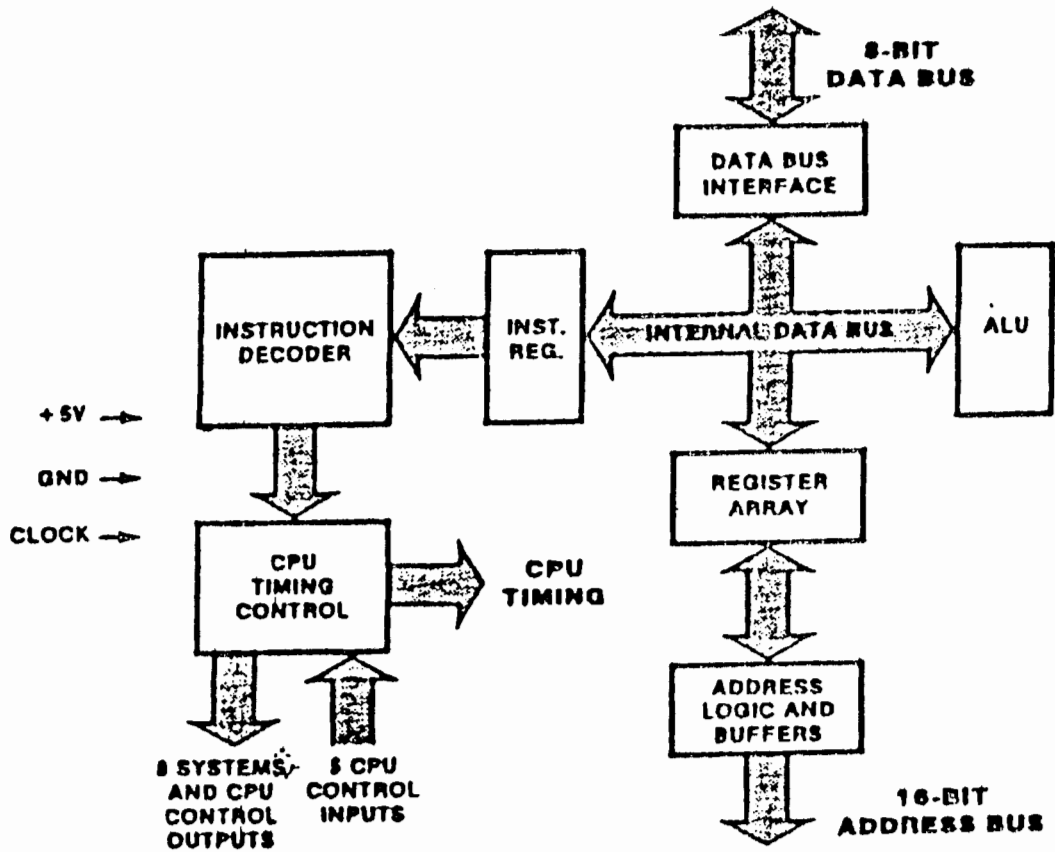
#### B. MEMORY

- ROM           This 32K ROM contains the MSX BASIC interpreter programme as well as the monitor programme.
- RAM           64K Bytes
- Basic ROM starts from 0000H to 7FFFH; and RAM starts from FFFFH and grows downward on the memory map. (Refer to Chapter 4 - memory map for the details)

### 1.3 SYSTEM ARCHITECTURE



1.4 Z80A MICROPROCESSOR



Z80 CPU Block Diagram

Detail information about Z80 CPU, please refer to books or catalogues published by I.C. manufacturer.

1.5 VIDEO DISPLAY PROCESSOR  
 (TMS-9129 FOR PAL OR TMS-9118 FOR NTSC)

The TMS-9118/TMS-9129 VDPs are N-channel MOS LSI devices used in video systems where data display on a master-scanned home colour television set or colour monitor is desired. These devices generate all necessary video, control, and synchronization signals and also control the storage, retrieval, and refresh of display data in the dynamic screen refresh memory.

The TMS-9118 have a 525 line format for U.S. television while the TMS-9129 has a 625 line format for use with the European PAL system. The TMS-9118/TMS-9129 VDPs operate only in a non-interlaced mode. Therefore, TMS-9129 is used for PAL system while TMS-9118 for NTSC system.

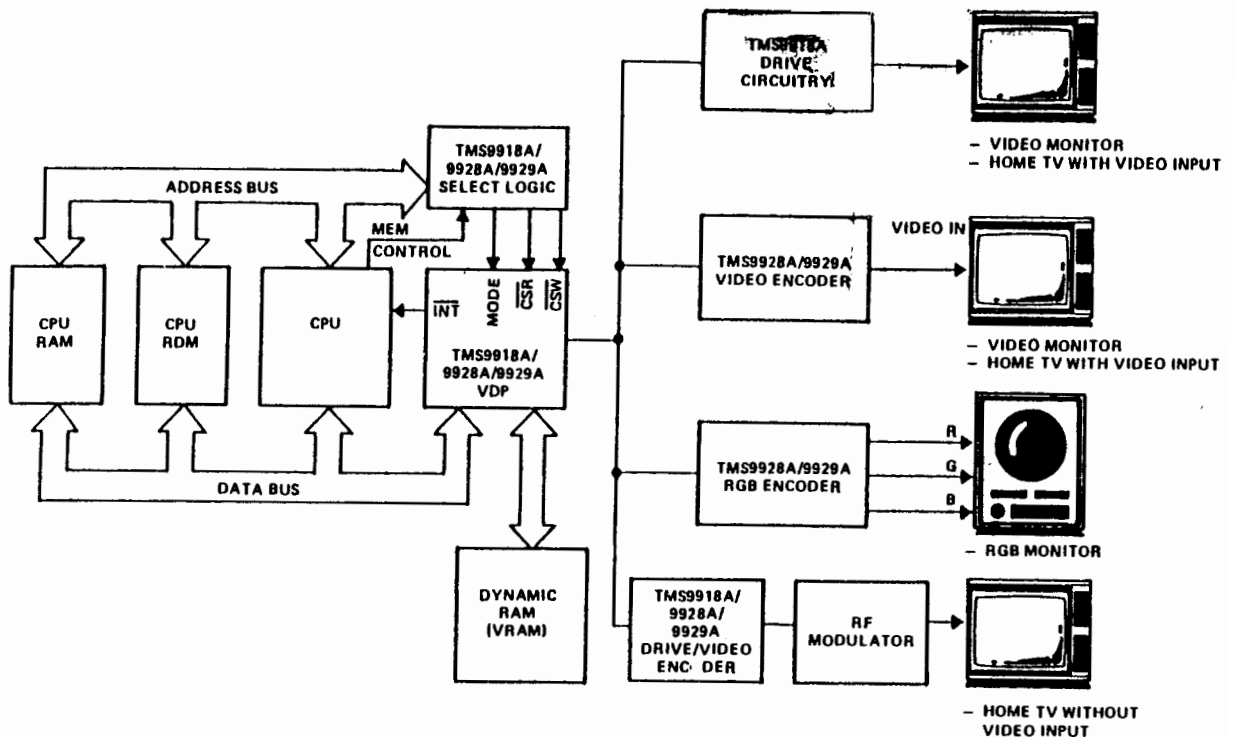


Figure 1.1 - VDP System Block Diagram

## VDP INTERFACES

The VDP has 3 basic interfaces: CPU, colour monitor, and VRAM, the contents of which define the TV image. The VDP also has 8 write-only registers and a read-only status register.

### VDP-CPU INTERFACE

The VDP communicates with the CPU via an 8-bit bidirectional data bus (CD0 to CD7). 3 control lines, decoded from the CPU address and enable lines, determine interpretation of the bus. Through the bus, the CPU can write to VRAM, read from VRAM, write to VDP registers, and read the VDP status. The VDP also generates an interrupt signal after every refresh of the TV display if the interrupt is enabled.

### VDP-VRAM INTERFACE

The VDP use TMS-4416 (16K x 4) dynamic RAMs. Since the early write cycle is used by the VDP, G on the TMS-4416 must be tied to ground.

The VDP accesses up to 16,384 bytes of VRAM using a 14-bit VRAM address. The VDP fetches data from the VRAM in order to process the video image. The VDP also stores data in or reads out data from the VRAM during a CPU-VRAM data transfer. The VDP automatically refreshes the VRAM.

The VDP-VRAM interface consists of a bidirectional 8-bit data bus and 3 control lines as shown in Figure 1.2.

The VDP reads from and writes data to the VRAM on the VRAM data bus (RD0 - RD7). The VDP outputs the address to the VRAM over the VRAM address bus (AD0 - AD7). The VRAM row address is output when RAS is active (low). The column address is output when CAS is active (low). Data is output to the VRAM when Read/Write is active (low). Figure 1.3 illustrates pin connections used by the TMS-4416-15/20 and TMS-4116-15 VRAMs. Figure 5 shows the address partitioning for both VRAM devices.

### VDP-MONITOR INTERFACE

The interface to the monitor can consist either of wiring the TMS-9118 composite video output pin (suitably buffered) to the input of a colour or black-and-white monitor, or an appropriate RF modulator can be used to feed the signal into the TV antenna terminals. The TMS-9128/TMS-9129 require additional encoder circuitry to interface to a R-G-B or to a composite video monitor.

## VDP-POWER SUPPLY

A 10uF to 50uF decoupling capacitor and choke coil are suggested between Vcc and Vss to guarantee proper VDP operation.

## VDP INTERRUPT

The VDP INT output pin is used to generate an interrupt at the end of each active-display scan, which is about every 1/60 second for the TMS-9118 and 1/50 second for the TMS-9129. The INT output is active when the Interrupt Enable bit (IE) in VDP Register 1 is a 1 and the F bit of the status register is a 1. Interrupts are cleared when the status register is read.

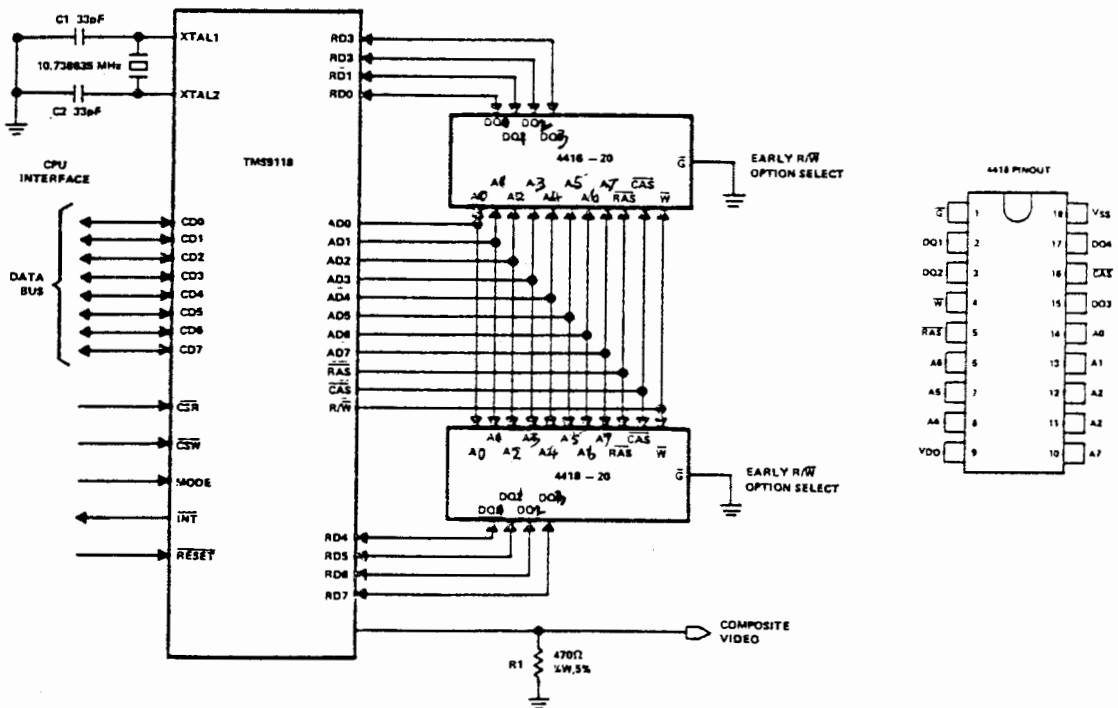
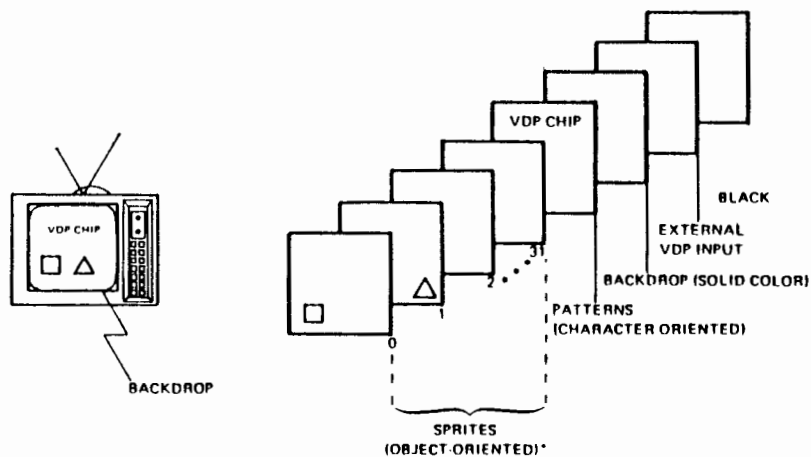


Figure 1.2 - VRAM Interface (TMS-4416-20)

## VIDEO DISPLAY MODES

The VDP can operate in any 1 of 4 modes, each of which can affect the way the VRAM is mapped onto the television screen. In Graphics I and II modes, characters are mapped onto the screen in 8 x 8 picture elements (pixels) blocks yielding 24 lines of 32 blocks (pattern positions) each. In Text mode, there are 24 lines of 40 blocks, each of which is 6 x 8 pixels. In Multi-colour mode, there are 48 lines of 64 blocks, each of which is composed of 4 x 4 pixels, all of 1 solid colour. In addition to these, objects termed sprites can be superimposed onto the television image for all modes except text. Furthermore, signals entering the TMS-9118 through the external VDP input can be used as a background to the TMS-9118.

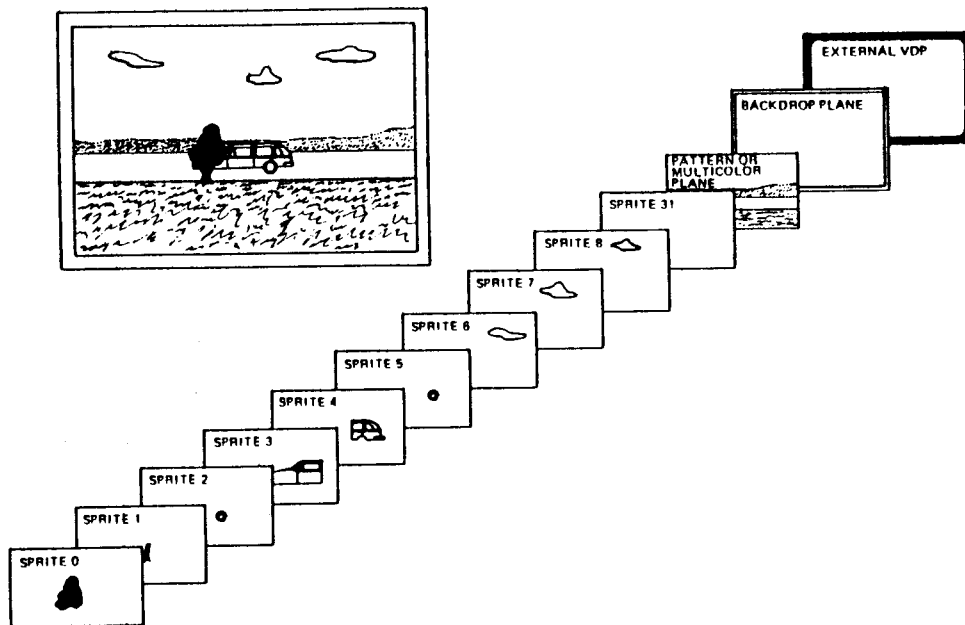
The VDP displays an image on the screen that can best be envisioned as a set of display planes sandwiched together. Figure 6 shows the definition of each of the planes. Objects on planes closest to the viewer have higher priority. In cases where 2 entities on 2 different planes are occupying the same spot on the screen, the entity on the higher priority plane will show at that point. For an entity on a specific plane to show through, all planes in front of that plane must be transparent at that point. The first 32 planes (Figure 7) each may contain a single sprite. The areas of the Sprite Planes, outside of the sprite itself, are transparent. Since the co-ordinates of the sprite are in terms of pixels, the sprite can be positioned and moved about very accurately. Sprites are available in 3 sizes: 8 x 8 pixels, 16 x 16 pixels, and 32 x 32 pixels.



VDP Display Planes

Behind the Sprite Planes is the Pattern Plane. The Pattern Plane is used for textual and graphics images generated by the Text, Graphics I, Graphics II, or Multi-colour modes. Behind the Pattern Plane is the backdrop, which is larger in area than the other planes so that it forms a border around the other planes. The last and lowest priority plane is the External VDP Plane. Its image is defined by the external VDP input pin which allows the TMS-9118 to mix the video signal from another VDP internal to the chip. A black default plane appears if all planes are transparent and the external VDP plane is inactive.

This mixing must occur outside of the chip for the TMS-9128 and TMS-9129. This is achieved through the colour difference outputs swinging to a special level (synchronization level is shown in Figure 8) not used by the colour difference signals in normal operation. This occurs when bit 7 of Register 0 is set high. External mixing circuitry is required to detect this change in the level of the colour difference signals and then switch from the VDP signals to the signals of another VDP source. (See Figures 9 and 10)

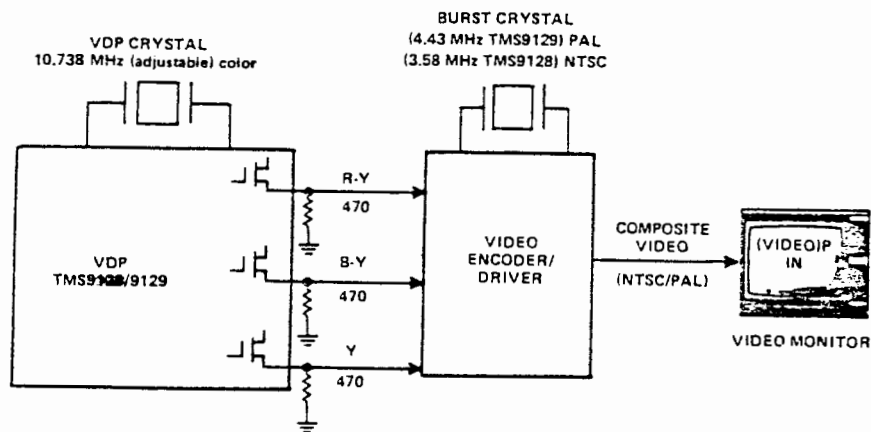


VDP Display Planes



## TMS-9129 MONITOR INTERFACE

The Y, R-Y and B-Y output signals require external encoder circuitry to drive a video colour monitor. The Y output signal contains all necessary horizontal and vertical synchronization signals as well as luminance. The R-Y and B-Y signals contain the unmodulated chrominance information and are used in the NTSC and PAL systems to modulate 2 carriers in quadrature. The internal output buffer devices on these pins are source-follower MOS transistors that require an external pull-down resistor to Vss as shown in Figure 11. a 470 ohm resistor is recommended.



Note: The LM1889 is typically used in the encoder circuitry.

TMS-9129 with Different Monitors

## OSCILLATOR AND CLOCK GENERATION

The VDP is designed to operate with a 10.738625 ( $\pm 0.005$ ) MHz crystal input to generate the required internal clock signals. A fundamental frequency, parallel-mode crystal is used as the frequency reference for the internal clock oscillator, which is the master time base for all system operations. This master clock is divided by two to generate the pixel clock (5.3 MHz) and by three to provide the CPUCLK (3.58 MHz).

## CPU WRITE TO VDP REGISTER

The VDP has 8 write-only registers and 1 read-only status register. The write-only registers control the VDP operation and determine the way in which VRAM is allocated. The status register contains interrupt, sprite coincidence and fifth sprite status flags.

Each of the 8 VDP write-only registers can be loaded using 2 8-bit data transfers from the CPU. Table 1 describes the required format for the 2 bytes. The first byte transferred is the data byte, and the second byte transferred controls the destination. The most-significant bit of the second byte must be a '1'. The next 4 bits are '0's, and the lowest 3 bits make up the destination register number. The MODE input is high for both byte transfers.

To write the data for an internal register after a byte of data has been loaded, the status register must be read so that internal logic will accept the next byte as data and not as a register destination. This situation may be encountered in interrupt-driven programme environments. Whenever the status of VDP write parameters is in question, this procedure should be used. Note that the CPU address is destroyed by writing to the VDP register.

## CPU WRITE TO VRAM

The CPU transfers data to the VRAM through the VDP using a 14-bit autoincrementing address register. 2-byte transfers are required to set up the address register. A 1-byte-transfer is then required to write the data to the addressed VRAM byte. The address register is then autoincremented. Sequential VRAM write require only 1-byte-transfer since the address register is already set up. During set up of the address register, the 2 most-significant bits of the second address byte must be '0' and '1' respectively. MODE is high for both address transfers and low for the data transfer. CSW is used in all transfers to strobe the 8 bits into the VDP. See Table 1.

#### CPU READ FROM VDP STATUS REGISTER

The CPU can read the contents of the status register with a single-byte transfer. MODE is high for the transfer. CSR is used to signal the VDP that a read operation is required.

#### CPU READ FROM VRAM

The CPU reads data from the VRAM through the VDP using the autoincrementing address register. A 1-byte transfer is then required to read the data from the addressed VRAM byte. The address register is then autoincremented. Sequential VRAM data reads require only a 1-byte transfer since the address register is already set up. During set up of the address register, the 2 most-significant bits of the second address byte must be '0's. By setting up the address this way, a read cycle to VRAM is initiated and read data will be available for the first data transfer to the CPU (see Table 1). MODE is high for the address byte transfers and low for the data transfers. The VDP requires approximately 8 microseconds to fetch the VRAM byte following a data transfer and 3 microseconds following address set up.

OPERATION	0	1	2	3	4	5	6	7	$\overline{\text{CSW}}$	$\overline{\text{CSR}}$	Mode
WRITE TO VDP REGISTER											
Byte 1: Data Write	D7	D6	D5	D4	D3	D2	D1	D0	0	1	1
Byte 2: Register Select	1	0	0	0	0	RS1	RS1	RS0	0	1	1
WRITE TO VRAM											
Byte 1: Address Set Up	A7	A6	A5	A4	A3	A2	A1	A0	0	1	1
Byte 2: Address Set Up	0	1	A13	A12	A11	A10	A9	A8	0	1	1
Byte 3: Data Write	D7	D6	D5	D4	D3	D2	D1	D0	0	1	0
READ ROM VDP REGISTER											
Byte 1: Data Read	D7	D6	D5	D4	D3	D2	D1	D0	1	0	1
READ FROM VRAM											
Byte 1: Address Set Up	A7	A6	A5	A4	A3	A2	A1	A0	0	1	0
Byte 2: Address Set Up	0	0	A13	A12	A11	A10	A9	A8	0	1	1
Byte 3: Data Read	D7	D6	D5	D4	D3	D2	D1	D0	1	0	0

CPU/VDP Data Transfer

1.5.1 SCREEN DISPLAY

- LIST OF DISPLAY MODES

MODE		RESOLUTION	SIZE	NO.*	COLOUR	MOVING OBJECT	SCREEN DISPLAY
Graphic	LSI spec.	25x192	8x8	256	16	Yes	32x24
Multi-colour	LSI spec.	64x48 blk	4x4/blk	-	16	Yes	32x24
Text **	LSI spec.	256x192	8x6	256	2 from 16	No	40x24

\* Pattern number

\*\* Text mode is not supported by BASIC

## 1.6 PROGRAMMABLE SOUND GENERATOR AY-3-8910

The AY-3-8910 is a register oriented Programmable Sound Generator (PSG). Communication between the processor and the PSG is based on the concept of memory-mapped I/O. Control commands are issued to the PSG by writing to 16 memory-mapped registers. Each of the 16 registers within the PSG is also readable so that the microprocessor can determine, as necessary, present states or stored data values.

All functions of the PSG are controlled through its 16 memory-mapped registers. Each of the 16 registers within the PSG is also readable so that the microprocessor can determine, as necessary, present states or stored data values.

All functions of the PSG are controlled through its 16 registers which once programmed, generate and sustain the sounds, thus freeing the system processor for other tasks.

### SOUND GENERATING BLOCK

The Basic Blocks in the PSG which produce the programmed sounds include:

#### Tone Generators

Produce the basic square wave tone frequencies for each channel (A, B, C).

#### Noise Generator

Produce a frequency modulated pseudo random pulse width square wave output.

#### Mixers

Combine the outputs of the Tone Generators and the Noise Generator. 1 for each channel (A, B, C).

#### Amplitude Control

Provides the D/A connectors with either a fixed or variable amplitude pattern. The fixed amplitude is under direct CPU control; the variable amplitude is accomplished by using the output of the Envelope Generator.

## Envelope Generator

Produces an envelope pattern which can be used to amplitude modulate the output of each mixer.

## D/A Converters

The 3 D/A Converters each produce up to a 16 level output signal as determined by the Amplitude Control.

Since virtually all uses of microprocessor-based sound would require interfacing between the outside world and the processor, 2 I/O ports (A and B) has been included in the PSG.

To output data from the CPU bus to a peripheral device connected to I/O Port A would require only the following steps:

1. Latch address R7 (select Enable register)
2. Write data to PSG (setting B6 of R7 to "1")
3. Latch address R16 (select IOA register)
4. Write data to PSG (data to be output on I/O Port A)

To input data from I/O Port A to the CPU bus would require the following:

1. Latch address R7 (select Enable register)
2. Write data to PSG (select B6 to R7 to "0")
3. Latch address R16 (select IOA register)
4. Read data from PSG (data from I/O Port A)

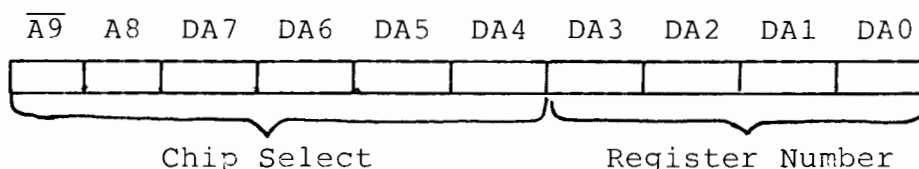
To do input/output on I/O Port B, use Register 17, instead of Register 16. Note that once loaded with data in the input mode, the data will remain on the I/O Ports until changed either by loading different data, by applying a reset (grounding the Reset pin), or by switching to the input mode.

AY-3-8910 PROGRAMMABLE SOUND GENERATOR PIN ASSIGNMENTS

DA7-DA0 (input/output/high impedance): pins 30-37

Data Address 7-0

These 8 lines comprise the 8-bit bidirectional bus used by the microprocessor to send both data and addresses to the PSG and to receive data from the PSG. In the data mode, DA7-DA0 correspond to Register Array bits B7-B0. In the address mode, DA3-DA0 select the register #(0-17) and DA7-DA4 in conjunction with address input A9 and A8 form the high order address (chip select).



A8 (input) : pin 25

$\overline{A9}$  (input) : pin 24

Address 9, Address B

These "extra" address bits are made available to enable the positioning of the PSG (assignment a 16 word memory space) in a total 1024 word memory area rather than in a 256 word memory area as defined by address bits DA7-DA0 alone. If the memory size does not require the use of these extra address lines they may be left unconnected as each is provided with either an on-chip pull down (A9) or pull-up (A8) resistor. In SVI-728, A9 and A8 are tied to an external ground and +5V respectively.

$\overline{\text{RESET}}$  (input) : pin 23

For initialization/power-on purpose, applying a logic "0" (ground) to the Reset pin will reset all registers to "0". The Reset pin is provided with an on-chip pull-up resistor.

CLOCK (input) : pin 22

This TTL-compatible input supplies the timing reference for the Tone, Noise and Envelope Generators.

BDIR, BC2, BC1 (inputs) : pins 27, 28, 29



Bus DIRection, Bus Control 2, 1

These bus control signals are generated directly by Z80A series of microprocessors to control all external and internal bus operations in the PSG. When using a processor other than the Z80A, these signals can be provided either by comparable bus signals or by simulating the signals on I/O lines of the processor.

This could simplify the programming of the bus control signals to the following, which would only require that the processor generate 2 bus control signals (BDIR and BC1, with BC2 tied to +5V):

BDIR	BC2	BC1	PSG FUNCTION				PSG
----	---	---	-----				
0	1	0	INACTIVE.	---	FROM	-----	BDIR BC2 BC1
0	1	1	READ FRIN PSG.	---	PROCESSOR	+5---	
1	1	0	WRITE TO PSG.			-----	
1	1	1	LATCH ADDRESS				

ANALOG CHANNEL A, B, C (output)

Each of these signals is the output of its corresponding mix of T4, and provides an up to 1V peak-peak signal representing the complex sound waveshape generated by the PSG.

IOA7 - IOA0 (input/output) : pins 14 - 21  
IOB7 - IOB0 (input/output) : pins 6 - 13

Input/Output A7-A0, B7-B0

Each of these 2 parallel input/output ports provides 8 bits of parallel data to/from the PSG/CPU bus from/to any external devices connected to the IOA or IOB pins. Each pin is provided with an on-chip pull-up resistor, so that when in the "input" mode, all pins will read normally high. Therefore, the recommended method for scanning external switches, for example, would be to ground the input bit.

TEST 1 : pin 39  
TEST 2 : pin 26

These pins are for user test purposes only and should be left open - do not use as tie-points.

Vcc : pin 40

Nominal +5 Volt power supply to the PSG.

Vss : pin 1

Ground reference for the PSG.

### The Programmable Sound Generator

The PSG AY-3-8910 generates all required sound under software control. There are 2 8-bit I/O ports which are programmed as follow:

Port A: Programmed as input port and is used for the input signal from 2 joysticks controllers.

Port B: Programmed as output port.

Detail signals for the ports are listed in the table below:

AY-3-8910 I/O Port A (Input Port)	
D0	----- Forward*
D1	----- Backward*
D2	----- Left*
D3	----- Right*
D4	----- TRGA1*
D5	----- TRGA2*
D6	----- NC
D7	----- Cassette Tape Read

\* Used by Joystick 1 when bit 6 of Port B is low  
Used by Joystick 2 when bit 6 of Port B is high

AY-3-8910 I/O Port B (Output Port)	
D0	-- )
D1	-- ) Make the "H" level, when used as an output port
D2	-- ) Tied an open collector buffer to the output
D3	-- )
D4	-- O/P
D5	-- O/P
D6	-- Input Select
D7	-- NC

## 1.7 THE PROGRAMMABLE PERIPHERAL INTERFACE 8255A

The 8255A PPI is used to strobe the keyboard line, to select slots and to control the cassette tape system. The ports are programmed as follow:

Port A : Programmed as output port. Bits 0-7 are the address slots select signals that choose the slots to be used.

Port B : Programmed as output port and is used for keyboard read data.

Port C : Programmed as output port. Bits 0-3 outputs BCD data for keyboard scanning. Bits 4-6 outputs control signals for the cassette. Bit 7 is used to mix PSG sound data.

The functional configuration of the 8255A is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

### Data Bus Buffer

This 3-state bidirectional 8-bit buffer is used to interface the 8255A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

### Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control buses in turn, issues commands to both of the Control Groups.

### $\overline{CS}$

Chip Select. A "low" on this input pin enables the communication between the 8255A and the CPU.

( $\overline{RD}$ )

Read. A "low" on this input pin enables the 8255A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255A.

( $\overline{WR}$ )

Write. A "low" on this input pin enables the CPU to write data or control words into the 8255A.

(A0 and A1)

Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of 1 of the 3 ports or the control word registers. They are normally connected to the least significant bits of the address bus (A0 and A1).

#### 8255A Basic Operation

A	A	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	PORT A ===== DATA BUS
0	1	0	1	0	PORT B ===== DATA BUS
1	0	0	1	0	PORT C ===== DATA BUS
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	DATA BUS ===== PORT A
0	0	0	1	0	DATA BUS ===== PORT B
0	1	0	1	0	DATA BUS ===== PORT C
1	0	0	1	0	DATA BUS ===== CONTROL
					OUTPUT OPERATION (WRITE)
X	X	X	X	1	DATA BUS ===== 3-STATE
1	1	0	1	0	ILLEGAL CONDITION
X	X	1	1	0	DATA BUS ===== 3-STATE

(RESET)

Reset. A "high" on this input clears the control register and all ports (A, B, C) are set to the input mode.

## Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 8255A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 8255A.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A - Port A and Port C upper (C7 - C4)  
Control Group B - Port B and Port C lower (C3 - C0)

The Control Word Register can only be written into. No Read operation of the Control Word Register is allowed.

## Port A, B and C

The 8255A contains three 8-bit ports (A, B and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255A.

Port A. 1 8-bit data output latch/buffer and 1 8-bit data input latch.

Port B. 1 8-bit data input/output latch/buffer and 1 8-bit data input buffer.

Port C. 1 8-bit data output latch/buffer and 1 8-bit data input buffer (no latch for input). This port can be divided into 2 4-bit port under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with ports A and B.

## Mode Selection

There are 3 basic modes of operation that can be selected by the system software:

Mode 0 - Basic Input/Output  
Mode 1 - Strobed Input/Output  
Mode 2 - Bi-directional Bus

When the reset input goes "high", all ports will be set to the input mode (i.e., all 24 lines will be in the high impedance state). After the reset is removed, the 8255A can remain in the input mode with no additional initialization required. During the execution of the system programme, any of the other modes may be selected using a single output instruction. This allows a single 8255A to serve a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into 2 portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance, Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.

The mode definitions and possible mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 8255A has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

#### Single Bit Set/Reset Feature

Any of the 8 bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

## Interrupt Control Functions

When the 8255A is programmed to operate in mode 1 or mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the bit set/reset function of port C.

This function allows the programmer to disallow or allow a specific I/O device to interrupt the CPU without affecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) - INTE is SET - Interrupt enable  
(BIT-RESET) - INTE is RESET - Interrupt enable

Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

### 8255 Port A (Input Port)

D0 -----) 0000 - 3FFF address slot select signal  
D1 -----)  
D2 -----) 4000 - 7FFF address slot select signal  
D3 -----)  
D4 -----) 8000 - BFFF address slot select signal  
D5 -----)  
D6 -----) C000 - FFFF address slot select signal  
D7 -----)

### 8255 Port B (Input Port)

D0 - D7 ----- Keyboard Read Data

### 8255 Port C (Output Port)

D0 - D3 ----- Keyboard BCD Output Data  
                  (For Keyboard Scanning)  
D4 ----- Cassette Motor On  
                  (0 = Motor On, 1 = Motor Off)  
D5 ----- Cassette Write Data  
D6 ----- Cassette Audio Control  
                  (1 = Enable other channel in;  
                  (0 = Disable other channel in)  
D7 ----- Mix PSG Sound Data

### 8255 Control Word (Write Only)

Initial Set	D7	D6	D5	D4	D3	D2	D1	D0
	1	0	0	1	0	0	1	0

## 1.8 FLOPPY DISK CONTROLLER DESCRIPTION

### Functional Description

The Floppy Disk Formatter block diagram (STM-003-C). The primary sections include the Parallel Processor Interface and the Floppy Disk Interface.

**DATA SHIFT REGISTER** - This 8-bit register assembles serial data from the Read Data input (RAW READ) during Read operations and transfers serial data to the Write Data output during Write operations.

**DATA REGISTER** - This 8-bit register is used as a holding register during Disk Read and Write operations. In Disk Read operations, the assembled data byte is transferred in parallel to the Data Register from the Data Shift Register. In Disk Write operations information is transferred in parallel from the Data Register to the Data Shift Register.

When executing the Seek command the Data Register holds the address of the desired Track position. This register is loaded from the DAL and gated onto the DAL under processor control.

**TRACK REGISTER** - This 8-bit register holds the track number of the current Read/Write head position. It is incremented by one every time the head is stepped in (towards track 76) and decremented by one when the head is stepped out (towards track 00). The contents of the register are compared with the recorded track number in the ID field during disk Read, Write and Verify operations. The Track Register can be loaded from or transferred to the DAL. This Register should not be loaded when the device is busy.

**SECTOR REGISTER (SR)** - This 8-bit register holds the address of the desired sector position. The contents of the register are compared with the recorded sector number in the ID field during disk Read or Write operations. The Sector Register contents can be loaded from or transferred to the DAL. This register should not be loaded when the device is busy.

**COMMAND REGISTER (CR)** - This 8-bit register holds the command presently being executed. This register should not be loaded when the device is busy unless the new command is a force interrupt. The command register can be loaded from the DAL, but not read onto the DAL.



STATUS REGISTER (STR) - This 8-bit register holds device Status information. The meaning of the Status bits is a function of the type of command previously executed. This register can be read onto the DAL, but not loaded from the DAL.

CRC LOGIC - This logic is used to check or to generate the 16-bit Cyclic Redundancy Check (CRC). The polynomial is:

$$G(x) = x^{16} + x^{12} + x^5 + 1$$

The CRC includes all information starting with the address mark and up to the CRC characters. The CRC register is present to ones prior to data being shifted through the circuit.

ARITHMETIC/LOGIC UNIT (ALU) - The ALU is a serial comparator, incrementer, and decrementer and is used for register modification and comparisons with the disk recorded ID field.

TIMING AND CONTROL - All computer and Floppy Disk Interface controls are generated through this logic. The internal device timing is generated from an external crystal clock.

The FD1791/3 has 2 different modes of operation according to the state of DDEN. When DDEN = 0, double density (MFD) is assumed. When DDEN = 1, single density (FM) is assumed.

AM DETECTOR - The address mark detector detects ID, data and index address marks during read and write operations.

Whenever a Read or Write command (Type II or III) is received the FD179X samples the Ready input. If this input is logic low the command is not executed and an interrupt is generated. All type I commands are performed regardless of the state of the Ready input. Also, whenever a Type II or III command is received, the TG43 signal output is updated.

#### PROCESSOR INTERFACE

The interface to the processor is accomplished through the 8 Data Access Lines (DAL) and associated control signals. The DAL are used to transfer Data, Status, and Control words out of, or into the FD1793. The DAL are 3 state buffers that are enabled as output drivers when Chip Select (CS) and Read Enable (RE) are active (low logic state) or act as input receivers when CS and Write Enable (WE) are active.

When transfer of data with the Floppy Disk Controller is required by the host processor, the device address is decoded and CS is made low. The address bits A1 and A0, combined with the signals RE during a Read operation or WE during a Write operation are interpreted as selecting the following registers:

A1-A0	READ ( $\overline{RE}$ )	WRITE ( $\overline{WE}$ )
0 0	Status Register	Command Register
0 1	Track Register	Track Register
1 0	Sector Register	Sector Register
1 1	Data Register	Data Register

During Direct Memory Access (DMA) types of data transfers between the Data Register of the FD179X and the processor, the Data Request (DRQ) output is used in Data Transfer control. This signal also appears as status bit 1 during Read and Write operations.

On Disk Read operations the Data Request is activated (set high) when an assembled serial input byte is transferred in parallel to the Data Register. This bit is cleared when the Data Register is read after 1 or more characters are lost, by having new data transferred into the register prior to processor readout, the Lost Data bit is set in the Status Register. The Read operation continues until the end of sector is reached.

On Disk Write operations the Data Request is activated when the Data Register transfers its contents to the Data Shift Register, and requires a new data byte. It is reset when the Data Register is loaded with new data by the processor. If new data is not loaded at the time the next serial byte is required by the Floppy Disk, a byte of zeroes is written on the diskette and the Lost Data bit is set in the Status Register.

At the completion of every command an INTRQ is generated. INTRQ is reset by either reading the status register or by loading the command register with a new command. In addition, INTRQ is generated if a Force interrupt command condition is met.

## FLOPPY DISK INTERFACE

The 1793 has 2 modes of operation according to the state of DDEN (Pin 37). When DDEN = 1, single density is selected. In either case, the CLK input (Pin 24) is at 2 MHz. However, when interfacing with the mini-floppy, the CLK input is set at 1 MHz for both single density and double density. When the clock is at 2 MHz, the stepping rates of 3, 6, 10, and 15 ms are obtainable. When CLK equals 1 MHz these times are doubled.

## DISK READ OPERATIONS

Sector lengths of 128, 256, 512 or 1024 are obtainable in either FM or MFM formats. For FM, DDEN should be placed to logical "1". For MFM formats, DDEN should be placed to a logical "0". Sector lengths are determined at format time by a special byte in the "ID" field. If this sector length byte in the ID field is zero, then the sector length is 128 bytes. If 01 then 256 bytes. If 02, then 512 bytes. If 03, then the sector length is 1024 bytes. The number of sectors per track as far as the FD179 is concerned can be from 1 to 255 sectors. The number of tracks as far as the FD179 is concerned is from 0 to 255 tracks. For IBM 3740 compatibility, sector lengths are 128 bytes with 26 sectors per track. For System 34 compatibility (MFM), sector lengths are 256 bytes/sector with 26 sectors/track; or lengths of 1024 bytes/sector with 8 sectors/track. (See Sector Length Table)

For read operations, the FD1793 requires RAW READ Data (Pin 27) signal which is a 250 ns pulse per flux transition and a Read Clock (RCLK) signal to indicate flux transition spacings. The RCLK (Pin 26) signal is provided by some drives but if not it may be derived externally by Phase lock loops, one shots, or counter techniques. In addition, a Read Gate Signal is provided as an output (Pin 25) which can be used to inform phase lock loops when to acquire synchronizatin. When reading from the media in FM, RG is made true when 2 bytes of zeroes are detected. The FD1793 must find an address mark within the next 10 bytes; otherwise RG is reset and the search for 2 bytes of zeroes begins all over again. If an address mark is found within 10 bytes, RG remains true as long as the FD1793 is deriving any useful information from the data stream. Similarly for MFM, RG is made active when 4 bytes of "00" or "FF" are detected. The FD1793 must find an address mark within the next 16 bytes, otherwise RG is reset and search resumes.

During read operations (WG = 0), the VFOE (Pin 33) is provided for phase lock loop synchronization. VFOE will go active when:

- (a) Both HLT and HLD are True
- (b) Setting Time, if programmed, has expired
- (c) The 1793 is inspecting data off the disk

If WF/VFOE is not used, leave open or tie to a 10K resistor to +5.

#### DISK WRITE OPERATION

When writing is to take place on the diskette the Write Gate (WG) output is activated, allowing current to flow into the Read/Write head. As a precaution to erroneous writing the first data byte must be loaded into the Data Register in response to a Data Request from the FD1793 before the Write Gate signal can be activated.

Writing is inhibited when the Write Protect input is a logic low, in which case any Write command is immediately terminated, an interrupt is generated and the Write Protect status bit is set. The Write Fault input, when activated, signifies a writing fault condition detected in disk drive electronics such as failure to detect write current flow when the Write Gate is activated. On detection of this fault the FD1793 terminates the current command, and sets the Write Fault bit (bit 5) in the Status Word. The Write Fault input should be made inactive when the Write Gate output becomes inactive.

For write operations, the FD1793 provides Write Gate (Pin 30) and Write Data (Pin 31) outputs. Write data consists of a series of 500 ns pulses in FM (DDEN = 1) and 250 ns pulses in MFM (DDEN = 0). Write Data provides the unique address marks in both formats.

Also during write, 2 additional signals are provided for write precompensation. These are EARLY (Pin 17) and LATE (Pin 18). EARLY is active true when the WD pulse appearing on (Pin 30) is to be written early. LATE is active true when the WD pulse is to be written LATE. If both EARLY and LATE are low when the WD pulse is present, the WD pulse is to be written at nominal. Since write precompensation values vary from disk manufacturer to disk manufacturer, the actual value is determined by several 1 shots or delay lines which are located external to the FD1793. The write precompensation signals EARLY and LATE are valid for the duration of WD in both FM and MFM formats.

## WRITE PRECOMPENSATION

Write precompensation (which counteracts the 'drifting' when flux transitions are placed close together on the more cramped inside tracks) is available to the user on any tracks that he feels necessary.

## STATUS REGISTER SUMMARY

BIT	ALL TYPE 1 COMMANDS	READ ADDRESS	READ SECTOR
S7	Not Ready	Not Ready	Not Ready
S6	Write Protect	0	0
S5	Head Loaded	0	Record Type
S4	Seek Error	RNF	RNF
S3	CRC Error	CRC Error	CRC Error
S2	Track 0	Lost Data	Lost Data
S1	Index	DRQ	DRQ
S0	Busy	Busy	Busy
BIT	READ TRACK	WRITE SECTOR	WRITE TRACK
S7	Not Ready	Not Ready	Not Ready
S6	0	Write Protect	Write Protect
S5	0	Write Fault	Write Fault
S4	0	RNF	0
S3	0	CRC Error	0
S2	Lost Data	Lost Data	Lost Data
S1	DRQ	DRQ	DRQ
S0	Busy	Busy	Busy

STATUS FOR TYPE 1 COMMANDS

BIT NAME	MEANING
S7 NOT READY	This bit when set indicates the drive is not ready. When reset it indicates that the drive is ready. This bit is an inverted copy of the Ready input and 'ored' with MR. The Type II and III commands will not execute unless the logically 'ored' with MR.
S6 PROTECTED	When set, indicates Write Protect is activated. This bit is an inverted copy of WRPT input.
S5 HEAD LOADED	When set, it indicates the head is loaded and engaged. This bit is a logical "and" of HLD and HLT signals.
S4 SEEK ERROR	When set, the desired track was not verified. This bit is reset to 0 when updated.
S3 CRC ERROR	CRC encountered in ID field.
S2 TRACK 00	When set, indicates Read/Write head is positioned to Track 0. This bit is an inverted copy of the TR00 input.
S1 INDEX	When set, indicates index mark detected from drive. This bit is an inverted copy of the IP input.
S0 BUSY	When set command is in progress. When reset no command is in progress.

STATUS FOR TYPE II AND III COMMANDS

BIT NAME	MEANING
S7 NOT READY	This bit when set indicates the drive is not ready. When reset, it indicates that the drive is ready. This bit is an inverted copy of the Ready input and 'ored' with MR. The Type II and III commands will not execute unless the drive is ready.
S6 WRITE PROTECT	On Read Record: Not used. On Read Track: Not used. On any Write: It indicates a Write Protect. This bit is reset when updated.
S5 RECORD TYPE/WRITE FAULT	On Read Record: It indicates the record-type code from data field address mark. 1 = Deleted Data Mark. 0 = Data Mark. On any Write: It indicates a Write Fault. This bit is reset when updated.
S4 RECORD NOT FOUND (RNF)	When set, it indicates that the desired track, sector, or side were not found. This bit is reset when updated.
S3 CRC ERROR	If S4 is set, an error is found in one or more ID fields; otherwise it indicates error in data field. This bit is reset when updated.
S2 LOST DATA	When set, it indicates the computer did not respond to DRQ in one byte time. This bit is reset to zero when updated.
S1 DATA REQUEST	This bit is a copy of the DRQ output. When set, it indicates the DR is full on a Read Operation or the DR is empty on a Write operation. This bit is reset to zero when updated.
S0 BUSY	When set, command is under execution. When reset, no command is under execution.

CHAPTER 2

INPUT/OUTPUT SPECIFICATION

2.1 LIST OF CONNECTORS

PIN NAME	SPECIFICATION
1. Composite Video Output	RCA 2 pins connector
2. RF Modulated Signal	RCA 2 pins connector
Cassette	DIN 8 pins connector (DIN-45326)
I/O Port	AMP 9 pins connector
Printer	Amphenol 14 pins connector
Cartridge Bus	2.54 pace, 50 pins DGE connector for system expansion
Auáio	RCA 2 pins connector



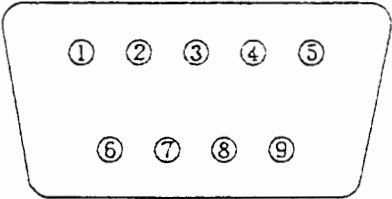
## 2.2 CASSETTE INTERFACE

- INPUT (To Computer)            From earphone terminal of tape recorder (white)
- OUTPUT (From Computer)        To microphone terminal of tape recorder (red)
- SYNCHRONIZATION                Asynchronous
- BAUD RATE                        1200 Baud (1200Hz - 1 wave "0", 2400Hz - 2 waves "1")  
2400 Baud (2400Hz - 1 wave "0", 4800Hz - 2 waves "1")  
(Tape recorder may have to be specified by maker when used 2400 Baud because it depends on cassette recorder quality.)
- MODULATION                      FSK
- REMOTE MOTOR CONTROL         Yes (Block)
- CONNECTOR                        DIN 45326 (8 pins)
- TABLE OF SIGNAL PINS

PIN NO.	SIGNAL NAME	DIRECTION	PIN CONNECTION
1	GND	---	
2	GND	---	
3	GND	---	
4	CMTOUT	OUTPUT	
5	CMTIN	INPUT	
6	REM +	OUTPUT	
7	REM -	OUTPUT	
8	GND	---	

### 2.3 INPUT/OUTPUT PORTS

- SPECIFICATION                   8-bit parallel
- INPUT/OUTPUT                   Input 4 bit, output 1 bit,  
bidirectional 2 bit per port
- LOGIC                            Active high
- LEVEL                            TTL
- CONNECTOR                      AMP 9 pins compatible
- LIST OF PINS

PIN NO.	SIGNAL NAME	DIRECTION	PIN CONNECTION
1	FWD	INPUT	
2	BACK	INPUT	
3	LEFT	INPUT	
4	RIGHT	INPUT	
5	+5V*	---	
6	TRG 1	INPUT/ OUTPUT	
7	TRG 2	INPUT/ OUTPUT	
8	OUTPUT	OUTPUT	
9	GND	---	

\* Current capacity is 50mA

## 2.4 PRINTER INTERFACE

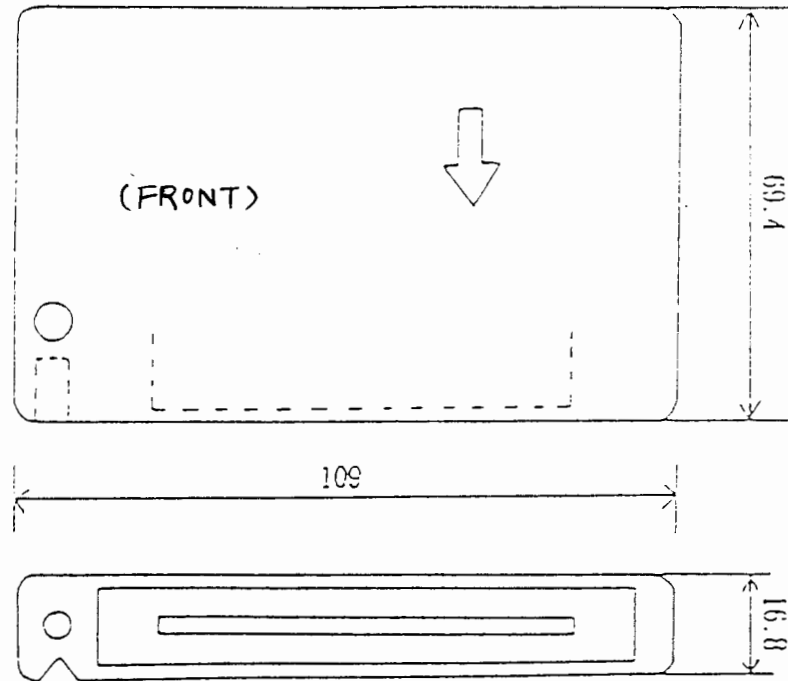
- SPECIFICATION                   8-bit parallel  
                                     (modified centronics type)
- LEVEL                            TTL
- CHARACTER CODE                Same to MSX display code
- CONNECTOR                     AMP 14 pins compatible
- LIST OF PINS

PIN NO.	SIGNAL NAME	PIN CONNECTION
1	$\overline{\text{STB}}$	
2	PDB0	
3	PDB1	
4	PDB2	
5	PDB3	
6	PDB4	
7	PDB5	
8	PDB6	
9	PDB7	
10	N.C.	
11	BUSY	
12	N.C.	
13	N.C.	
14	GND	

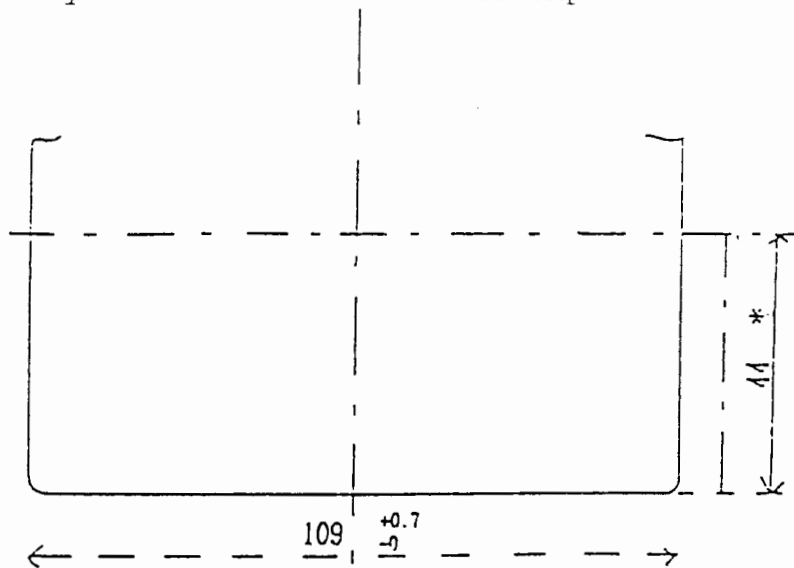
2.5 CARTRIDGE SLOT

2.5.1 SPECIFICATION OF CARTRIDGE

- Physical dimension of the standard cartridge



- Physical dimension of the expanded cartridge



Note: (#) The dimension beyond this size is not specified.

## 2.5.2 CARTRIDGE

### - LIST OF SIGNAL PINS

PIN NO.	NAME	I/O*	PIN NO.	NAME	I/O
1	$\overline{CS1}$	O	2	$\overline{CS2}$	O
3	$\overline{CS12}$	O	4	$\overline{SLTSL}$	O
5	RESERVED #	-	6	$\overline{RFSH}$	O
7	$\overline{WAIT}$ %	I	8	$\overline{INT}$ %	I
9	$\overline{M1}$	O	10	$\overline{BUSDIR}$	I
11	$\overline{IORQ}$	O	12	$\overline{MERQ}$	O
13	$\overline{WR}$	O	14	$\overline{RD}$	O
15	$\overline{RESET}$	O	16	RESERVED #	-
17	A9	O	18	A15	O
19	A11	O	20	A10	O
21	A7	O	22	A6	O
23	A12	O	24	A8	O
25	A14	O	26	A13	O
27	A1	O	28	A0	O
29	A3	O	30	A2	O
31	A5	O	32	A4	O
33	D1	I/O	34	D0	I/O
35	D3	I/O	36	D2	I/O
37	D5	I/O	38	D4	I/O
39	D7	I/O	40	D6	I/O
41	GND	-	42	CLOCK	O
43	GND	-	44	SW1	-
45	+5V	-	46	SW2	-
47	+5V	-	48	+12V	-
49	SOUNDIN	I	50	-12V	-

\* The direction of Input/Output is based on basic unit side.

# Reserved PIN must not be used.

% OPEN COLLECTOR output

- SIGNAL PIN ILLUSTRATION

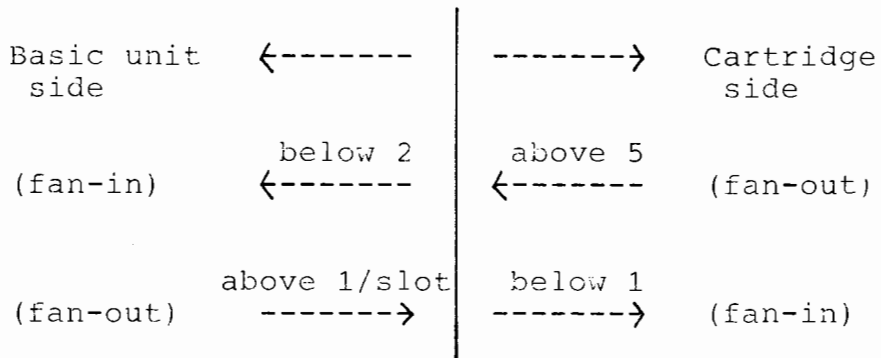
PIN NO.	NAME	DESCRIPTION
1	<u>CS1</u> %	ROM 4000-7FFF selected signal
2	<u>CS2</u> %	ROM 8000-BFFF selected signal
3	<u>CS12</u> %	ROM 4000-BFFF selected signal (for 256K Bit ROM)
4	<u>SLTSL</u>	Slot select signal
5	RESERVED	For future use only. Do not use this pin.
6	<u>RFSH</u>	Refresh signal
7	<u>WAIT</u>	Wait signal to CPU
8	<u>INT</u>	Interrupt request signal
9	<u>MI</u>	Fetch cycle signal of CPU
10	<u>BUSDIR</u>	This signal controlled the direction of external data bus buffer when the cartridge is selected. It is low level when the data is sent by the cartridge.
11	<u>IORQ</u>	I/O request signal
12	<u>MERQ</u>	Memory request signal
13	<u>WR</u>	Write signal
14	<u>RD</u>	Read signal
15	RESET	System reset signal
16	RESERVED	For future use only. Do not use this pin.
17-32	A0-A15	Address bus
33-40	D0-D7	Data bus
41	GND	Ground
42	CLOCK	CPU clock 3.579MHz
43	GND	Ground
44, 46	SW1, SW2	Insert/remove detect for protection
45, 47	+5V	+5V power supply
48	+12V	+12V power supply
49	SOUNDIN	Sound input (-5dBm)
50	-12V	-12V power supply

% Note that CS signals imply memory request and read signal

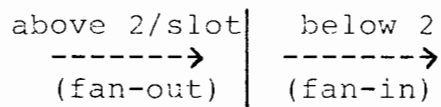
### 2.5.3 CONDITIONAL OF CARTRIDGE CONNECTION

- Fan-in, fan-out (LS-TTL load)

Data and Address bus



- Control signals



- Voltage level TTL level

### 2.5.4 POWER CAPACITY

+5V	300mA/Slot
+12V	50mA
-12V	50mA

## 2.6 SOUND

- LSI AY-3-8910 compatible
- OCTAVE 8 octave (3 voices output)
- SOUND EFFECT Yes
- SOFTWARE SOUND OUTPUT 1 bit from output port
- OUTPUT LEVEL -5dBm (If system has output connector)
- CONNECTOR RCA 2 pins (If system has audio output connector)



2.7 SVI-728 EXPANSION MODULE INTERFACE  
AND GAME SLOT SIGNAL DESCRIPTION

PIN NO.	NAME	DESCRIPTION
1	$\overline{\text{CS 1}}$	ROM 4000 - 7FFF select signal
2	$\overline{\text{CS 2}}$	ROM 8000 - BFFF select signal
3	$\overline{\text{CS 12}}$	ROM 4000 - BFFF select signal
4	$\overline{\text{SLTSL}}$	Slot selected signal. Fixed select signal for each slot.
5		Reserved for future use only
7	$\overline{\text{WAIT}}$	Wait signal to CPU
8	$\overline{\text{INT}}$	Interrupt request signal
9	$\overline{\text{M 1}}$	Fetch cycle signal of CPU
10	$\overline{\text{BUSDIR}}$	This signal controls the direction of external data bus buffer when the cartridge is selected. It is low level when the data is sent by the cartridge.
11	$\overline{\text{IORQ}}$	I/O request signal
12	$\overline{\text{MERQ}}$	Memory request signal
13	$\overline{\text{WR}}$	Write signal
14	$\overline{\text{RD}}$	Read signal
15	$\overline{\text{RESET}}$	System reset signal
16		Reserved for future use only
17-32	A0-A15	Address bus
33-40	D0-D7	Data bus
41	GND	Ground
42	CLOCK	CPU clock 3.579MHz
43	GND	Ground
44,46	SW1,SW2	Insert/remove protect
45,47	+5V	+5V power supply
48	+12V	+12V power supply
49	SUNDIN	Sound input (-5dbm)
50	-12V	-12V power supply

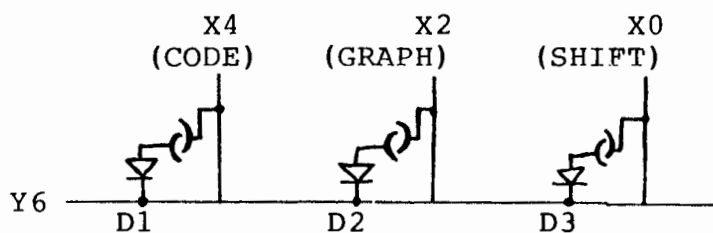
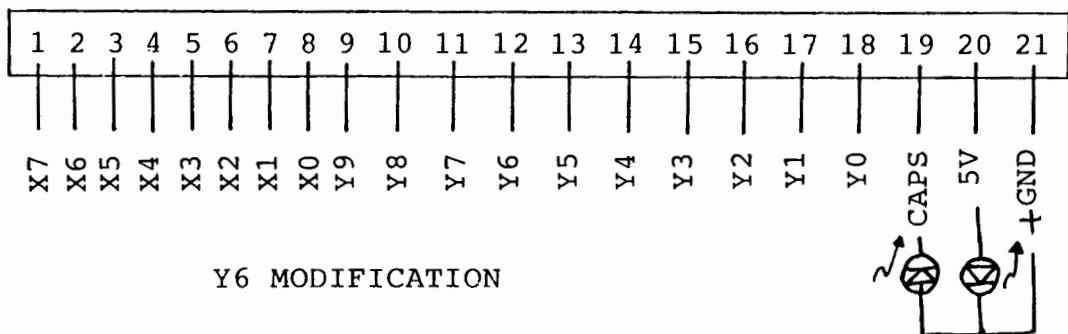
## 2.8 KEYBOARD

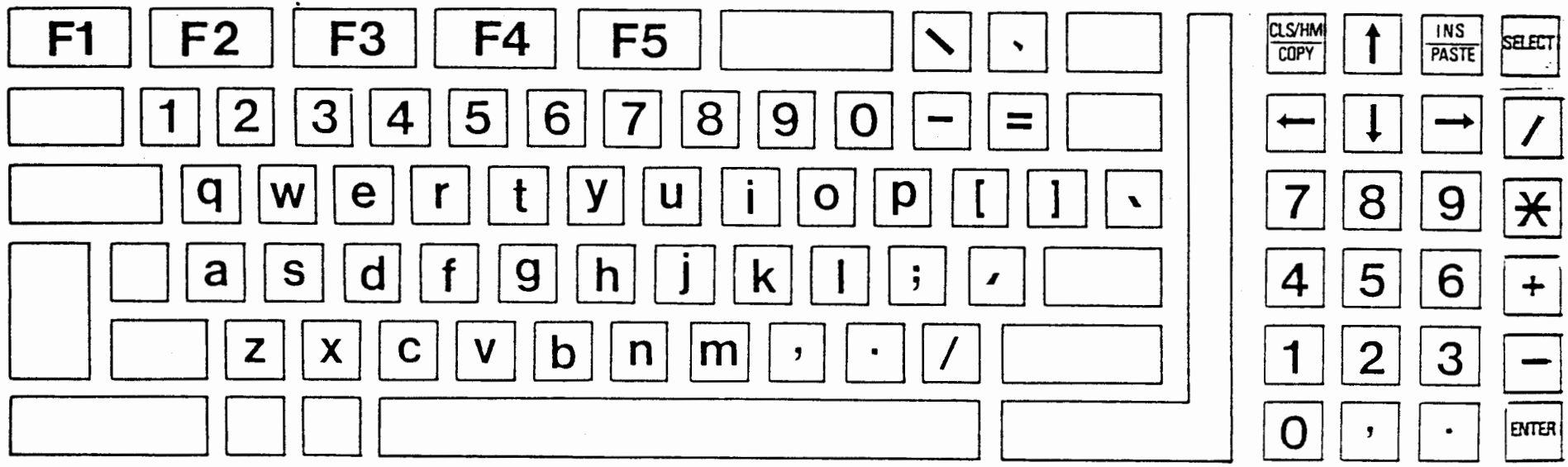
- Layout : Alphanumerical: ASCII standard
- Scanning : Software scanning
- Number of keys : 90
- Matrix diagram :

	X7	X6	X5	X4	X3	X2	X1	X0
Y0	7 0	6 ^	5 %	4 \$	3 #	2 @	1 !	0 )
Y1	; : b B	] [ } a A	[ ] { " "	\ /   - _	= + _ -	- = > <	9 ( ) ~	8 * "
Y2	j J r R	i I q Q	h H p P	g G o O	f F n N	e E m M	d D l L	c C k K
Y3	z Z y Y	y Y x X	x X w W	w W v V	v V u U	u U t T	t T s S	s S
Y4	F3	F2	F1	CODE	CAPS	GRPH	CTRL	SHIFT
Y5	RETURN	SELECT	BS	STOP	TAB	ESC	F5	F4
Y6	→	↓	↑	←	DEL	INS	HOME	SPACE
Y7					/ (k)	* (k)	- (k)	+ (k)
Y8								
Y9								

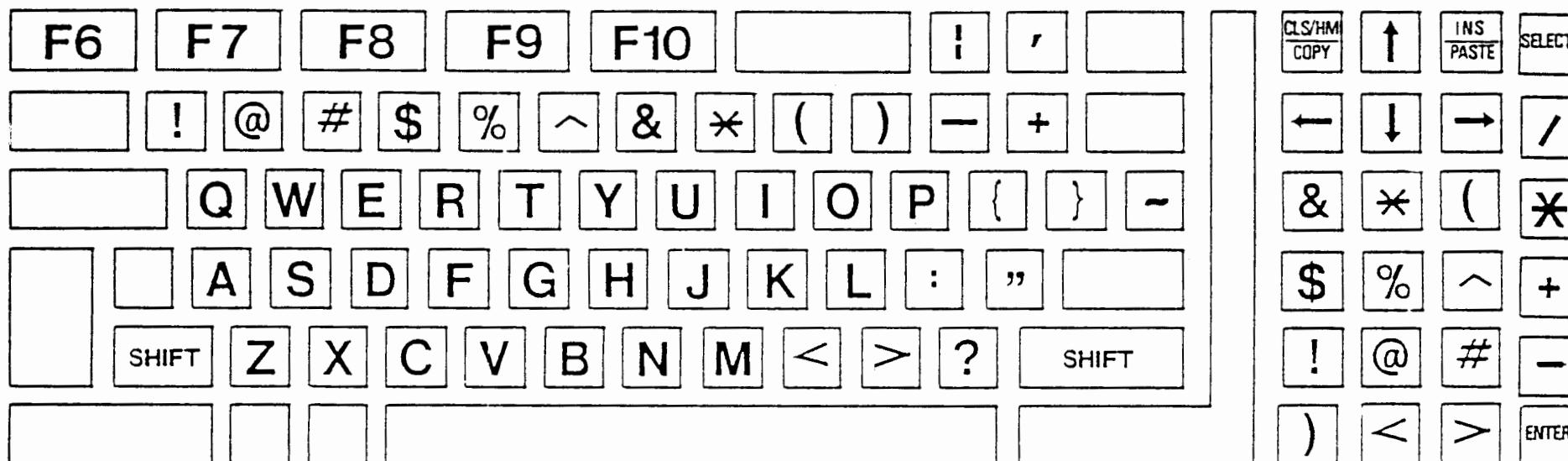
MSX international version 1:1 keyboard matrix

(k) Keys in numerical key pad





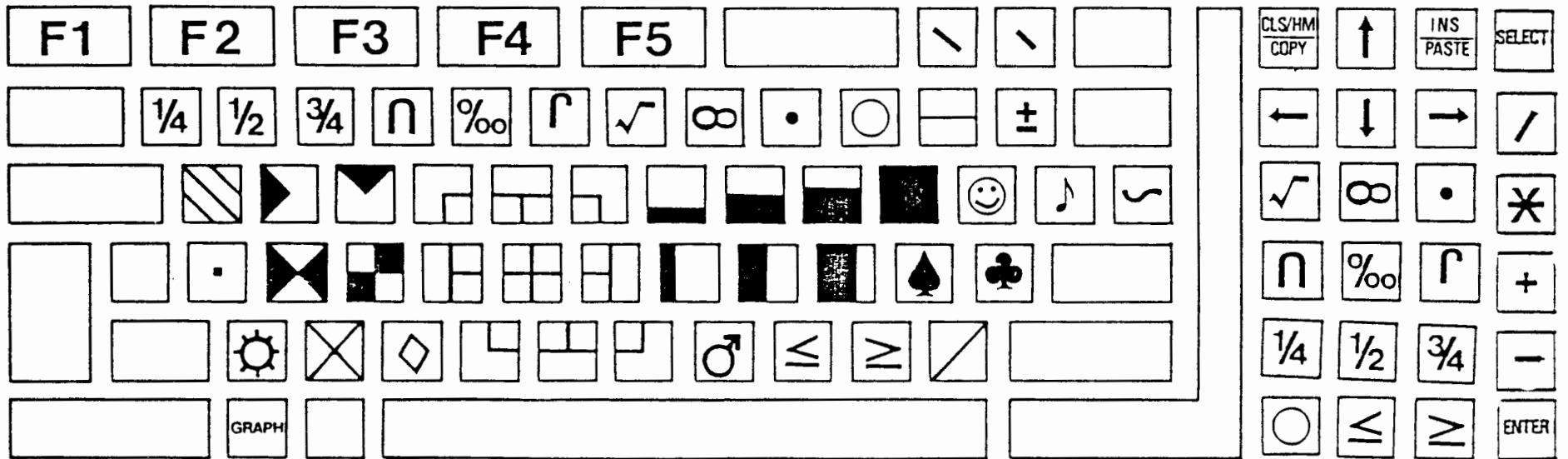
STANDARD KEYBOARD LAYOUT



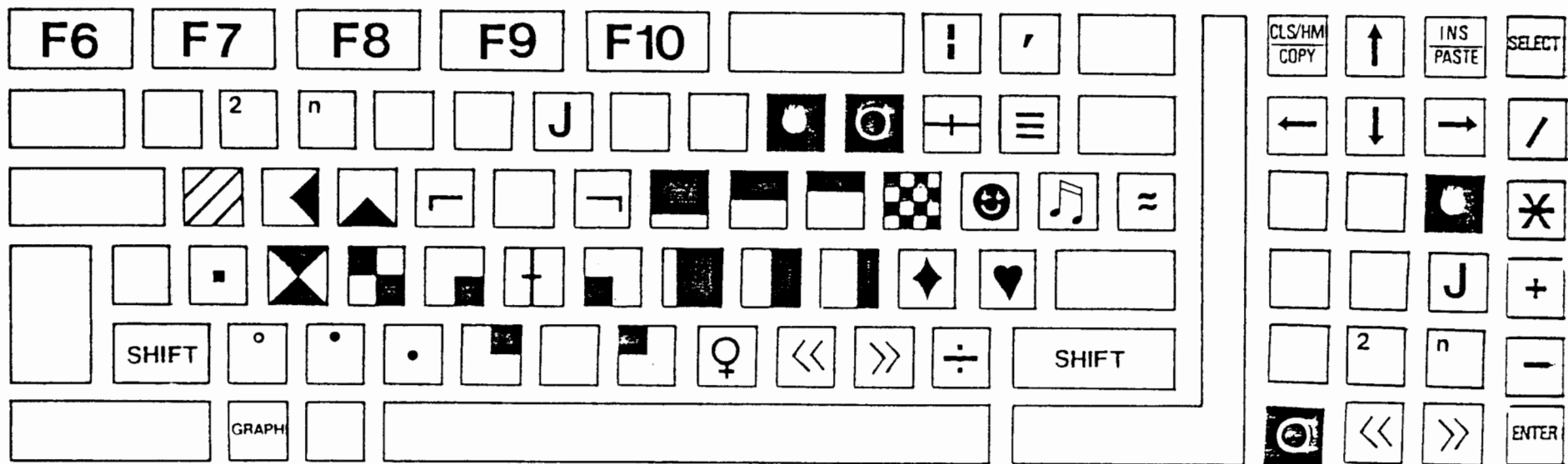
STANDARD KEYBOARD LAYOUT  
WITH SHIFT KEY PRESSED



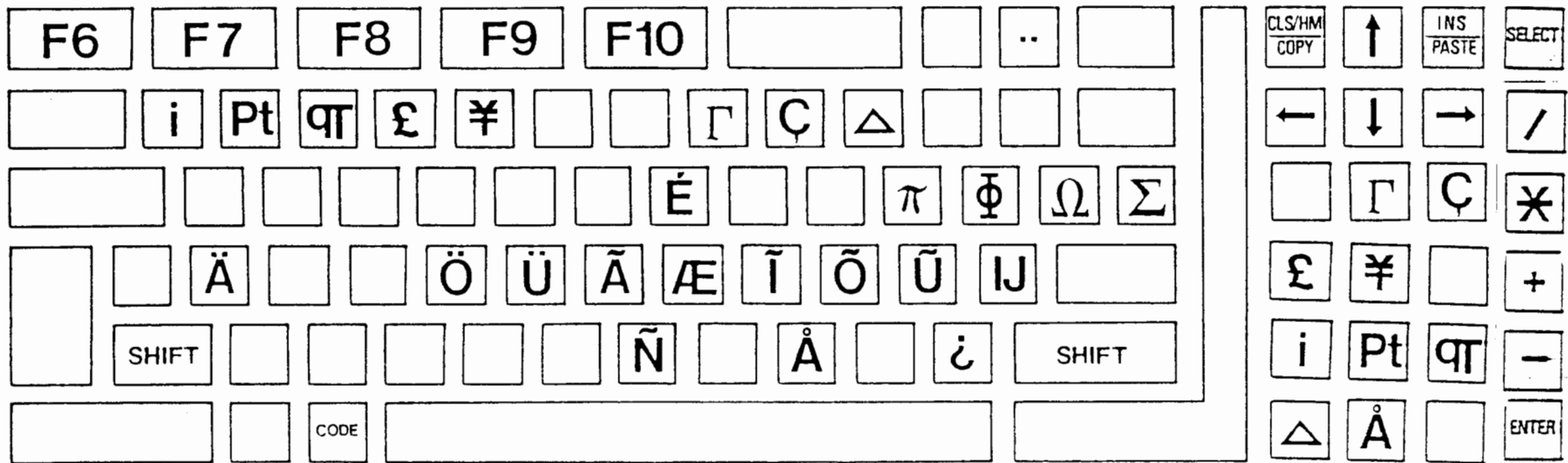
STANDARD KEYBOARD LAYOUT  
WITH CODE KEY PRESSED



STANDARD KEYBOARD LAYOUT  
WITH GRAPH KEYS PRESSED



STANDARD KEYBOARD LAYOUT  
WITH SHIFT & GRAPH KEYS PRESSED



STANDARD KEYBOARD LAYOUT  
WITH SHIFT & CODE KEYS PRESSED

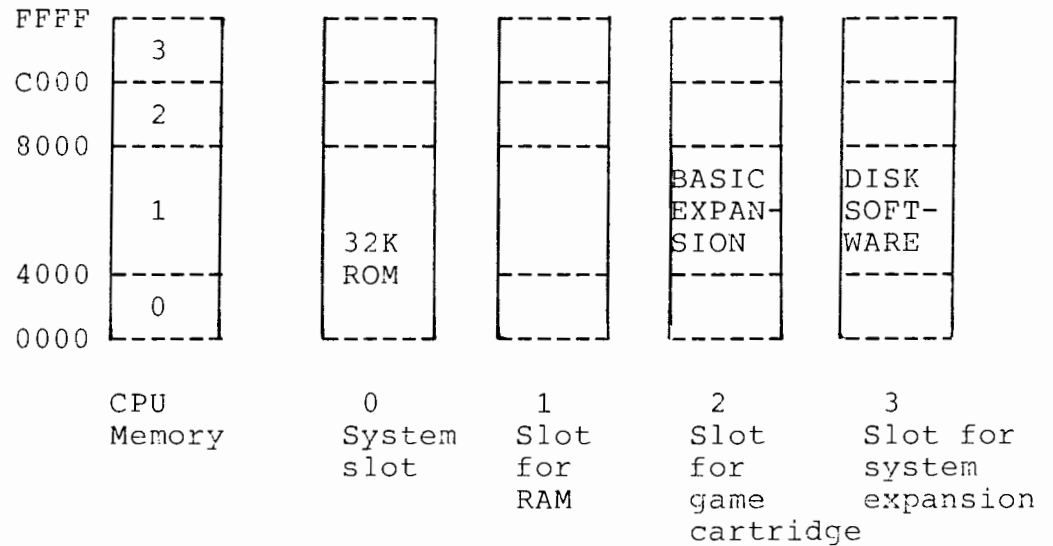


## CHAPTER 3

### MEMORY SYSTEM

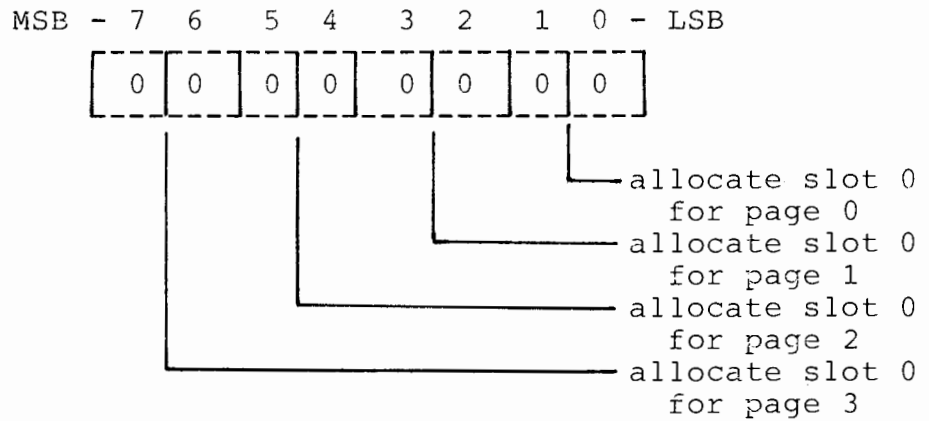
#### 3.1 MEMORY MAP

- Following is the memory map for SVI-728 computer.



- MSX BASIC uses the largest available RAM area that is installed from FFFF to 8000 contiguous for its system working RAM area. This can be placed on any slots including expanded slots.
- Slot select register, which is the port A of 8255, maps physical memory space to the logical CPU memory space in 16K bytes unit (page).

For example, following value in slot select register allocates page 0 and 1 from slot 0, page 2 from slot 2 and page 3 from slot 0.



The physical memory is always allocated to same memory page in the CPU address space. It is not possible to allocate to different page like page 3 of slot 3 to page 0 of CPU memory space.

NOTES: The meaning of "slot" does not imply that it must have connector for cartridge, however, the slot for cartridge must have connector, of course. Refer to APPENDIX B I/O INTERFACE.

### 3.2 I/O MAP

#### I/O ADDRESS DEVICE

FF	I/O ADR	RW	DESCRIPTION	REMARK	
	&HA8	W R	PORT A DATA WRITE PORT A DATA READ	8255A COMPA- TIBLE	
	&HA9	W R	PORT B DATA WRITE PORT B DATA READ		
	&HAA	W R	PORT C DATA WRITE PORT C DATA READ		
	&HAB	W	MODE SET		
B0	PPI	&HA0	W	ADDRESS LATCH	AY-3-8910
A8		&HA1	W	DATA WRITE	COMPA-
	PSG	&HA2	R	DATA READ	TIBLE
A0	VDP	&H98	W R	V-RAM DATA WRITE V-RAM DATA READ	9918A COMPA- TIBLE
98	(*)				
90		&H99	W R	COMMAND, ADDRESS SET STATUS READ	
	(**)				
80	80	&H90	W R	STROBE OUTPUT (b0) STATUS INPUT (b1)	LATCH OUT- PUT BUSY '1'
	COLUMN				
	DISPLAY	&H91	W	PRINT DATA	LATCH OUT- PUT
78	NOT				
	SPECI-	&H89	R	TONE DIAL DATA PORT	
	FIED	&H88	R	MODEM CONTROL PORT	
00		&H80	W R	DATA WRITE DATA READ	I-8251A COMPA- TIBLE
		&H81	W R	COMMAND/MODE SET STATUS READ	
		&H78	R	ADDRESS LATCH OF 6845 (CRT CONTROLLER)	
		&H79	R/W	DATA READ/WRITE OF 6845	

- I/O address 80 - FF are assigned for system usage. The empty area are reserved for system use.

Although these addresses are defined here, any software should not access those devices directly through the addresses listed above. Every access to the I/O must be done through the BIOS calls. This is because to keep every software independent from the hardware difference.

Only exception is the access to VDP. Location 6 and 7 of MSX system ROM contains read and write address of VDP register. Those softwares that have to access VDP very quickly may access VDP directly through those addresses stored in ROM.

NOTE: (\*) PRINTER  
(\*\*) RS-232C

### 3.3 PPI BIT ASSIGNMENT

PORT	BIT	I/O	SIGNAL NAME	DESCRIPTION
A	0	↑ O ↓	CS0L	0000 - 3FFF address slot select signal
	1		CS0H	
	2		CS1L	4000 - 7FFF address slot select signal
	3		CS1H	
4	O	CS2L	8000 - BFFF address slot select signal	
5		CS2H		
	6	↓	CS3L	C000 - FFFF address slot select signal
	7		CS3H	
B	0   7	I		Keyboard return signal
C	0	↑	KB0	Keyboard scan signal
	1		KB1	
	2		KB2	
	3	KB3		
	4	O	CASON	
	5	↓	CASW	Cassette write signal
	6		CAPS	CAPS lamp signal ("L" -- ON)
	7		SOUND	Sound output by software

### 3.4 PSG BIT ASSIGNMENT

PORT	BIT	I/O	CONNECTOR PIN NO.	SIGNALS FOR JOYSTICK
A	0		J3-1 PIN	
	1		J4-1 PIN*	#2 FWD2
			J3-2 PIN	#1 BACK1
	2		J4-2 PIN*	#2 BACK2
			J3-3 PIN	#1 LEFT1
	3		J4-3 PIN*	#2 LEFT2
			J3-4 PIN	#1 RIGHT1
	4		J4-4 PIN*	#2 RIGHT2
		J3-6 PIN	#1 TRGA1	
5		J4-6 PIN*	#2 TRGA2	
		J3-7 PIN	#1 TRGB1	
6		J4-7 PIN*	#2 TRGB2	
	7		CSAR (CASSETTE TAPE READ)	
B	0		J3-6 PIN	#3
	1		J3-7 PIN	#3 "H" LEVEL
	2		J4-6 PIN*	#3
	3		J4-7 PIN*	#3
	4		J3-8 PIN	
	5		J4-8 PIN*	
	6		PORT A (INPUT SELECT)	
	7			

#1 Available when bit 6 of port B is low used by JOYSTICK 1

#2 Available when bit 6 of port B is high used by JOYSTICK 2

#3 Make the "H" level, when used as an output port. Tied an open collector buffer to the output. (Refer to APPENDIX C-1)

Remark      PIN5 +5V  
              PIN9 GND

- On the minimum system, signals are not connected to J4

### 3.5 I/O MAPPING OF MSX DRIVE

I/O ADDRESS	R/W	DESCRIPTION
7FB8	W	Command port of 1793
7FB8	R	Status port of 1793
7FB9	R/W	Track register of 1793
7FBA	R/W	Sector register of 1793
7FBB	R/W	Data register of 1793
7FBC	R	(MSB) Bit 7 - interrupt of 1793 (1 for interrupt)
		Bit 6 - data request of 1793 (0 for request)
7FBC	W	Bit 3 - motor on (1 for ON)
		Bit 2 - side select (0 for side 0)
		(LSB) Bit 0 - drive enable (1 for ENABLE)
7FBE		Software switch to turn on CP/M, boot ROM and turn off MSX DOS ROM.
7FBF		Software switch to turn off CP/M, boot ROM and turn on MSX DOS ROM.

## CHAPTER 4

### SVI-728 PERIPHERALS

#### 4.1 SVI-707 MSX DISK DRIVE (320K)

SVI-707 MSX Disk Drive (320K) is your gateway into the expanding universe of MSX computer software and programming tools. It is very useful as an external memory unit because of its large memory capacity and high access speed.

The 5.25" double-sided, double density disk drive provides 320K formatted memory capacity to utilize disk BASIC, most current CP/M, MSX DOS. The built-in disk drive controller allows it to hook up directly to the computer.

With a 13K spooler buffer when running in CP/M mode, it can read/write Kaypro II, Osborne I, Bondwell 12/14, and all single or double-sided disks formatted for the SVI-328.

#### SPECIFICATION

Drive Unit	:	Single
Floppy Diskette	:	- Double sided - Double density - Soft sectored 5.25" diskette
Memory Capacity		
- Unformatted	:	500K bytes
- Formatted	:	326K bytes
Disk Operating System	:	- MSX DOS - CP/M DOS
Rotational Speed	:	300 rpm
Disk Format	:	CP/M - 40 tracks/side - 17 sectors/track - 256 bytes/sector  MSX DOS - 40 tracks/side - 9 sectors/track - 512 bytes/sector



Access Time : - Track-to-track 26 msec  
- Setting time 20 msec  
- Motor start time 350 msec

Recording Density : 5536 bpi

Track Density : 48 tpi

Encoding Method : MFM

Power Source : 110V/220V  
(Separate power supply)

Humidity : 20% - 80%

Dimensions : 162 (W) x 240 (D) x 63 (H)

For trouble shooting, please refer to Service & Technical Manual for Disk Drive of SVI-707 - MSX Disk Drive.

#### 4.2 SVI-727 MSX 80 COLUMN VIDEO CARTRIDGE

SVI-727 MSX 80 Column Video Cartridge allows you to switch the video display from 40 column to 80 column. The cartridge is MSX compatible, you can use it with the SVI-728 or other MSX computer.

With this cartridge, any sophisticated CP/M programmes already available on the market can be run on your computer.

#### SPECIFICATION

Character : - 7 x 9 dot matrix with 2 blank dots between each character  
- 128 ASCII character set with inverse characters

Display Foramt : 80 column x 24 row

Video Output : composite video 2V p-p

Power Consumption : + 5V 300mA  
+ 12V 50mA

Dimensions (mm) : 170 (L) x 123 (W) x 32 (D)

Specifications for product improvement are subject to change without notice.

#### 4.3 SVI-737 MSX 300 BAUD MODEM WITH RS-232 INTERFACE

SVI-737 MSX 300 Baud Modem with RS-232 is a modem plus built-in RS-232 interface.

The Modem which is BELL 103 compatible, can be directly coupled to telephone line. Auto-answering and full-duplex operation is carried out at 300 bps transmission rate. The RS-232 is Electronic Industrial Association (EIA) standard with selectable baud rate, word length, stop bits and parity bit.

SVI-737 is MSX compatible, you can connect it to SVI-728 or other MSX computer.

#### SPECIFICATION

##### Modem

Standard : CCITT V2.1 standard

Data Rate : 300 baud

Communication Mode : Asynchronous, full-duplex

Telephone Line Interface : FCC approved (pending) direct connection to 600 ohms line impedance

Modulation : Frequency shift keying (FSK)

Operation Functions : Originate - manual  
Answer - manual/automatic

Transmit Frequencies : Originate - mark : 1270 Hz  
space : 1070 Hz  
Answer - mark : 2225 Hz  
space : 2025 Hz

Receive Frequencies : Originate - mark : 2225 Hz  
space : 2025 Hz  
Answer - mark : 1270 Hz  
space : 1070 Hz

Receiver Sensitivity : -48 dBm

Transmitter Output Level : -9 dBm at 600 ohms line impedance

## RS-232 Interface

Baud Rate : 50 - 19200 bps  
(15 different baud rates)

Counter Channel Usage : CH0 - Receive baud rate clock  
CH1 - Transmit baud rate clock  
CH2 - Used by application  
(e.g. time out counter)

Power Consumption : + 5V 250mA  
+ 12V 30mA  
- 12V 28mA

Dimensions (mm) : 170 (L) x 123 (W) x 32 (D)

Specifications for product improvement are subject to change without notice.

#### 4.4 SVI-747 MSX 64K RAM MEMORY CARTRIDGE

SVI-747 MSX 64K RAM Memory Cartridge provides continuous 64K bytes memory to your computer. Since it is MSX compatible, it can be connected to SVI-728 or other MSX computer.

With the additional memory capacity, you can run a lot of software. It plugs into the cartridge or cartridge expansion slots of all MSX computers. It can be used as a RAM disk in CP/M system with the SVI-707.

#### SPECIFICATION

Memory : 64K RAM  
Power Consumption : 5V 200mA  
Dimensions (mm) : 170 (L) x 123 (W) x 32 (D)

N.B. - No additional power supply

- Adapt to game slot of SVI-728 MSX computer
- Provide continuous 64K byte memory for those MSX computers which do not have 64K memory built-in

Specifications for product improvement are subject to change without notice.

#### 4.5 SVI-757 MSX RS-232 INTERFACE CARTRIDGE

SVI-757 MSX RS-232 Interface Cartridge provides Electronic Industrial Association (EIA) standard for data communication between your computer and external equipment.

With the installed cartridge, your computer can link up with another computer, modem, teletype equipment or printer etc. Since the cartridge is MSX compatible, you can connect it to the SVI-728 or other MSX computer.

#### SPECIFICATION

Baud Rate	:	50 - 19200 bps (15 different baud rate)
Counter Channel Usage	:	CH0 - Receive baud rate clock CH1 - Transmit baud rate clock CH2 - Used by application (e.g. time out counter)
Power Consumption	:	+ 5V 250mA + 12V 25mA - 12V 28mA
Dimensions (mm)	:	170 (L) x 123 (W) x 32 (D)

#### 4.6 SVI-767 MSX DATA CASSETTE

The SVI-767 is an inexpensive device that uses ordinary cassette tape to add storage and retrieval capability to the computer system. The built-in Automatic Level Control (ALC) ensures reliable and high quality recording. Tape counter, LED indicator and auto-stop are standard features.

Specifications for product improvement are subject to change without notice.

#### 4.7 SVI-709 NETWORK INTERFACE CARD

This cartridge transforms your computer into a station in the SVI-609 Local Area Network. These stations are given access to the mass-storage and rapid printing resources of the SVI-609. Each station can make use of a portion of the SVI-609's 10M-byte hard disk for its own file storage, and all stations can share a common storage area, printer spooler, and software library.

#### 4.8 SVI-101 MSX JOYSTICK

The SVI-101 MSX Joystick has 2 separate fire buttons for the many new games that require 2 buttons to play. An improvement of the original award-winning "QuickShot" design, the SVI-101 enhances the excitement and arcade flavour of your game play. It's a sure winner!

#### SVI-102 MSX JOYSTICK

The SVI-102 MSX Joystick is the upgraded model of the original QuickShot. It has a locking auto fire switch and dual fire buttons. Top fire button is an extra large, curving back towards rear of joystick to give more comfortable and more fun.

Specifications for product improvement are subject to change without notice.

## CHAPTER 5

### DISSASSEMBLY/ASSEMBLY

This chapter contains step by step procedure for disassembling and reassembling of SVI-728. Read this chapter over carefully and follow the procedure given before attempting to disassemble the computer.

Figures 5-1 and 5-2 show all the I/O Ports of SVI-728.

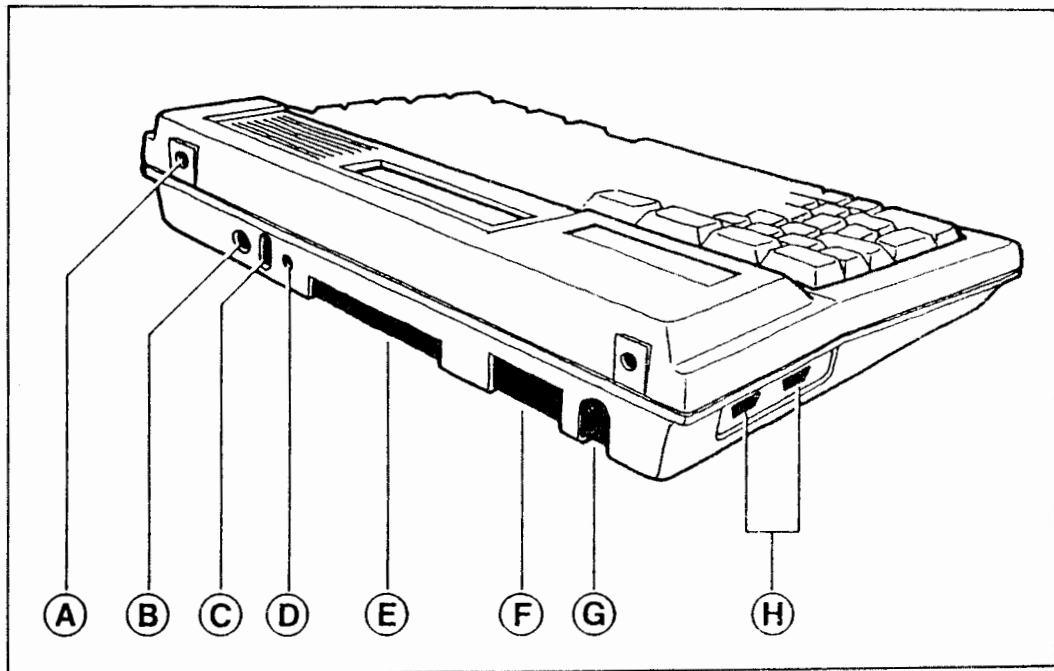


Figure 5-1 Location of I/O Ports of SVI-728

- |                            |                               |
|----------------------------|-------------------------------|
| A. Reserve Opening         | E. Expansion Slot             |
| B. Modulator Output Port   | F. Printer Port               |
| C. Audio/Video Output Port | G. Cassette Input/Output Port |
| D. Ground Port             | H. Dual Joystick Ports        |



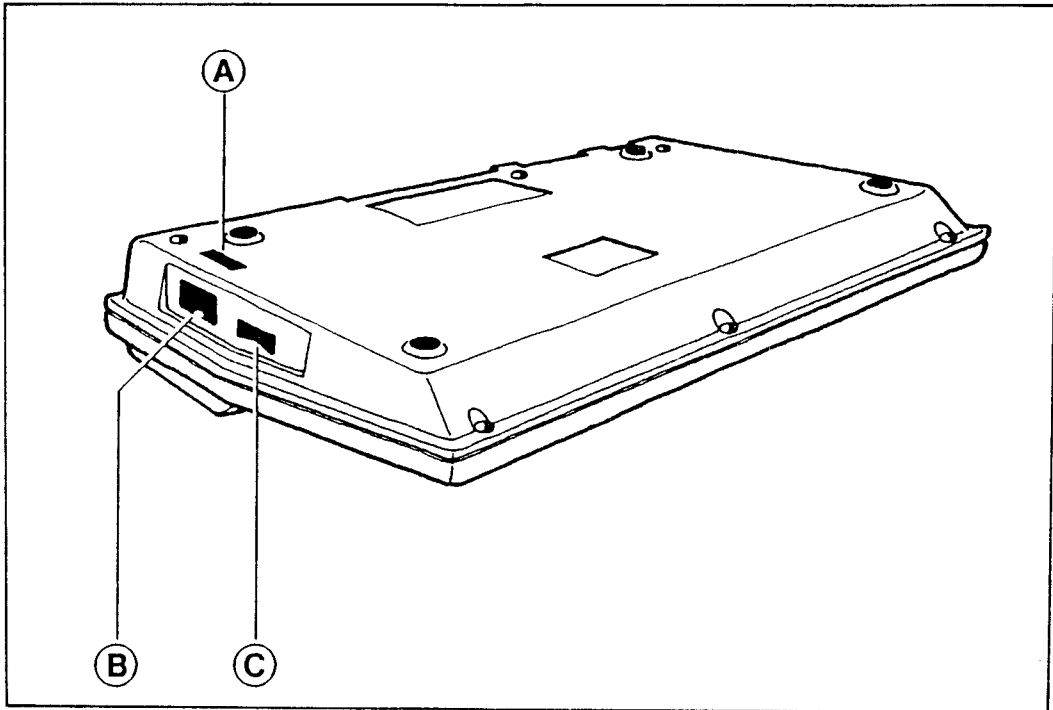


Figure 5-2 Location of Channel Switch, Power Socket and Power Switch of SVI-728

A. Channel Switch\*  
B. Power Socket

C. Power Switch

\* Available for some countries only.

### 5.1 General Caution

- (1) Be very careful about mixing screws. There are 2 types of screws; self-tap and machine screw. If wrong size or type of screws are used permanent damage to both the screws and the plastic housing may result.
- (2) Never tighten the screws too tight, this will strip the plastic housings for the self-tap screws.
- (3) Be very careful with the Flex Cable; never fold them.
- (4) Use soft-padding when turning the Console bottom up, in order to prevent damage to the plastic surfaces.
- (5) Static control precautions must be used when handling any Printed Circuit Board.

- (6) Never switch on the power when the Heat Sink Plate is removed, this will cause damage to IC T10 and to IC T5.
- (7) Make sure that the computer is free from all external cables before starting the disassembling procedure.

## 5.2 KEYBOARD AND PCB ACCESS

- (1) Turn the Console bottom up.
- (2) Loosen and remove 6 self-tap screws from the Bottom Housing (See figure 5-3).

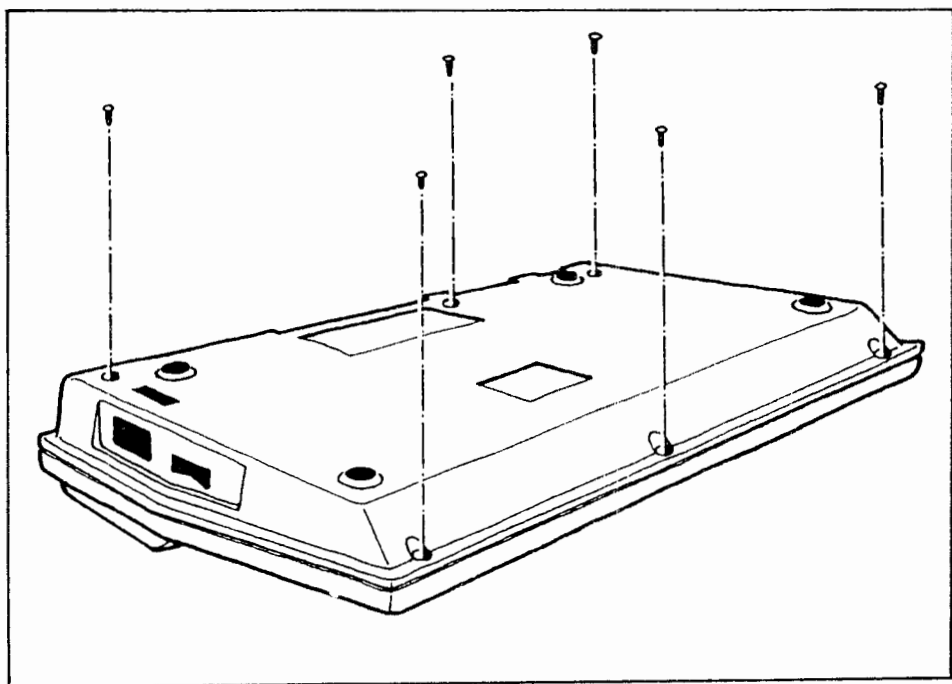


Figure 5-3 Location of Screws on Bottom Housing

- (3) Turn the Console up right again.
- (4) With the front of the Console facing you, carefully lift up the Keyboard and flip it over towards you. (See Figure 5-4)
- (5) In some countries, a metal shielding cover is required. It can be removed by loosening the self-tap screws found on it.

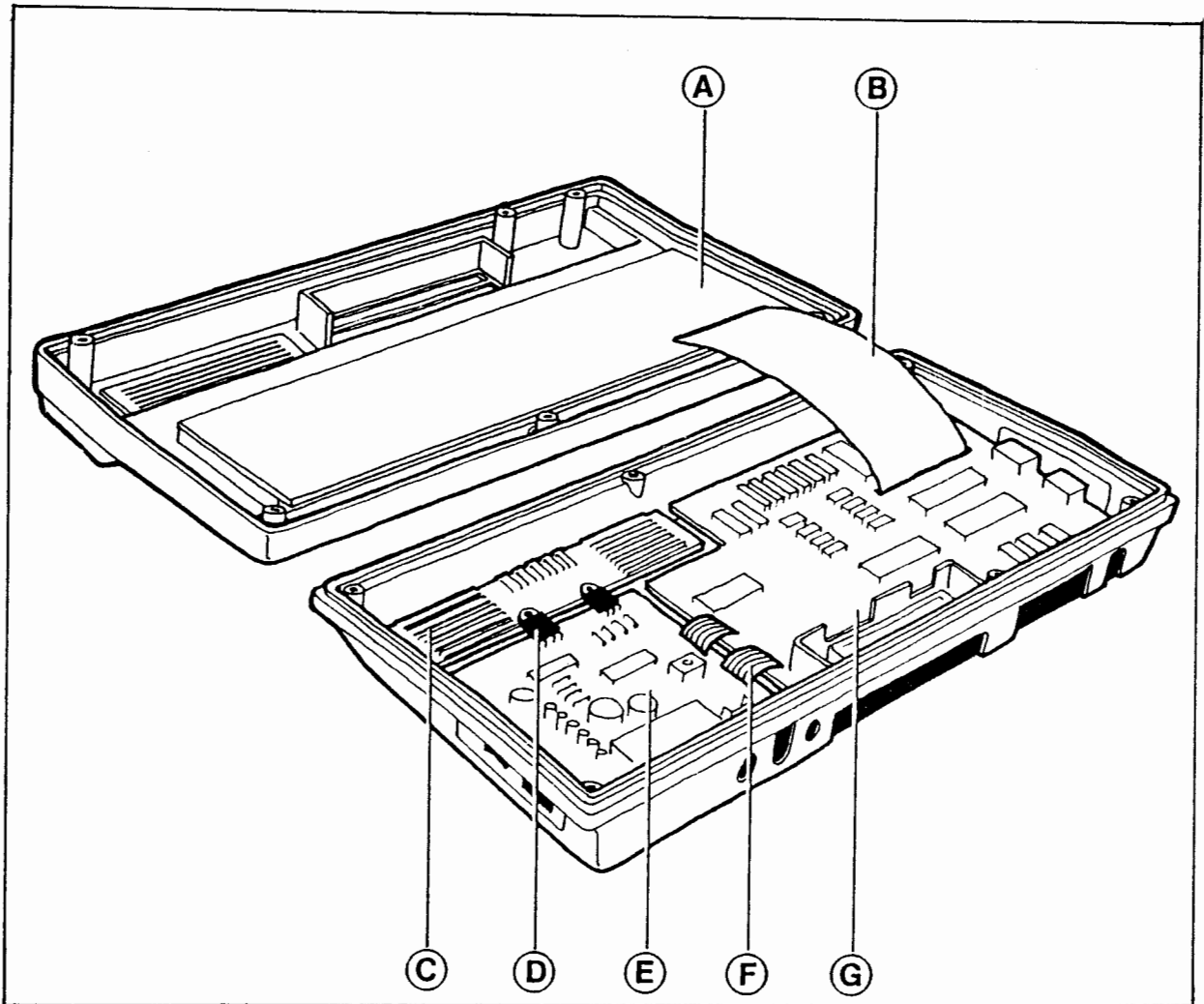


Figure 5-4 Internal Components of SVI-728

- |                              |                          |
|------------------------------|--------------------------|
| A. Keyboard PCB              | E. Video and Power Board |
| B. Flex Cable                | F. Flex Cable            |
| C. Heat Sink Plate           | G. Logic Board           |
| D. Power Transistor (IC T10) |                          |

### 5.3 REASSEMBLE OF THE CONSOLE

- (1) Carefully flip the Keyboard up right so that it is on top of the Bottom Housing.
- (2) Gently (do not force) lower the Keyboard onto the Bottom Housing, until the 2 snap in place together.
- (3) Turn the Console up side down.
- (4) Tighten the 6 self-tap screws. (See Figure 5-3)
- (5) Turn the unit right side up again when done.

#### 5.4 REMOVAL OF THE HEAT SINK PLATE

CAUTION: Never switch the Power on when the Heat Sink Plate is removed.

- (1) Loosen the 2 machine screws attached to IC T5 and IC T10. (See Figure 5-5)

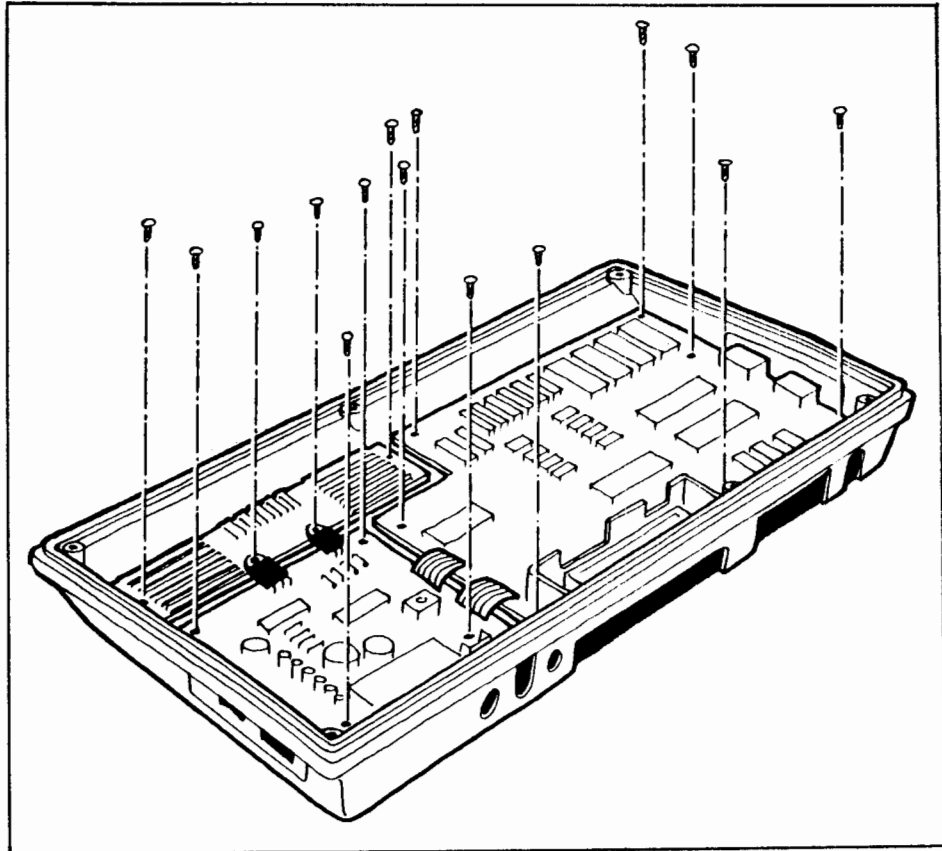


Figure 5-5 Location of Screws on the Printed Circuit Board

- (2) Carefully remove the 2 plastic spacers underneath the transistors.
- (3) Loosen the remaining 2 self-tap screws. (See figure 5-5)
- (4) Carefully slid and lift out the Heat Sink Plate.

## 5.5 REINSTALLING OF THE HEAT SINK PLATE

Reverse the procedures listed in section 5.4.

## 5.6 REMOVAL OF VIDEO AND POWER BOARD

- (1) Loosen the 2 self-tap screws on the Heat Sink Plate.
- (2) Loosen the 4 machine screws on the Video and Power Board.
- (3) Gently lift up the Heat Sink Plate.
- (4) Slide the whole unit to the left until the Power Switch is inside the Bottom Housing.
- (5) Rotate the unit 90 degree to the left and insert the Heat Sink Plate into one of the window slot when doing trouble shooting. (See figure 5-6)

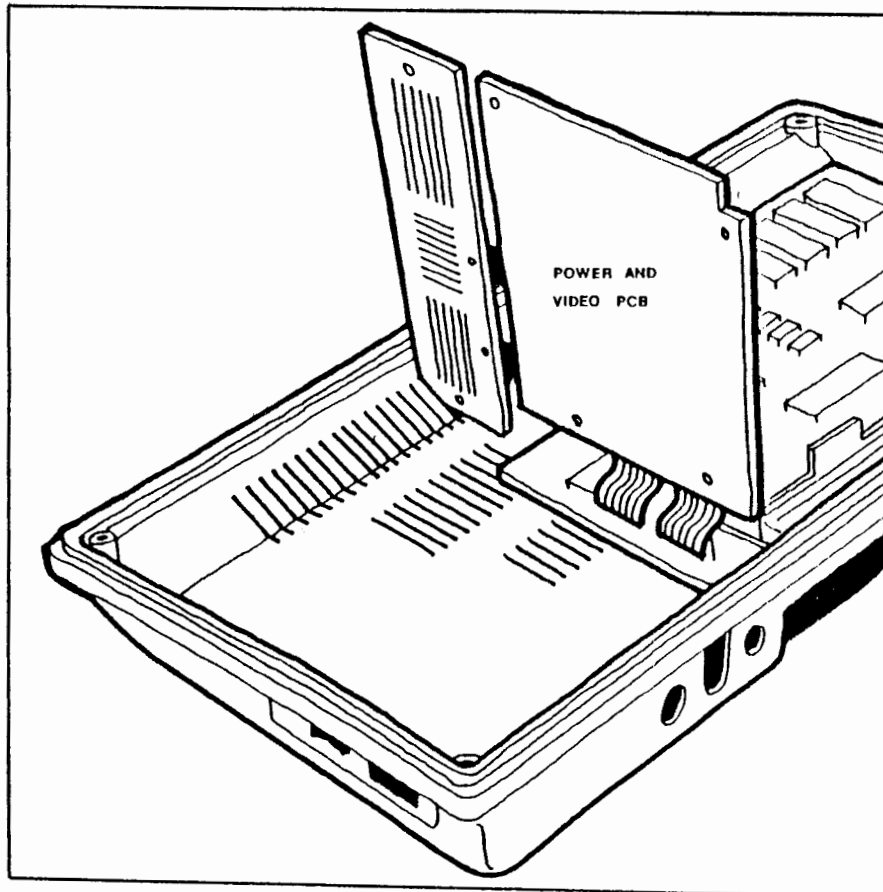


Figure 5-6 Position of the Video and Power Board for Easy Access to the IC's

## 5.7 REINSTALLATION OF THE VIDEO AND POWER BOARD

Reverse the procedures outlined in section 5.6.

## 5.8 REMOVAL OF THE LOGIC BOARD

CAUTION: Since the Logic Board and the Video and Power Board are linked together by Flex Cable, it is advised to remove the two as one unit in order to minimize possible damage.

- (1) Loosen the 2 self-tap screws on the Heat Sink Plate. (See figure 5-5)
- (2) Loosen 4 and 7 machine screws on the Video and Power Board and Logic Board respectively. (See figure 5-5)
- (3) Remove the Ground Connector (which is located behind the Game Slot) from the Logic Board.
- (4) Gently slide the Logic Board to the right until the 2 Joystick Ports are inside the housing.
- (5) Lift up the Logic Board and slide it towards the front of the Console until the clamps for the Printer Cable are inside the housing.
- (6) Slide the whole unit to the left, until the Power Switch is inside the housing.
- (7) Gently remove the whole until from the housing.

## 5.9 REINSTALLATION OF THE LOGIC BOARD

- (1) Insert the Power Switch and the Power Socket into their respective slots.
- (2) Insert the clamps for the Printer Cable into the Printer Port slot.
- (3) Slide the Joystick Ports into their respective slots.
- (4) Reinstall the Ground Connector and tighten the screw.

- (5) Tighten 7 and 4 machine screws on the Logic Board and the Video and Power Board respectively.
- (6) Tighten 2 self-tap screws on the Heat Sink Plate.

#### 5.10 REMOVAL OF KEYBOARD ASSEMBLY FROM UPPER HOUSING

NOTE: When disassembling the Keyboard be very careful with the Flex Cable.

- (1) Carefully flip the Top Housing of the Console up side down. (See Figure 5-4)
- (2) Locate and loosen 10 self-tap screws. (See figure 5-7)

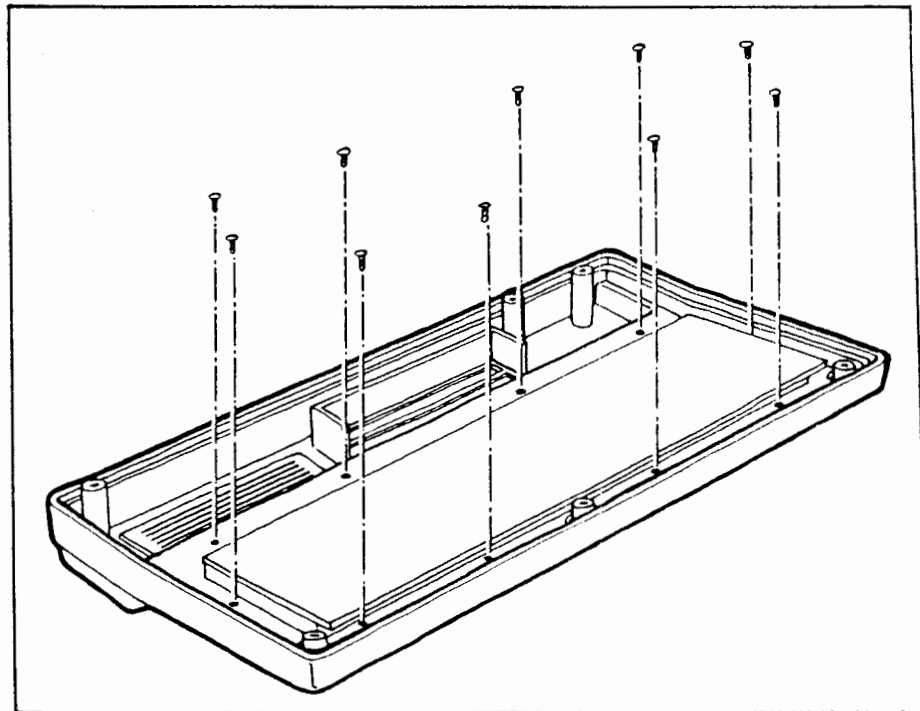


Figure 5-7 Location of the Screws on the Upper Housing

- (3) The Keyboard Assembly can now be removed from the Top Housing.

#### 5.11 REINSTALLATION OF KEYBOARD ASSEMBLY TO UPPER HOUSING

To reinstall, reverse the procedure listed in section 5-10.

## 5.12 REMOVAL OF KEYTOPS

- (1) Figure 5-8 shows the cross section of a single Keytop.

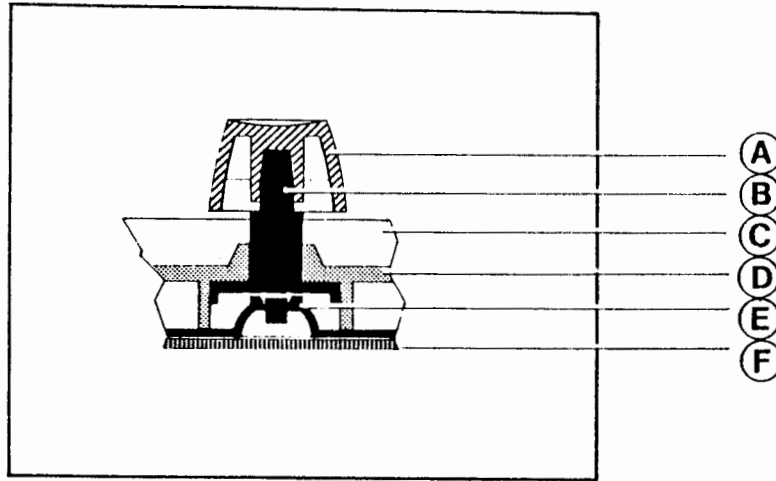


Figure 5-8 Cross Section of a Single Keytop

- |                 |                  |
|-----------------|------------------|
| A. Keytop       | D. Template      |
| B. Key Shaft    | E. Rubber Spacer |
| C. Upper Casing | F. PCB           |

- (2) Use a flat head screw driver to gently (do not force) pry off the Keytop except for the Space Bar.
- (3) To remove the Space Bar, gently pry at the middle of it until it is free from the Key Shaft.
- (4) Free the Hing Rod from the Hing Rod Holders.
- (5) Now the Space Bar can be removed. Be careful not to lose the Spring which is attached to the Key Shaft.

## 5.13 REINSTALLATION OF KEYTOPS

- (1) To install the Space Bar, position the Spring so that it is inside the Spring Holder.
- (2) Snap the Hing Rod into the Hing Rod Holders.



- (3) Push the Key Shaft down, and place the Space Bar over the Key Shaft.
- (4) Check and make sure that the latches (which are white in colour) fall properly into their openings. If it does not, then free the Hing Rod from the Hing Rod Holders and turned the Spacer Bar around and repeat the above procedures.
- (5) Align the Key Shaft with the Space Bar and push down at the middle of the Space Bar.
- (6) To reinstall the Keytop, align it with the Key Shaft and push down on it.
- (7) Repeat the above step until all the Keytops has been reinstalled.

#### 5.14 DISASSEMBLE OF KEYBOARD ASSEMBLY

- (1) Figure 5-9 shows the 6 levels of the Keyboard Assembly.

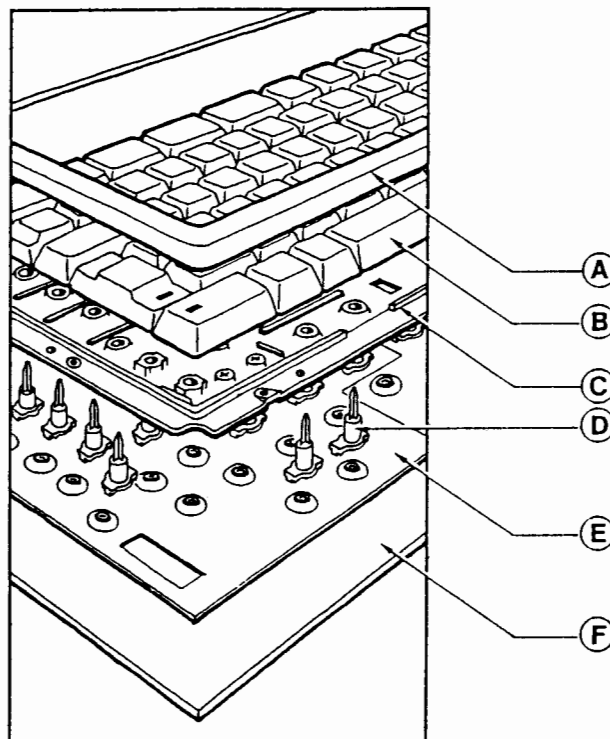


Figure 5-9 6 Levels of the Keyboard Assembly

- |                  |   |
|------------------|---|
| A. Upper Housing | D. Key Shaft                            |
| B. Key Tops      | E. Rubber Spacer with Silicon Conductor |
| C. Template      | F. Keyboard PCB                         |

- (2) To disassemble, locate and loosen 25 self-tap screws on the Keyboard PCB. (See Figure 5-10)

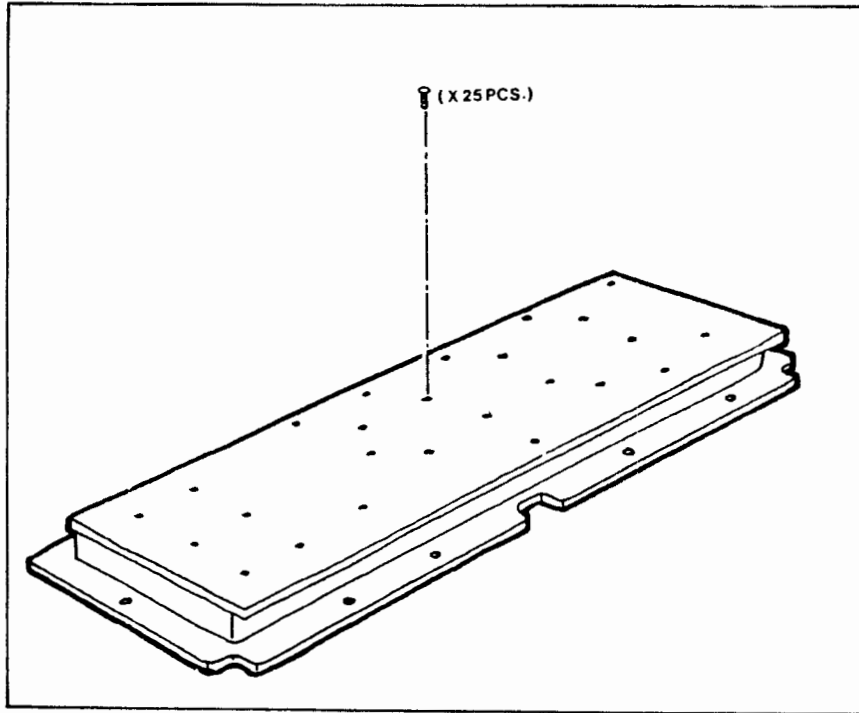


Figure 5-10 Location of Screws on Keyboard PCB

- (3) Remove the Keyboard PCB.
- (4) Remove the Rubber Spacer.
- (5) Turn the unit right side up.
- (6) Follow sections 5.12 and 5.13 for removal and reinstallation of Keytops and Space Bar.

#### 5.15 REASSEMBLE OF KEYBOARD ASSEMBLY

Reverse the procedure listed in section 5.14 for reassembling of Keyboard Assembly.

## CHAPTER 6

### TROUBLE SHOOTING

This chapter is intended to give an average technicians clues in repairing SVI-728 computer system. A quick reference table is given in symptom checklist for fast repairing needs. A detail step-by-step diagnostic flowchart is given in 6.3 for full system check. A test cartridge is needed for diagnosis to perform.

#### 6.1 SWAP OUT PROCEDURE

At many places in the diagnostic flowcharts or symptom checklist, a chip or a number of chips has to "swap-out" in particular order. The "swap-out" instruction means that you should replace the indicated components (one at a time) with a known good component of the same type. The SVI-728 should then be tested with the new, known-good component in place to see whether the "swap-out" solved the problem being checked. If the swap-out did not fix the problem, the known-good component should be removed, and the original component reinserted. In this way, you avoid needlessly replacing good components.

#### CAUTION:

Extreme care should be taken when handling the integrated circuit chips. They are all very sensitive to static electricity and can easily be damaged by careless handling. Always keep the chips in their plastic carrier tubes or on conductive foam when not handling them. Make certain you are well grounded when handling the chips.

In disassembly procedures, refer to Chapter 5 for reference.

#### 6.2 SYMPTOM CHECKLIST

The symptom checklist is designed to give a rapid diagnosis for problems. The checklist preferred to be used by experienced service technician and for less experienced personnel, a step-by-step diagnosis flowchart is recommended.

Each symptom is accompanied by suitable replacements and remarks for references.

<u>Symptom</u>	<u>Replacement</u>	<u>Remarks</u>
(1) No Power	Bad Adaptor; ZD1 (5.1V), ZD2, ZD3 ( 12V zener diode) IC3 (LM741CN) Microswitch (cartridge slot)	Refer to Power Diagnostic flowchart for more detail analysis
(2) Poor Colour	Check +5V, +12V power supply of VDP first tune VR2, VR3	Refer to symptom (1) for power Adjust colour quality
(3) No Colour	XT1, 4.43MHz crystal IC2 74LS02	
(4) Cassette		
Save Problem	IC41 (8255A)	Make sure problem is independent of tape and cassette first
CLOAD Problem	IC40 (PSG AY-3-8910)	
(5) No Printer	IC55, IC56 (74LS175) IC57 (74LS125)	
(6) Keyboard Error	Check cable connection between keyboard PCB and Logic Board. Any Open or Short happens.	Mostly the case. Take caution in handling cable connection.
	IC41 (8255A) IC42 (7445)	
(7) Boot Cartridge Fail	IC21 (CPU Z80A) IC34 (74LS139) Mask ROM	
(8) Joystick Fail	IC44, IC45 (74LS157)	

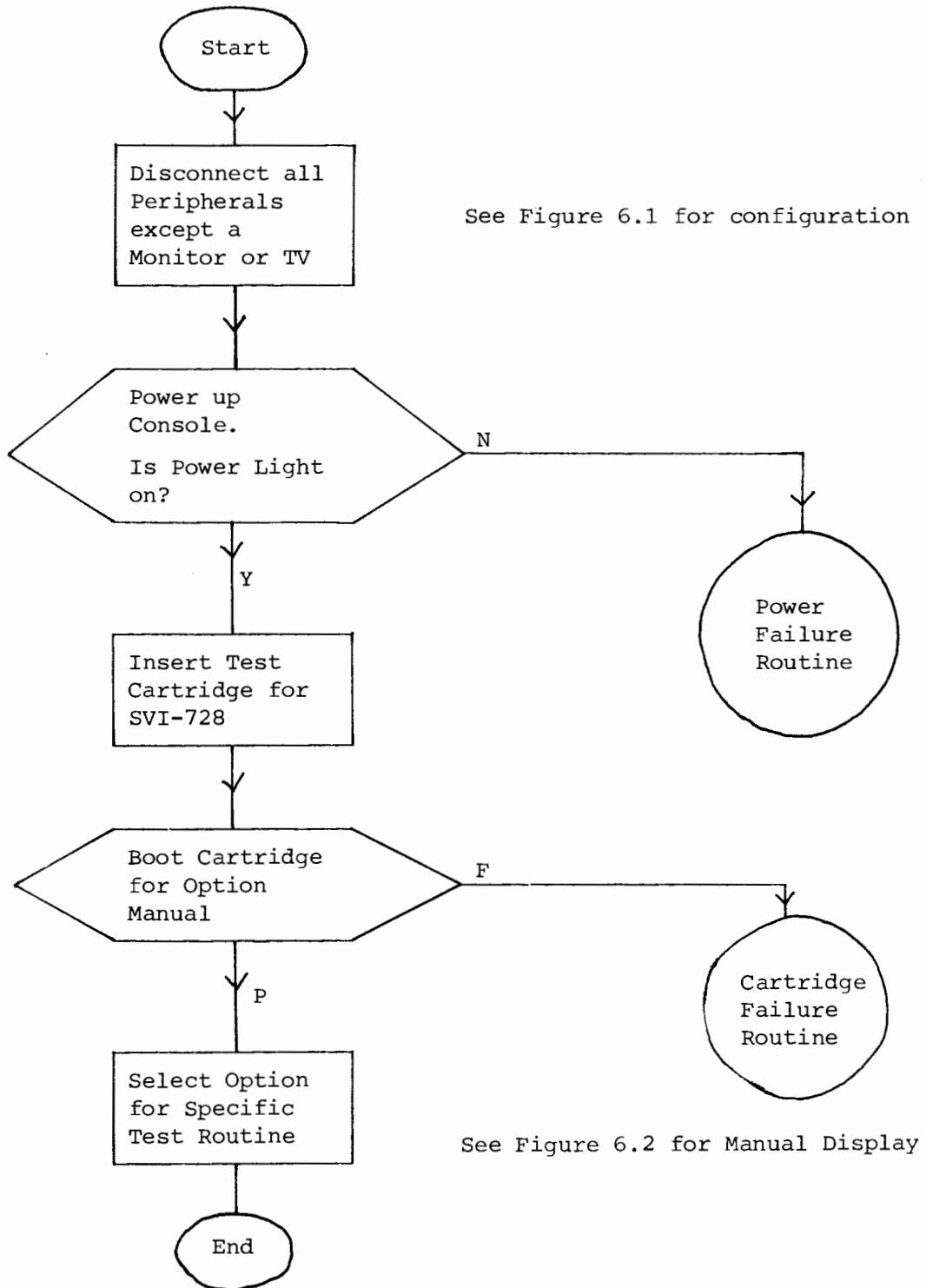
### 6.3 DIAGNOSTIC FLOWCHART FOR SVI-728 COMPUTER

The Diagnostic Flowchart is intended to be easy to use and the primary aid when troubleshooting the SVI-728 computer console. This flowchart is meant to be used with a test cartridge. Follow the flows in the order presented. A simple 'Yes' or 'No' decision in the flowchart would direct user to faulty area.

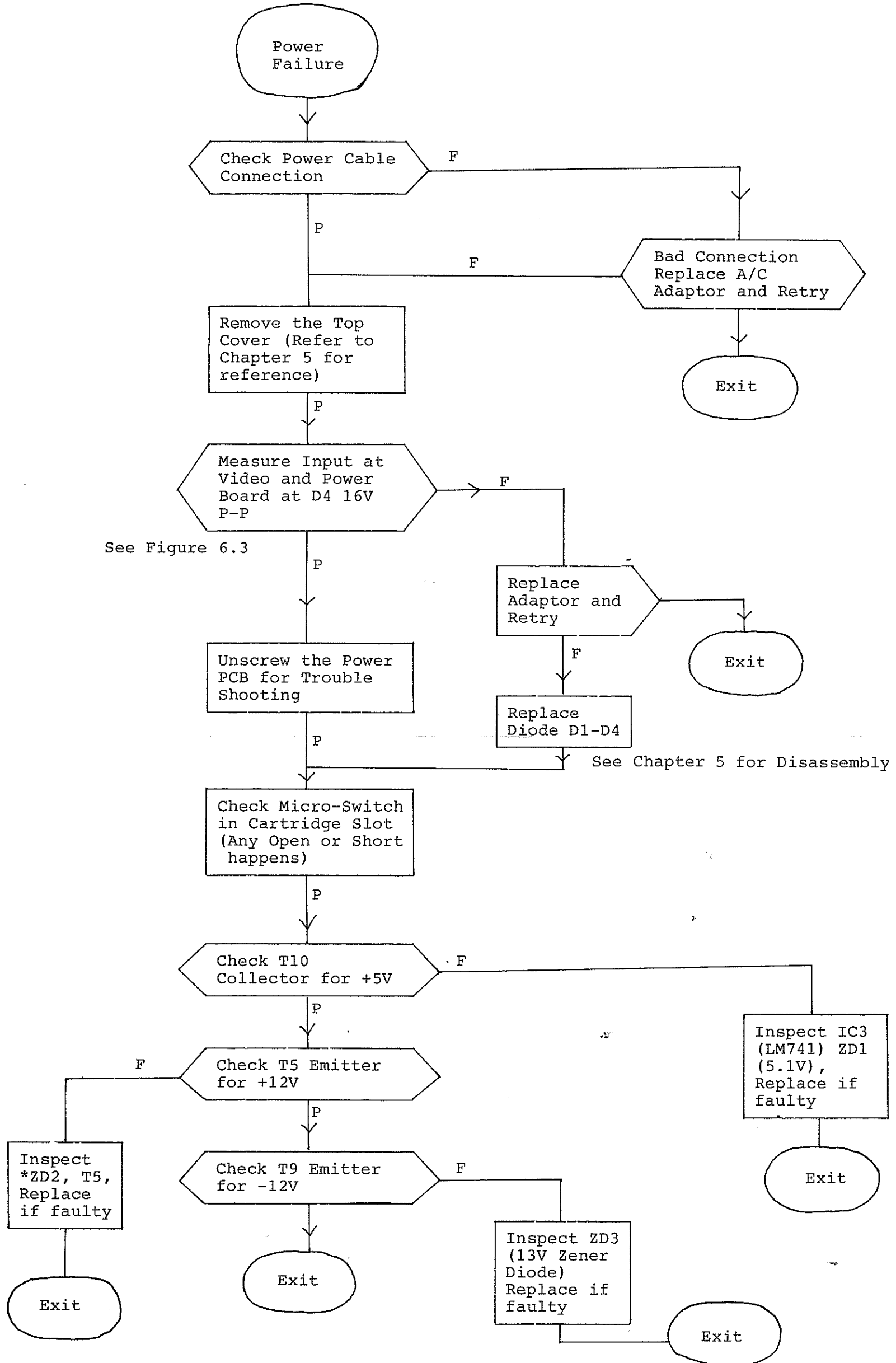
In case the Diagnostic Flowchart does not solve your problem, please consult experience repairman or your local agency for further advice and evaluation.

### 6.3.1 BASIC OPERATION

- \* F - Test Fails or Unexpected Display
- \* P - Test Pass
- \* Y - Yes to Question Asked
- \* N - No to Question Asked

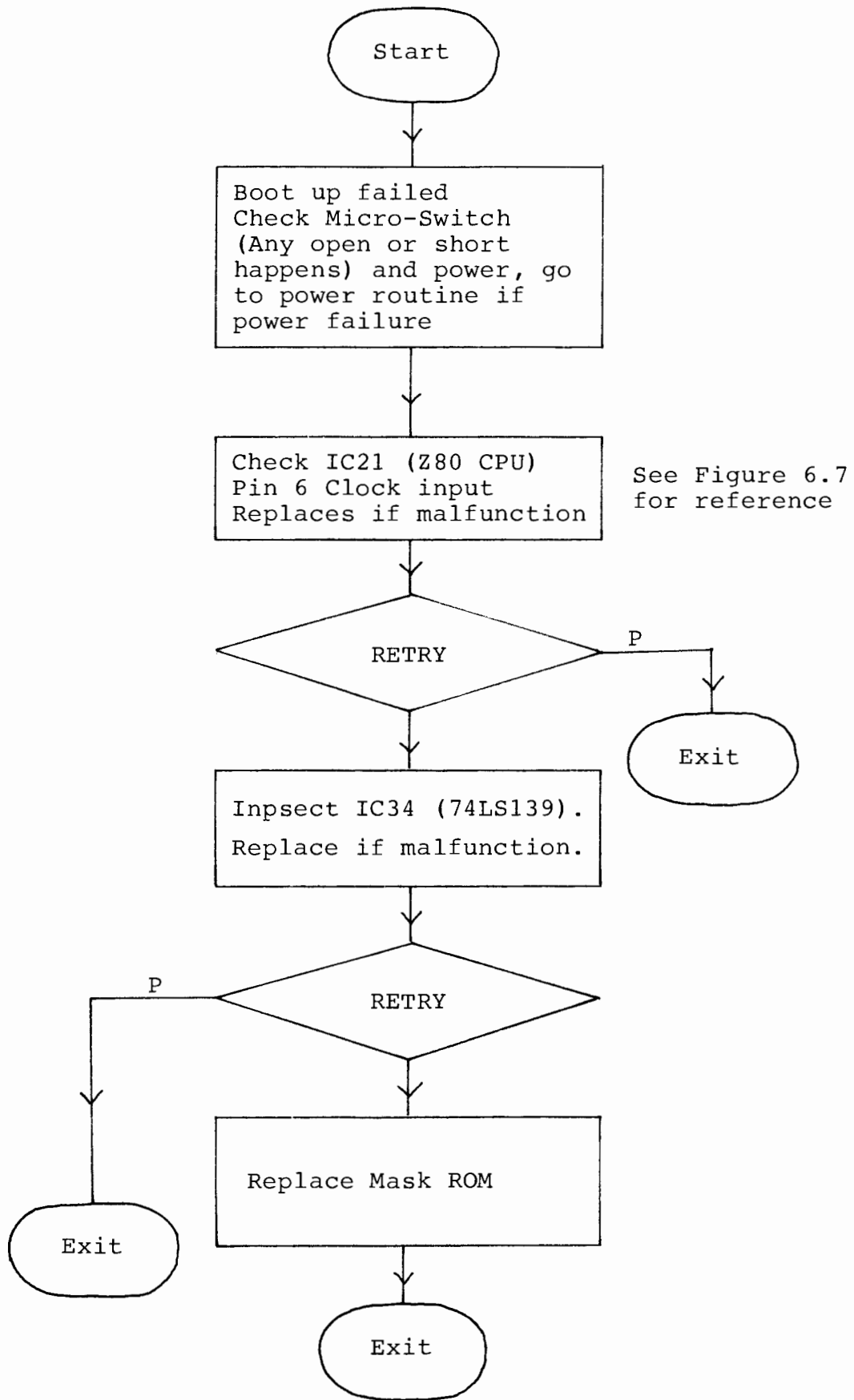


6.3.2 POWER FAILURE



\* 13V Zener Diode was used to replace IA78L12, diode and lu capacity in early version.

6.3.3 CARTRIDGE BOOT UP TEST

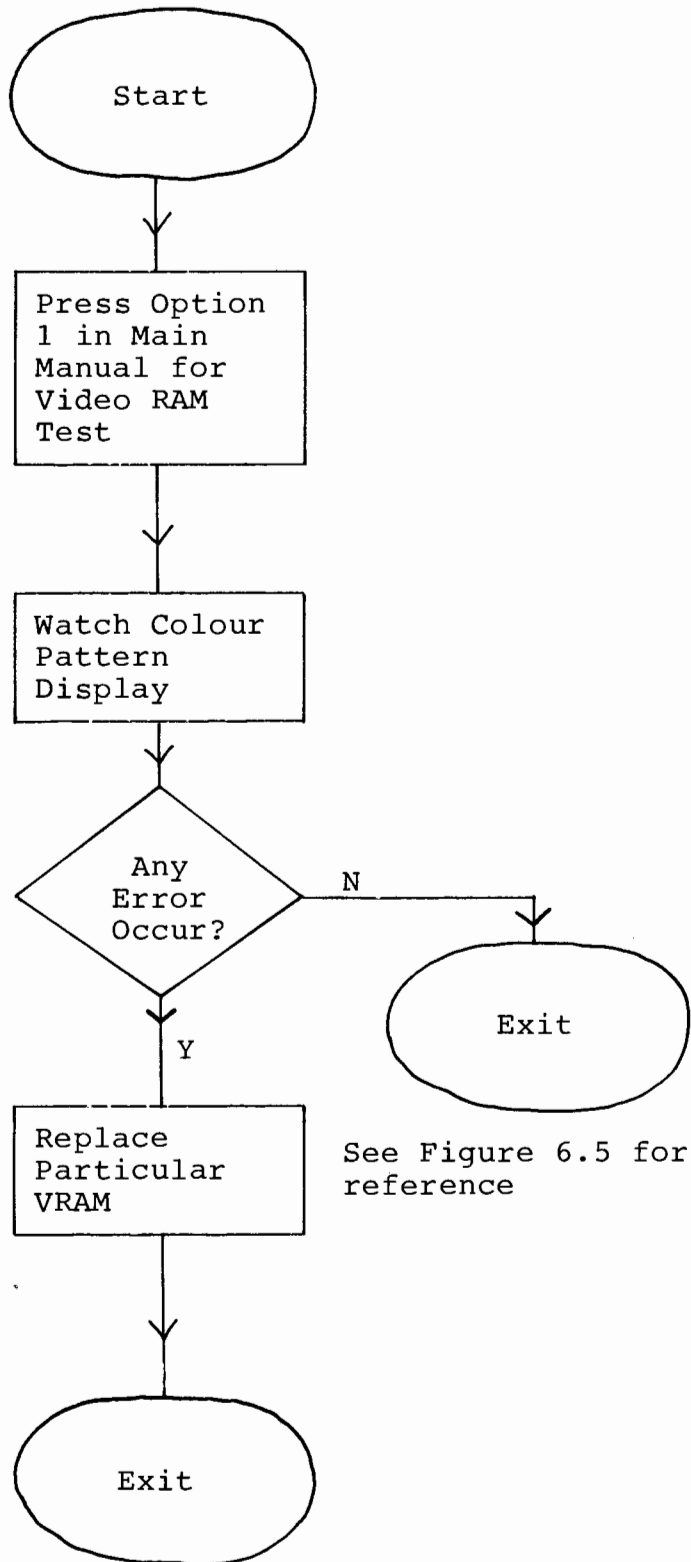




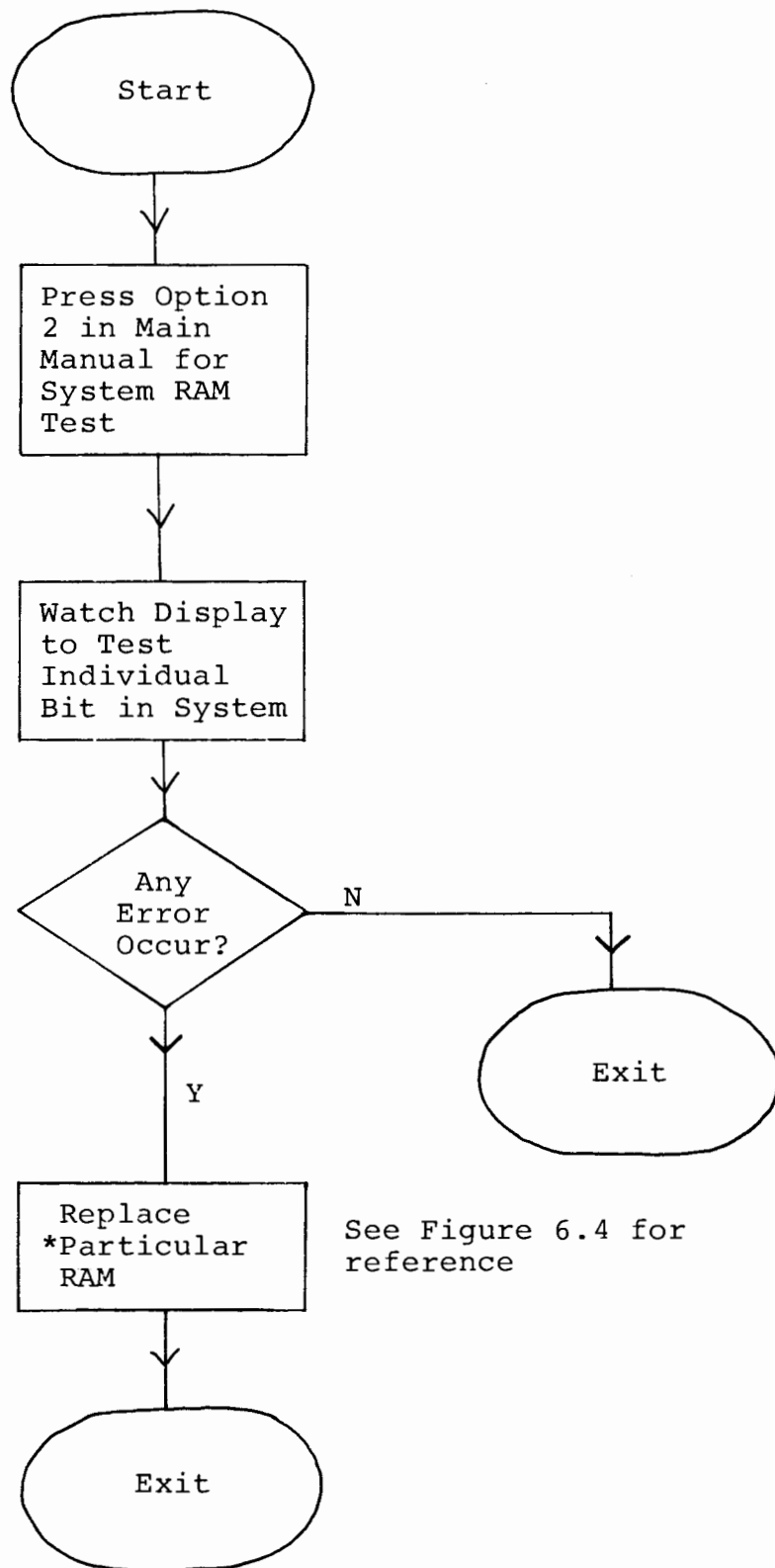
#### 6.3.4 SYSTEM ROM TEST

Checksum of system ROM is tested under test cartridge programme (option 0 in Main Manual). If any error occurs, replaces system ROM (4 x 2764 in EPROM version).

6.3.5 VIDEO RAM TEST

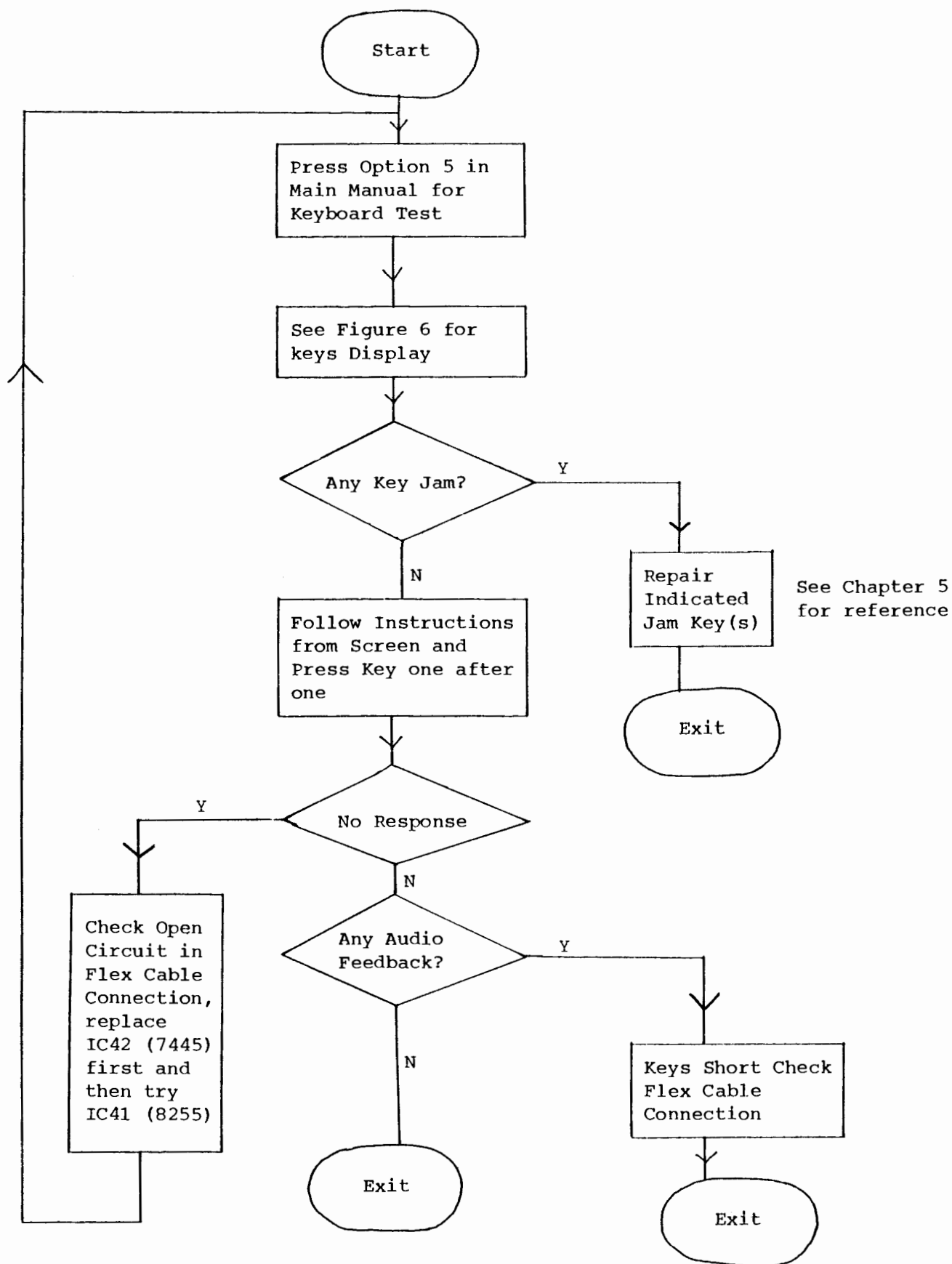


6.3.6 SYSTEM RAM TEST

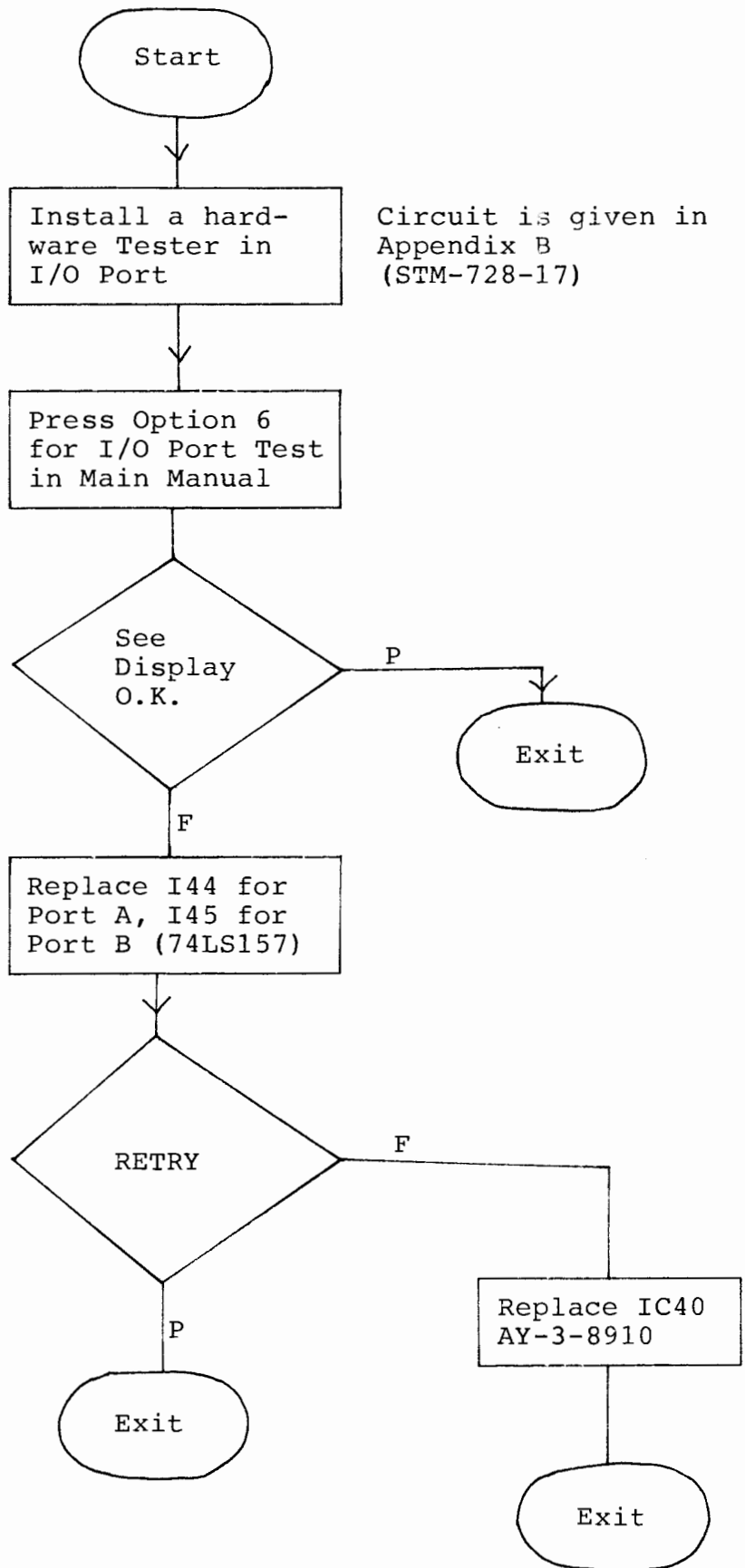


\* In some case (more than 1 error bit), all RAMs (IC12 - IC19) need to be replaced.

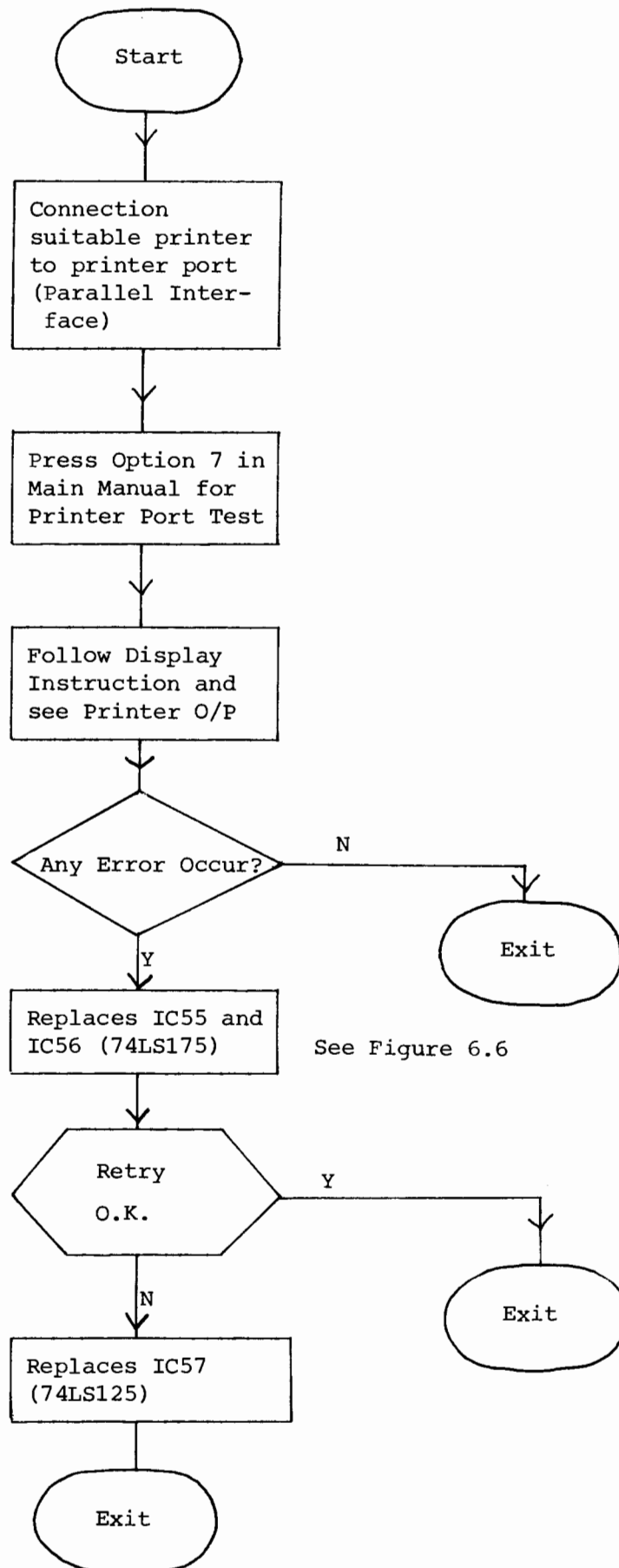
6.3.7 KEYBOARD TEST



6.3.8 I/O (JOYSTICK) PORT TEST



6.3.9 PRINTER PORT TEST



6.3.10 CASSETTE PORT TEST

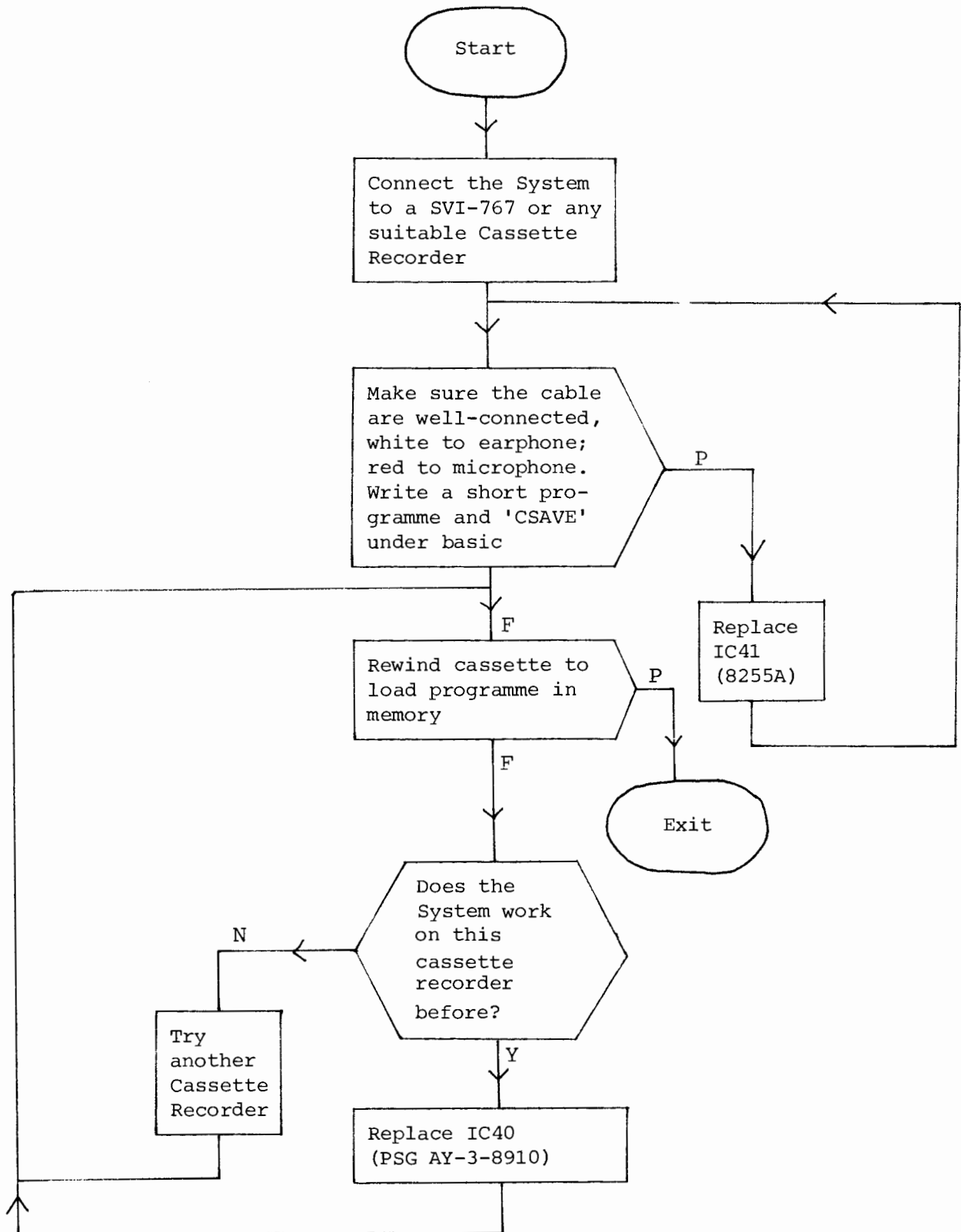




Figure 6.1 Test System Set Up

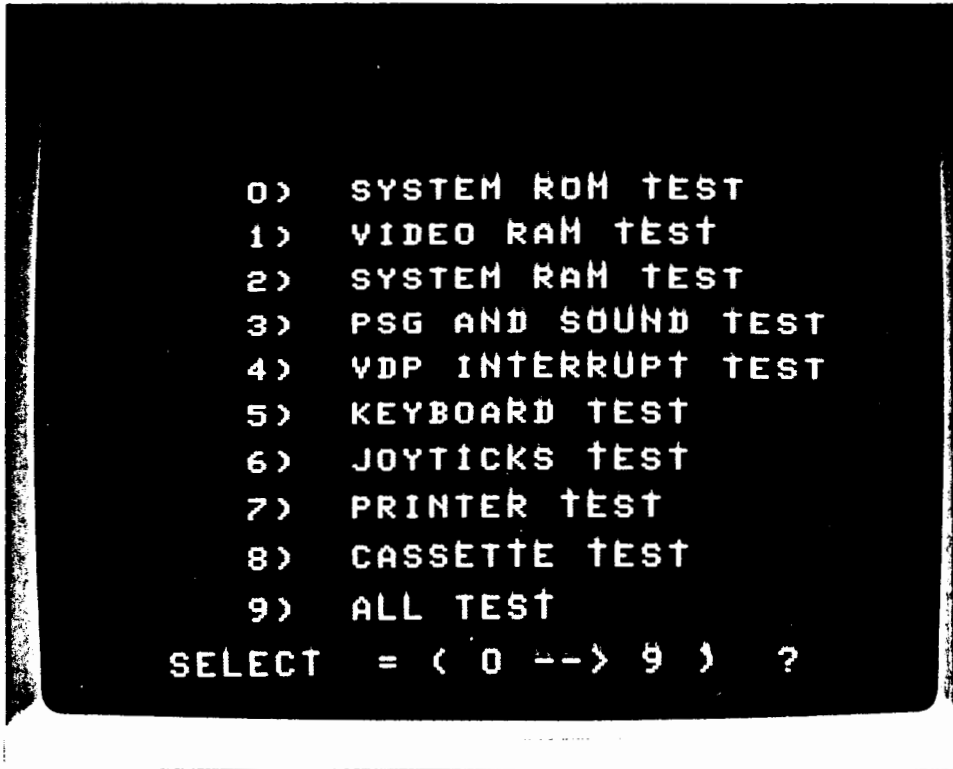


Figure 6.2 SVI-728 Test Cartridge Main Manual Display



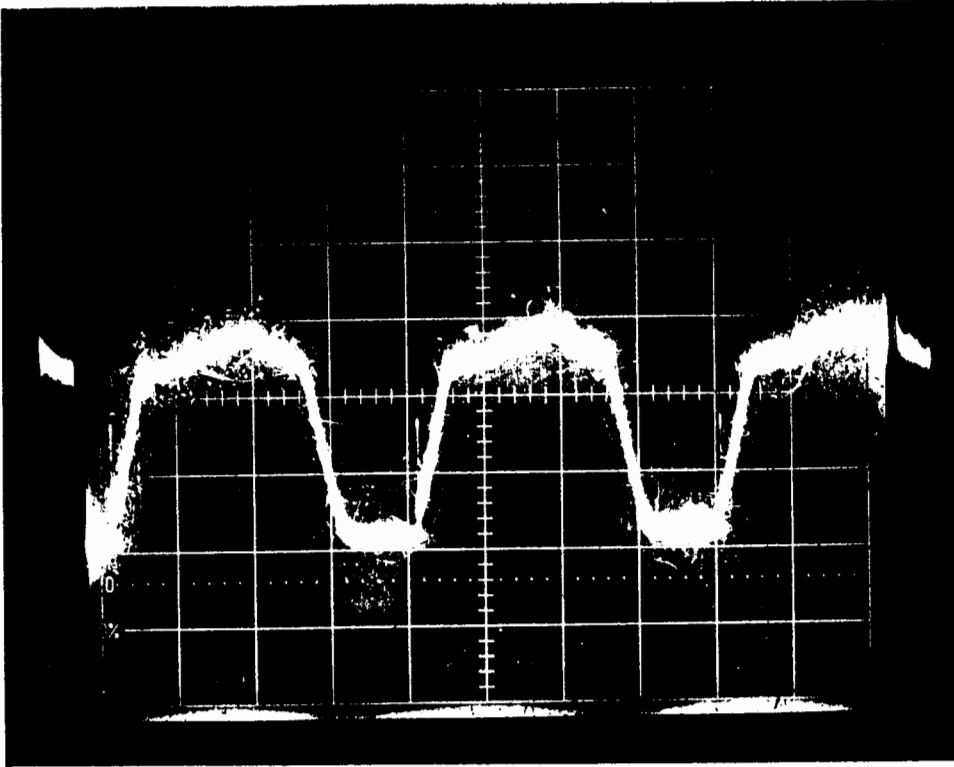


Figure 6.3 Rectified O/P at D4 (Power and Video Board) 5V/Div. 5ms/Div.

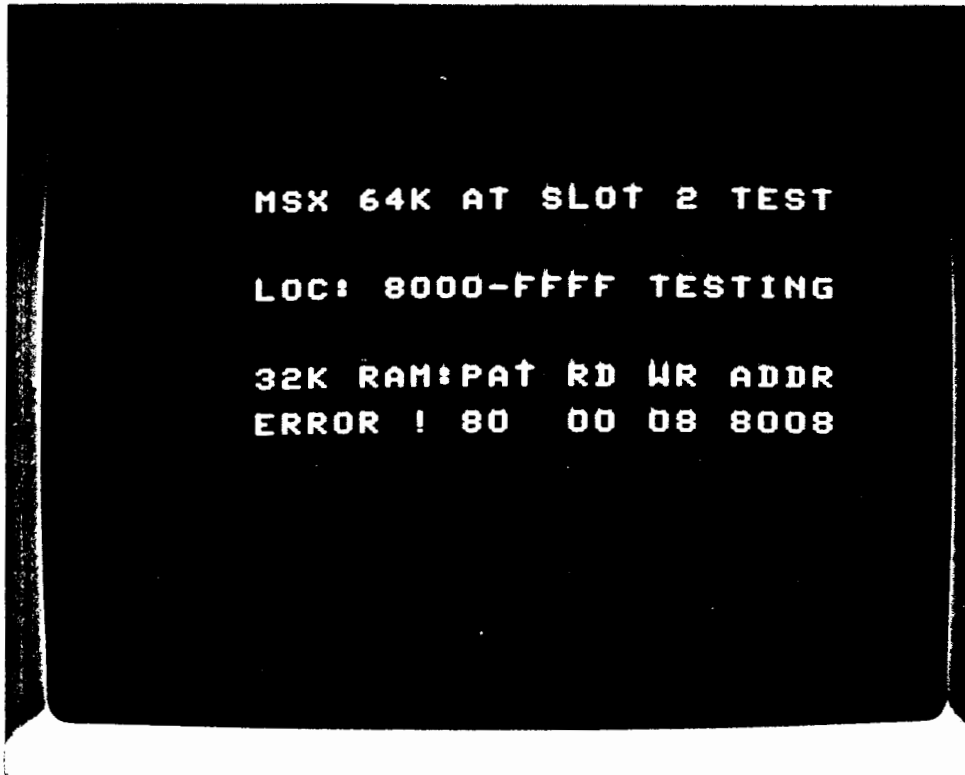


Figure 6.4 RAM Error Display

Error Location:

Bit No.	7	6	5	4	3	2	1	0	
RD Content →	0	0	0	0	0	0	0	0	(00)
WR Content →	0	0	0	0	1	0	0	0	(08)
					↑ Error Bit				
IC number to be replaced if error occurs	IC17	IC16	IC15	IC19	IC18	IC12	IC13	IC14	

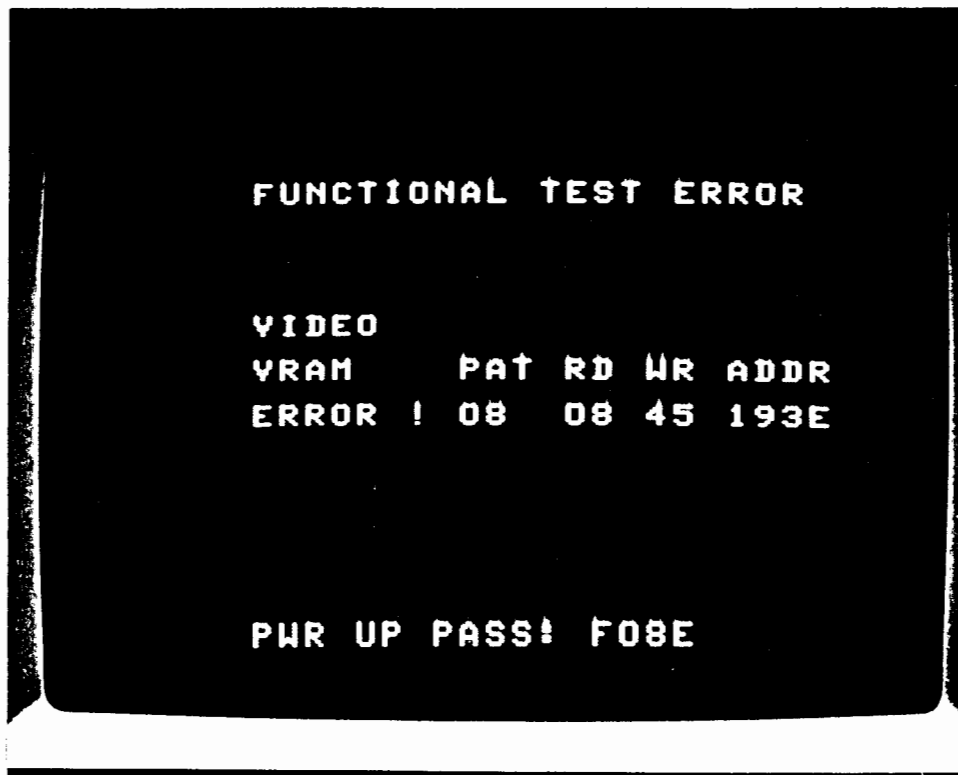


Figure 6.5 VRAM Error Display  
Replace IC18, IC19 (4416 VRAM)

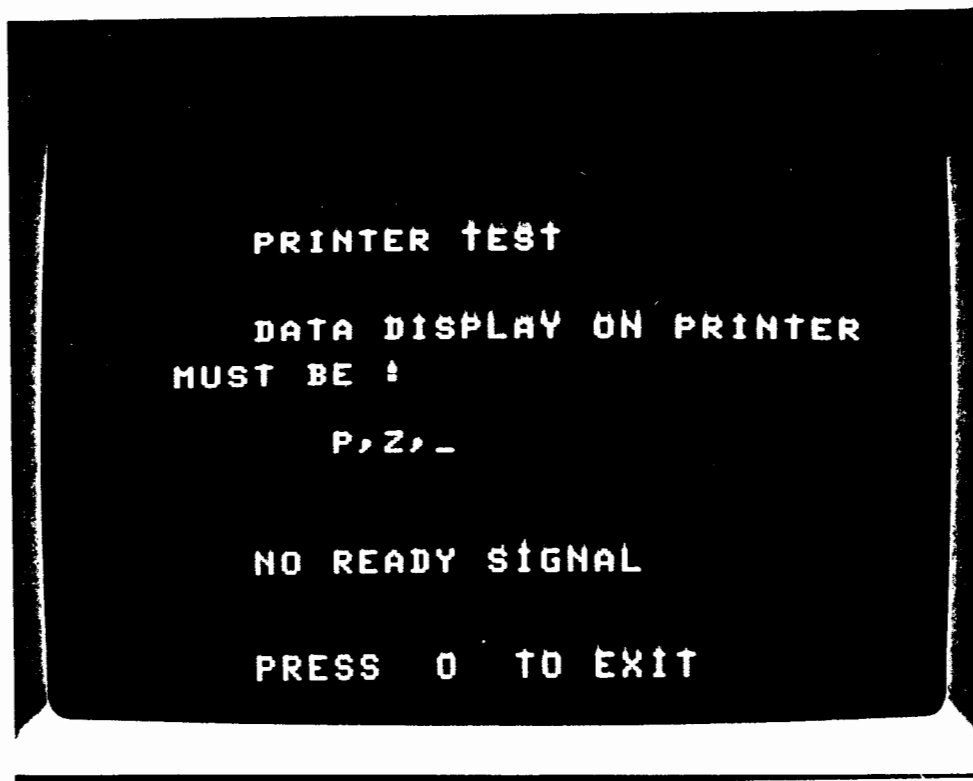


Figure 6.6 Printer Test Error Display (No Printer)

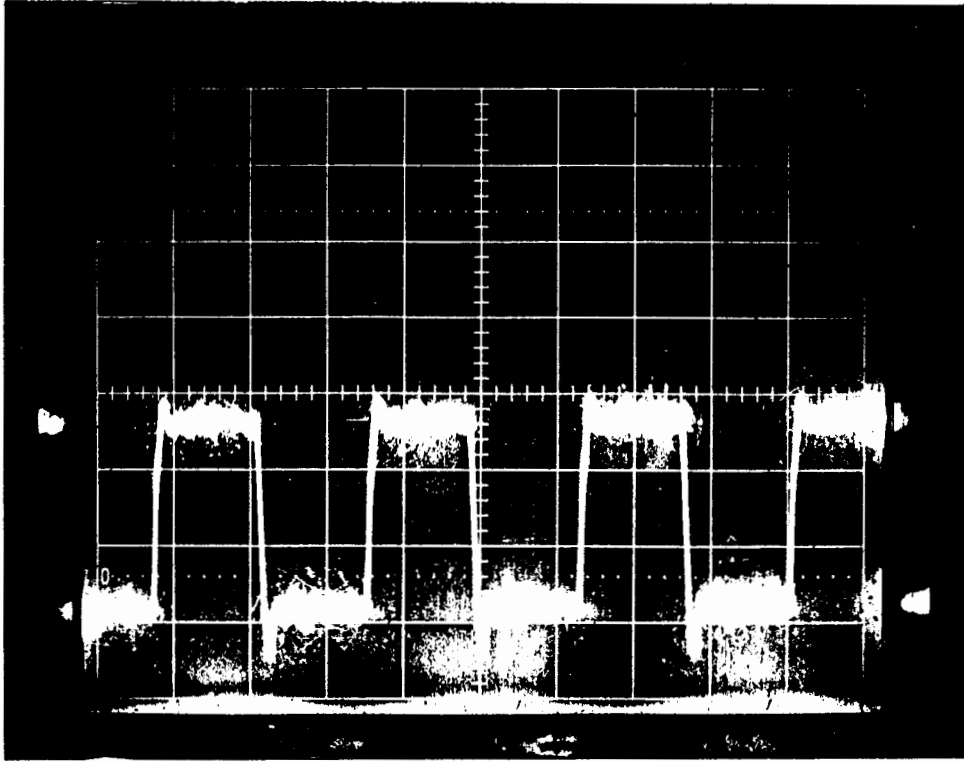


Figure 6.7 Clock Waveform of CPU (Z80) Div. 6  
2V/Div. 0.1us/Div. ( 3.58 MHz)

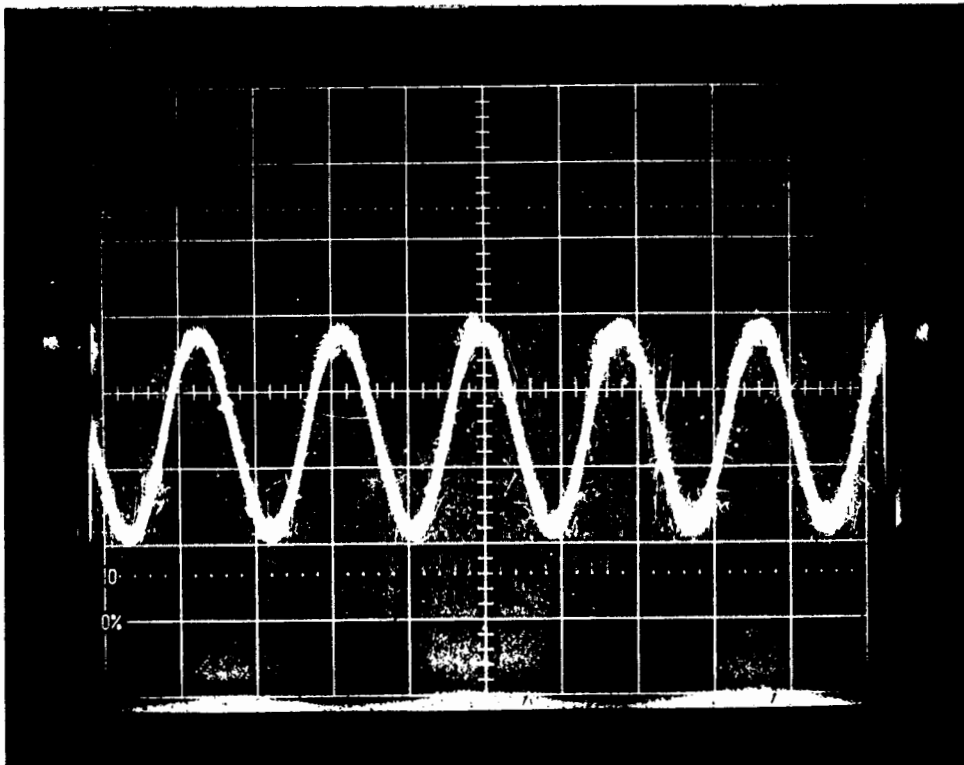


Figure 6.8 Clock Waveform of VDP (9918)  
2V/Div. 0.05us/Div. (10.74 MHz)

## 6.4 DIAGNOSTIC FLOWCHART FOR SVI-707 DISK DRIVE

### MAINTENANCE

#### MAINTENANCE AND MAINTENANCE EQUIPMENT

This section describes the necessary maintenance equipment, troubleshooting and maintenance procedures.

#### Alignment Diskette

The alignment diskette is used for verifying and adjusting the Shugart 455/465. Two alignment diskettes are available. The 455/465 has two read/write heads and requires written information on both surfaces. The Shugart 128 (48 tpi) alignment diskette should be used when performing service checks on the 455. The 465 requires the Shugart 126 (96 tpi) alignment diskette.

The following adjustments and checks can be made using the Shugart 128/126 alignment diskettes.

	Shugart 455 - 128	Shugart 465 - 126
a. Read/Write Head Radial Alignment	TRK 16	TRK 64
b. Index Photo Detector Alignment	Set at TRK 38 Verified at TRK 01	Set at TRK 76 Verified at TRK 02
c. Track 00 Head Position	TRK 00	TRK 00
d. Azimuth Angle (not field adjustable)	TRK 33	TRK 64
e. 125 k Hz Signal Recorded to Check Head Position on Inside Track	TRK 34	TRK 79

Caution should be used not to destroy prerecorded alignment tracks. The write protect tab must be installed to prevent accidental writing on the alignment diskette. If the write protect option is used, remove the write protect tab.

#### Exerciser PCB

The exerciser PCB can be used in a stand alone mode, built into a test station, or used in a test for field service.

The exerciser will enable the user to make all adjustments and checkouts required on the 455/465 minidiskette drive. It has no intelligent data handling capabilities but can write a 2F 125 k Hz signal which is the recording frequency used for amplitude checks on the 455/465 drive. The exerciser can start and stop the drive motor, and enable read in the 455/465 to allow checking for proper read back signals.

#### Special Tools

The following special tools are available for performing maintenance on the 455/465.

Description	Part Number
Shugart 128 Alignment Diskette	54573
Shugart 126 Alignment Diskette	54382
Exerciser PCB	54157
Head Cable Extender	54578
Phillips Screw Drivers	Medium and Small
Oscilloscope	Textronix 465 or equivalent

## DIAGNOSTIC TECHNIQUES

### Introduction

Incorrect operating procedures, faulty programming, damaged diskettes, and "soft errors" created by airborne contaminants, random electrical noise, and other external causes can produce error falsely attributed to drive failure or misadjustment. Unless visual inspection of the drive discloses an obvious misalignment or broken part, attempt to repeat the fault with the original diskette, then attempt to duplicate the fault on a second diskette.

### "Soft Error" Detection and Correction

Soft errors are usually caused by:

- a. Airborne contaminants that pass between read/write heads and disk. Usually these contaminants can be removed by cartridge self-cleaning wiper.
- b. Random electrical noise that usually lasts for a few microseconds.
- c. Small defects in written data and/or track not detected during write operation may cause soft errors during read.
- d. Improper grounding of power supply, drive, and/or host system. Refer to paragraph 3.2 for proper grounding requirements.
- e. Improper motor speed.

The following procedures are recommended to recover from the above mentioned soft errors:

- a. Reread track ten times or until such time as data is recovered.
- b. If data is not recovered after using step "a", access head to adjacent track in same direction previously moved, then return to desired track.
- c. Repeat step "a"
- d. If data is not recovered, error is not recoverable.

### Write Error

If an error occurs during a write operation, it will be detected on the next revolution by doing a read operation, commonly called a "write check." To correct the error, another write and check operation must be done. If the write operation is not successful after ten attempts have been made, a read operation should be attempted on another track to determine if the media or the drive is failing. If the error persists, the diskette should be replaced and the above procedure repeated. If the failure still exists, consider the drive defective. If the failure disappears, consider the original diskette defective and discard it.

### Read Error

Most errors that occur will be "soft errors." In these cases, performing an error recovery procedure will recover the data.

### Seek Error

- a. Stepper malfunction.
- b. Carriage binds.
- c. To recover from a seek error, recalibrate to track 00 and perform another seek to the original track or do a read ID to find on which track the head is located.

## TROUBLESHOOTING

Figures 1 through 5 provide troubleshooting procedures for the 455/465.

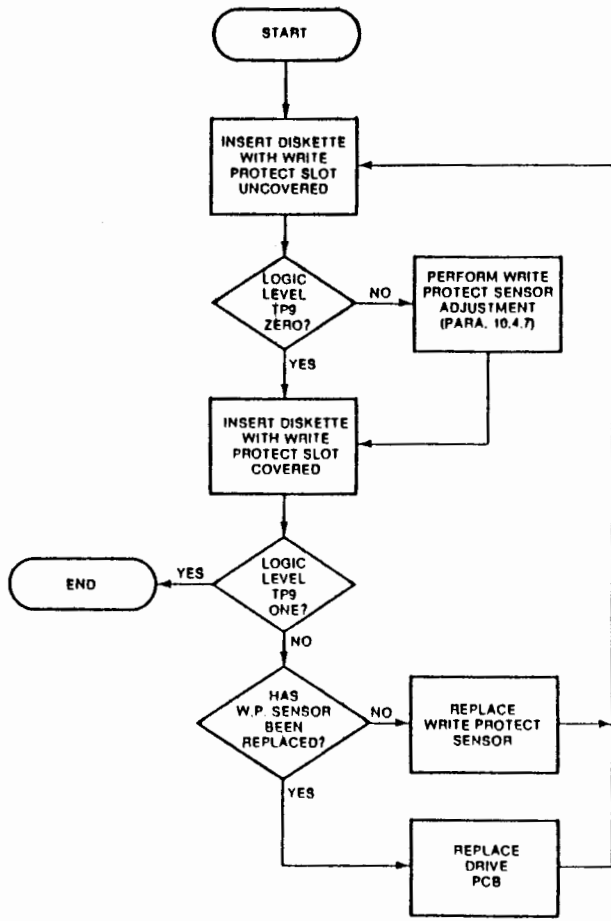
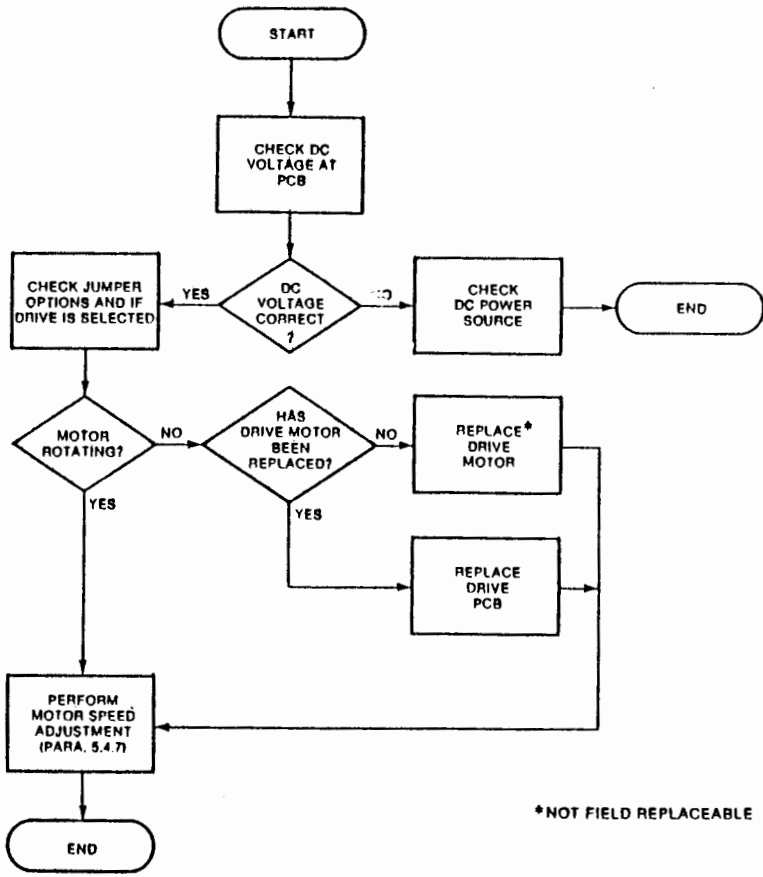


FIGURE 1. WRITE PROTECT INOPERATIVE



\*NOT FIELD REPLACEABLE

FIGURE 2. DISKETTE NOT ROTATING

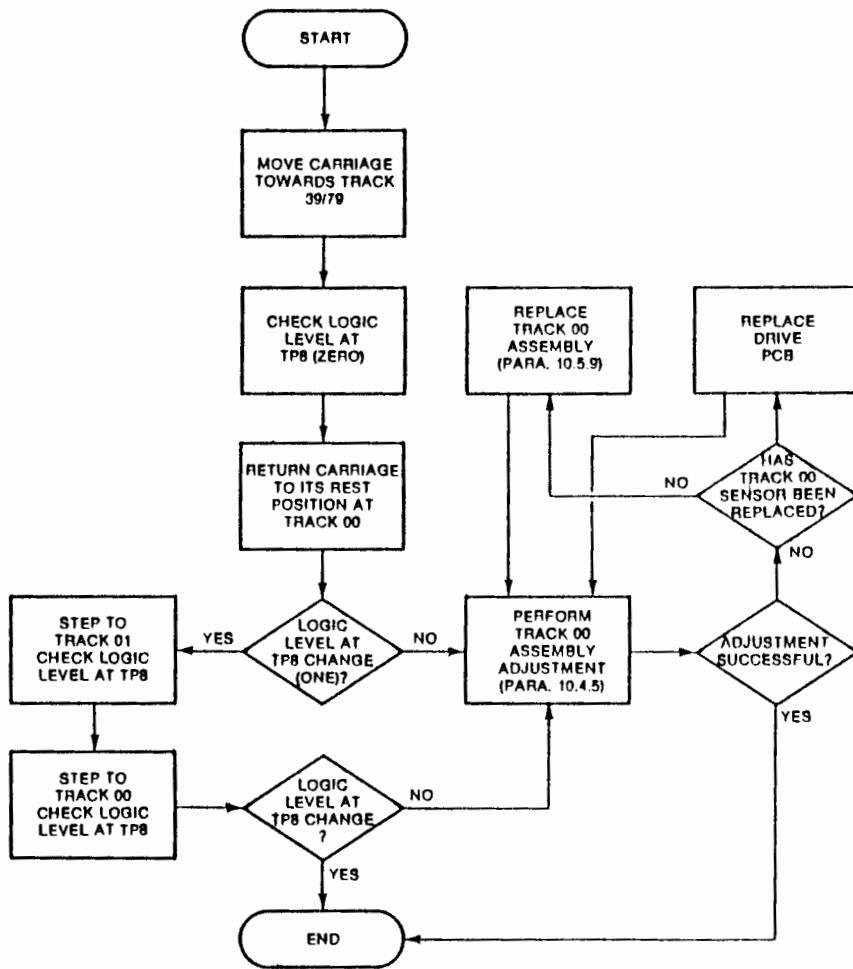


FIGURE 3. MLC 2 PCB COMPONENT LOCATIONS



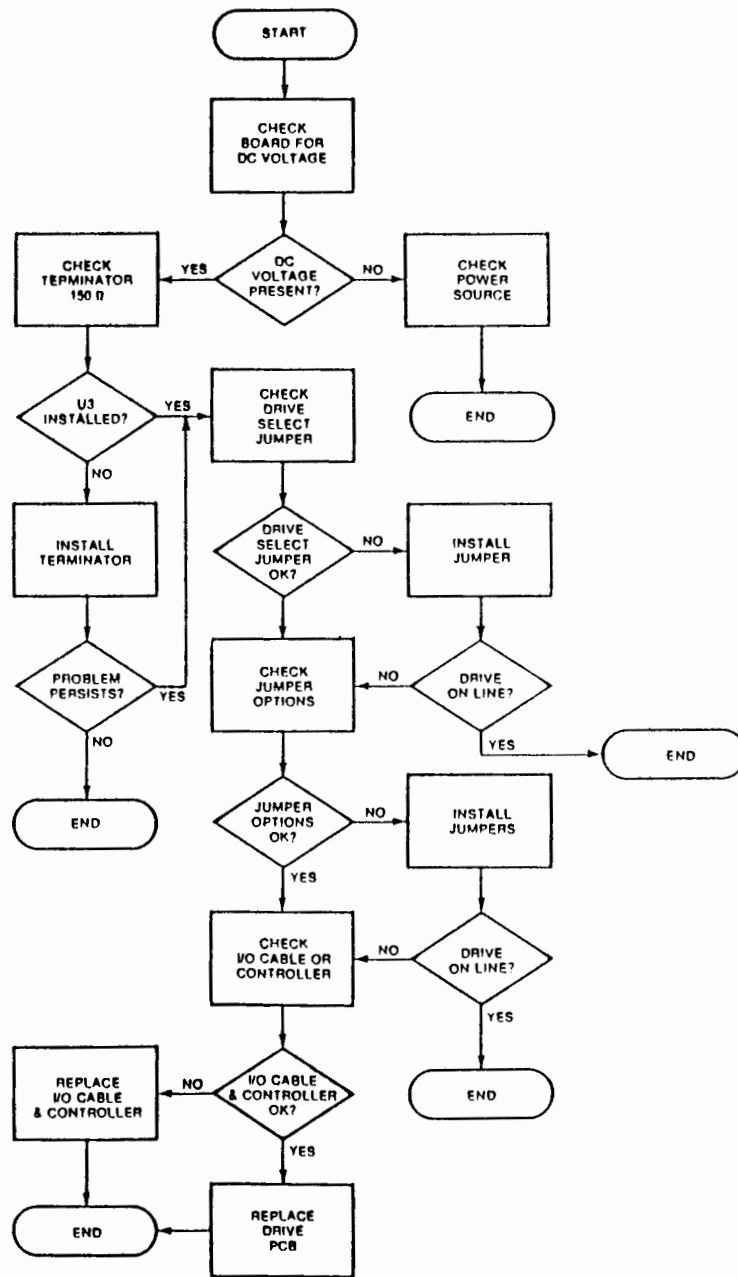


FIGURE 4. DRIVE NOT COMING ON LINE

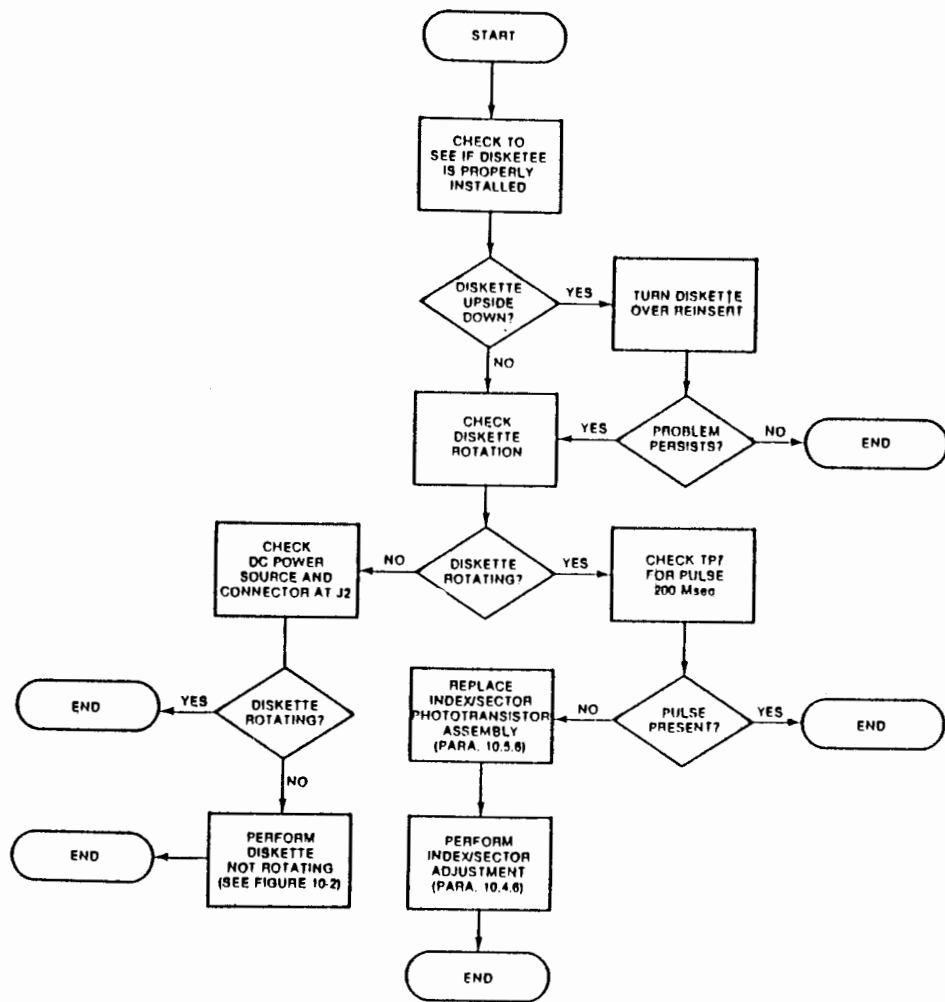


FIGURE 5. INDEX PULSE INOPERATIVE

## ADJUSTMENTS

### Head Radial Alignment

#### NOTE

The 465 read/write head assembly is aligned at factory and adjustment of head to head alignment is not field adjustable.

- a. Insert alignment diskette (Shugart 128 for 455 and Shugart 126 for 465)

#### NOTE

Alignment diskette should be at room conditions for at least 24 hours before alignment checks.

- b. Select drive and step head(s) to track 16 (455) or track 64 (465).
- c. Sync oscilloscope external negative on TP7 (-INDEX). Set time base to 20 msec per division. This will display over one revolution.
- d. Connect one probe to TP1 and other to TP2. Ground probes to PCB. Set inputs to ac, ADD, and invert one channel. Set vertical deflection to 50 mV/division.
- e. Amplitude of two lobes must be within 70 percent of each other. If lobes do not fall within specification, continue with procedure (see figure 10-6).

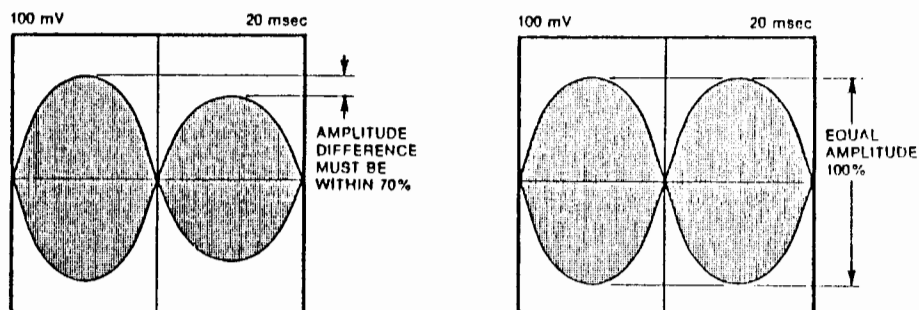


FIGURE 8. ALIGNMENT LOBES

- f. Loosen two mounting screws, which hold stepper motor to base casting (see figure 7).
- g. Adjust stepper motor.
- h. When lobes are of equal amplitude, tighten motor plate mounting screws (see figure 7).
- i. Check adjustment by stepping off track and returning. Check in both direction and readjust as required.
- j. Whenever head radial alignment has been adjusted, track 00 detector must be checked.

#### CAUTION

When tightening mounting screws, pressure must be applied to the rear of the stepper motor through the rectangular hole in the side of the casting to keep the motor bracket against the registering surfaces of the casting. Failure to do this will angle the band positioner causing track-to-track problems.

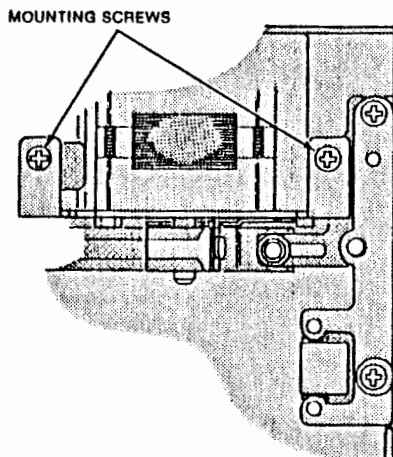


FIGURE 7. STEPPER MOTOR MOUNTING SCREWS

**Read/Write Head(s) Azimuth Check**

The azimuth is not field adjustable. If after performing this check the waveform on the oscilloscope is not within  $\pm 18$  minutes (455) or  $\pm 21$  minutes (465), the drive must be returned for replacement of read/write head assembly.

- a. Install alignment diskette (Shugart 128 for 455 and Shugart 126 for 465).
- b. Select drive and step to track 33 (455) or track 64 (465).
- c. Sync oscilloscope external negative on TP7, set time base to 0.5 msec per division.
- d. Connect one probe to TP1 and other to TP2. Invert one channel and ground probes to PCB. Set inputs to ac, ADD, and set vertical deflection to 50 mV per division.
- e. Compare waveform to figure 10-8. If not within range shown, replace read/write head assembly.

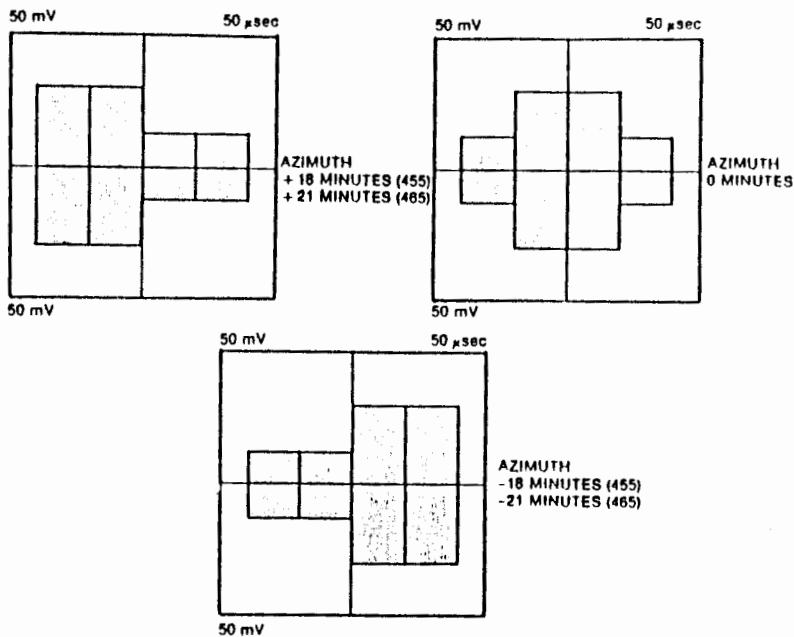


FIGURE 8. AZIMUTH CHECK

### Head Amplitude Check

These checks are only valid when writing and reading back as described below. Ensure the diskette used for this check is not "worn" or otherwise shows evidence of damage on either side.

- a. Install good media.
- b. Start motor.
- c. Select drive and step to track 39 (455) or track 79 (465).
- d. Sync oscilloscope external on TP7 (+Index); connect one probe to TP2 and TP1 on drive PCB. Ground probes to PCB, ADD, and invert one input. Set volts per division to 50 mV and time base to 20 msec per division.
- e. Select head 0 and write a 2F pattern on entire track. Average minimum amplitude peak-to-peak should be 100 mV.
- f. Select head 1 and write a 2F pattern on entire track. Average minimum amplitude peak-to-peak should be 100 mV.
- g. If either head fails to meet minimum amplitude specifications, continue with procedure.
- h. Install fresh media and recheck.
- i. Check motor speed as per paragraph 10.4.6.
- j. With oscilloscope in "chop" mode, verify that output exists at both TP1 and TP2. If one TP has no output, or significantly less output than other, turn head cable connector over at J4. Should same TP have little or no output, PCB is faulty and needs replacing. If opposite TP now exhibits problem, head assembly is at fault, and should be replaced.

### Track Zero Detector Assembly Adjustment

- a. Apply power to drive and install alignment diskette Shugart 128/126.
- b. Select drive and step to track 00.
- c. Sync oscilloscope external negative on TP7 (-Index). Set time base to 20 msec per division.
- d. Connect one probe to TP1 and other to TP2. Ground probes to PCB. Set input to ac, ADD, and invert one channel. Set vertical deflection to 100 mV/division.
- e. The 125 k Hz signal recorded should be observed at this time.
- f. If 125 k Hz signal is not present, step forward one track at a time and verify 125 k Hz signal is present. Step only five tracks.
- g. Step back towards track 00 detector and verify presence of 125 k Hz signal. Repeat stepping until signal is found.
- h. Once 125 k Hz signal is present on oscilloscope, carriage is located at track 00. Disconnect probes from TP1, TP2, and TP7. Connect one channel to TP8 and set input to dc. Set vertical deflection to 2 V per division. Trigger oscilloscope on selected input channel.
- i. Step to track 01 and verify that TP8 goes to zero.
- j. If not, loosen track 00 bracket.
- k. Set drive to seek alternately between tracks 01 and 02 (455); 02 and 03 (465).
- l. Adjust eccentric until a 50 percent duty cycle is obtained (see figure 10-9).
- m. Tighten track 00 bracket and recheck timing.
- n. If same signal is obtained, remove alignment diskette, power down drive, and reinstall PCB. If same signal is not obtained, repeat steps "k" through "n".

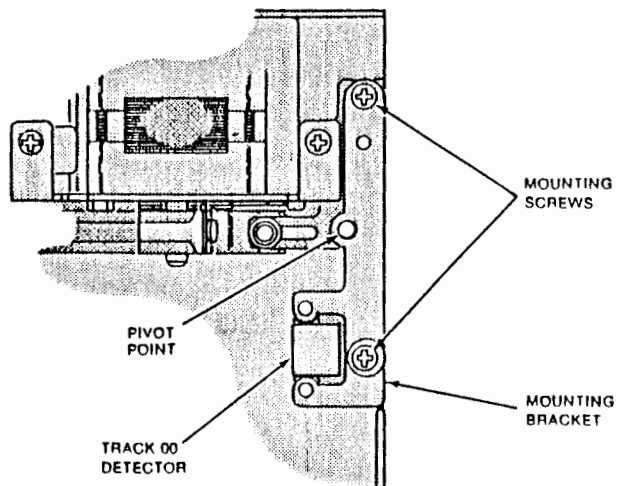


FIGURE 9. TRACK ZERO ADJUSTMENT

#### Index/Sector Timing Adjustment

- a. Insert alignment diskette Shugart 128/126.
- b. Start motor and select head 0.
- c. Step carriage to track 01 (455) or track 02 (465).
- d. Sync oscilloscope external positive on TP7 (+ Index). Set time base to 50  $\mu\text{sec}/\text{division}$ .
- e. Connect one probe to TP1 and other to TP2. Ground probes to PCB. Set Inputs to ac. ADD. and invert one channel. Set vertical deflection to 500 mV/division.
- f. Observe timing between start of sweep and first data pulse. This should be  $250 \pm 150 \mu\text{sec}$ . If timing is not within tolerance, continue on with adjustment. See figure 10-10.
- g. Loosen mounting screw in index detector block until assembly is just able to be moved. See figure 10-11.
- h. Step carriage to track 38 (455) or track 76 (465).
- i. Observing timing, adjust detector until timing is  $250 \pm 150 \mu\text{sec}$ . Ensure that detector assembly is against registration surface on hub frame.
- j. Tighten mounting screw.
- k. Step carriage to track 01 (455) or track 02 (465).
- l. Recheck timing.
- m. Repeat for head 1.

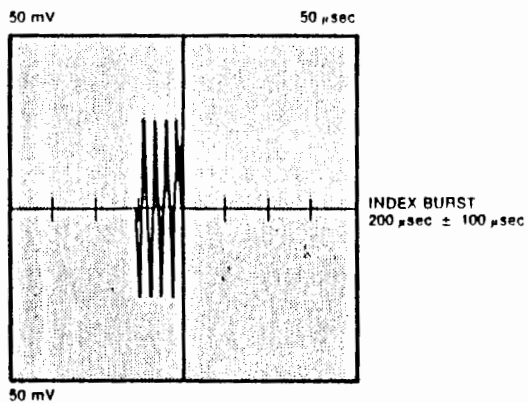


FIGURE 10. INDEX BURST

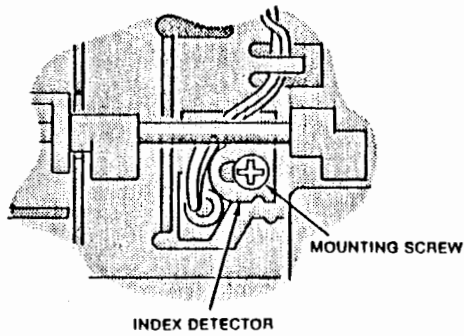


FIGURE 11. INDEX DETECTOR

**Motor Speed Adjustment (Using a Frequency Counter)**

- a. Install 128/126 or 154/155 diskette; start motor and step to track 32.
- b. Connect frequency counter to TP7 (+ Index) on drive PCB.
- c. Adjust pot located on the motor PCB for  $5 \pm 0.05$  Hz (Period =  $200 \pm 2$  msec).

**Write Protect Detector**

- a. Insert diskette into drive. Write protect notch must be open.
- b. Set oscilloscope to AUTO SWEEP, 2 V/division. Monitor TP9.
- c. Check to see if logic level changes when diskette is removed.

**PCB Test Point Locations**

Test points on the 455 and 465 are as follows:

Title	Designation	Description
+ Read Waveform	TP-1	Differential head voltage waveform after amplification and low-pass filter
- Read Waveform	TP-2	
	TP-3	Not used
	TP-4	Not used
Ground	TP-5	Tied to ground plane
+ Digital Read Data	TP-6	A positive pulse with the leading edge corresponding to the transition time of the recorded signal
+ Index Pulse	TP-7	A positive pulse corresponding to the time which the index detector transistor is turned on by the index LED
- Track 00 Detector	TP-8	A negative level corresponding to the track 00 detector being off
+ Write Protect Detector	TP-9	A positive level corresponding to write protect detector being off, (i.e., disk protected)
Signal Ground	TP-10	Tied to ground plane
	TP-11	Not used
- Step Pulse	TP-12	A buffered negative pulse equal to pin 20 of interface connector J-1
- Motor On	TP-13	A buffered negative level equal to pin 16 of interface connector J-1
	TP-14	Not used
Ground (dc)	TP-15	Tied to ground plane

See figure 12 for test point locations.

### NOTE

PCB component locations for P/N 25287 (465) will be supplied at a later date. Test point locations for both the PCB's are the same. See figure 13-2 (schematic) for details.

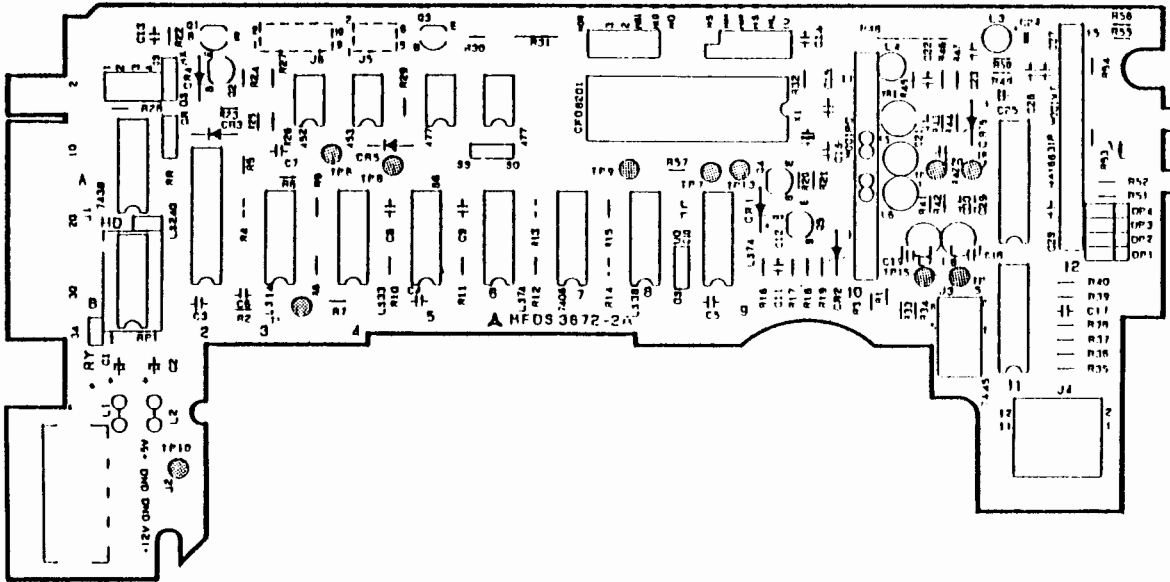


FIGURE 12. PCB COMPONENT/TEST POINT LOCATIONS, P/N 25284 (455), P/N 25286 (485)

## REMOVALS AND REPLACEMENTS

### NOTE

Read the entire procedure before attempting a removal or replacement.

#### Faceplate Latch

- a. Open door. Remove door latch.
- b. Remove mounting screw on each side of faceplate. Pull faceplate forward and away from drive casting.
- c. To reinstall, reverse the above procedure.
- d. No re-adjustment is required after replacement.

#### Direct Drive Motor Assembly

This assembly is not recommended for field replacement.



### **Head and Carriage Assembly**

This assembly is not recommended for field replacement.

### **Stepper Motor and Actuator Assembly**

This assembly is not recommended for field replacement.

### **Clamp Hub Assembly**

- a. Remove PCB.
- b. Open door.
- c. Remove clamp assembly front and rear screws.
- d. To reinstall: Position hub clamp with spacer and spring in place onto spindle hub. (Large end of spring is placed against hub frame.)
- e. Press hub frame down towards spindle until hub shaft is pushed through mounting hole in hub frame.
- f. Reinstall faceplate. Re-adjustment is not required.

### **Write Protect Sensor and Index Detector Assembly**

- a. Remove connector from PCB.
- b. Remove mounting screw from write protect assembly. This will free assembly.
- c. Remove index detector screw to free detector.
- d. Reverse instructions to reinstall.

### **Track Zero Photo Detector Assembly**

- a. Remove PCB and shields from drive.
- b. Remove: white wire from J6 Pin 2,  
green wire from J6 Pin 12,  
yellow wire from J6 Pin 10.
- c. Loosen mounting bracket screws.
- d. Remove two screws securing LED housing to track 00 plate.
- e. To reinstall, reverse above procedure.
- f. Adjust as directed in paragraph 10.4.5.

APPENDIX A

SPARE PART LIST

1.0 GENERAL RECOMMENDED SPARE PART LIST FOR  
728 COMPUTER SYSTEM

Part No.	Description	FOB H.K. US Dollar
-----	-----	-----
IC80A	Z80A CPU IC	2.20
IC8255A	8255A PPI	4.40
IC8910	AY-3-8910 PSG	3.70
IC9129	TMS9129 VDP	12.50
ID0001	IN4148: Silicon Diode	0.10
ID0005	IN4001 Diode	0.10
IL00LS	74LS00	0.30
IL04LS	74LS04	0.30
IL06	7406	0.40
IL07	7407	0.60
IL08LS	74LS08	0.30
IL125LS	74LS125	0.30
IL138LS	74LS138	0.40
IL139LS	74LS139	0.40
IL153LS	74LS153	0.40
IL157LS	74LS157	0.40
IL175LS	74LS175	0.40
IL32LS	74LS32	0.30
IL367LS	74LS367	0.40
IL45	7445	0.70
IL74LS	74LS74	0.30
IM27643	2764 EPROM 300NS	7.80
IM416420-1	4164 D. RAM 200NS	5.10
IM441620	4416 D. RAM 200NS	6.80
IT0004	Transistor 9014B	0.10
KE3010025	0.1U 25V Ceramic Cap	0.10
KT1045E	6.8-45p Top ADJ Trim Cap	0.20
MC32010	X'TAL 10.738635 MHz	0.70
MJ044	5 Pin Ribbon Wire	0.10
MJ045	Jumper Wire 55MM, 22#	0.10
MJ046	Jumper Wire 120mm, 30	0.10
MP082	8 Pin DIN Socket	0.30
MP091	9 Pin Joystick Socket	0.30
MP142A	14 Pin Printer Socket	1.60
MP282	28 Pin IC Socket	0.20
MP501C	25/50 Pin Header	1.50
MP502D	25/50 Pin Edge Connector	1.40
MS004	MIC SW SS-22F32G5	0.30
MS015	5V SPDT Relay	0.90
RF0223	22 Ohm 1/4W 5% Resistor	0.10

Part No.	Description	FOB H.K. US Dollar
-----	-----	-----
RF0823	82 Ohm 1/4W 5% Resistor	0.10
RF1103	100 Ohm 1/4W 5% Resistor	0.10
RF1333	330 Ohm 1/4W 5% Resistor	0.10
RF1473	470 Ohm 1/4W 5% Resistor	0.10
RF2103	1K Ohm 1/4W 5% Resistor	0.10
RF2333	3.3K Ohm 1/4W 5% Resistor	0.10
RF2473	4.7K Ohm 1/4W 5% Resistor	0.10
RF2683	6.8K Ohm 1/4W 5% Resistor	0.10
RF3103	10K Ohm 1/4W 5% Resistor	0.10
RF3203	20K Ohm 1/4W 5% Resistor	0.10
RF3563	56K Ohm 1/4W 5% Resistor	0.10
RF4473	470K Ohm 1/4W 5% Resistor	0.10
SP728004	728 Main PCB	6.80
WA019	QC Label (White)	0.10
WN728004	Edge Connector Bracket	0.20
XC318008M	Heat Sink B (VDP)	0.10
XN011	Nut, M2.6xP0.45	0.10
XN012	Nut M1.7xP0.35	0.10
XS054	M. Screw M3xP0.5x8	0.10
XS066	M. Screw Mx2.6xP0.45x10	0.10
XS067	M. Screw M1.7xP0.35x10	0.10
IA520	PAL Encoder TBA520	1.00
IC741	OA741 Operational AMP.	0.30
ID0002	Rectifier Diode 2A	0.10
ID0005	IN4001 Diode	0.10
IL04LS	74LS04	0.30
IT0004	Transistor 9014B	0.10
IT0006	Transistor PNP 9015	0.10
IT0007	Transistor TIP31	0.40
IT0008	Transistor TIP32	0.40
IT0010	Transistor 9014C	0.10
IZ0001	5V1 Zener 1/2W 5%	0.10
IZ0004	13V Zener Diode 1/2W 5%	0.10
KE3010025	Elect Cap 10uF 25V	0.10
KE3100016	100uF 16V	0.10
KE32200025	2200uF 25V Elect Cap	0.40
KE34700016	4700uF 16V Elect Cap	0.40
KE3470025	470uF 25V Elect Cap	0.20
KR1007050	7PF Ceramic Cap 50V	0.10
KR1015050	15PF Ceramic Cap 50V	0.10
KR1056050	56PF Ceramic Cap 50V	0.10
KR1100050	100PF 50V NPO Cap	0.10
KR1150050	150PF 50V Ceramic Cap	0.10
KR1390050	390PF 50V Ceramic Cap	0.10
KR1470050	470PF 50V Ceramic Cap	0.10
KR1680050	680PF 50V Ceramic Cap	0.10

Part No.	Description	FOB H.K. US Dollar
-----	-----	-----
KR2001050	1000PF 50V Ceramic Cap	0.10
KR2040050	0.047UF 50V Ceramic Cap	0.10
KR2100025	0.10UF 25V Ceramic Cap	0.10
KT1065E	6.8-45p ADJ Trim Cap (10PF)	0.10
MC32004	4.433619 MHZ X'TAL	0.70
MP041	AC Power M Socket 4 Pin	0.30
MS006	Power Switch 3P2T	0.20
RF0223	22 Ohm 1/4W 5% Resistor	0.10
RF1153	150 Ohm 1/4W 5% Resistor	0.10
RF1393	390 Ohm 1/4W 5% Resistor	0.10
RF1473	470 Ohm 1/4W 5% Resistor	0.10
RF1513	510 Ohm 1/4W 5% Resistor	0.10
RF1563	560 Ohm 1/4W 5% Resistor	0.10
RF1683	680 Ohm 1/4W 5% Resistor	0.10
RF2103	1K Ohm 1/4W 5% Resistor	0.10
RF2223	2.2K Ohm 1/4W 5% Resistor	0.10
RF2273	2.7K Ohm 1/4W 5% Resistor	0.10
RF2563	5.6K Ohm 1/4W 5% Resistor	0.10
RF3103	10K Ohm 1/4W 5% Resistor	0.10
RF3223	22K Ohm 1/4W 5% Resistor	0.10
RF3473	47K Ohm 1/4W 5% Resistor	0.10
RV2101H	1K Ohm VR (H. Type)	0.10
RV3101H	10K Ohm VR (H. Type)	0.10
SP728005	Power and Video PCB	0.60
TC0002	3.9uH Coil Q 130	0.10
TC0003	1.35-2.25uH V. Coil Q 100	0.10
XC318004N	Heat Sink A (Power)	0.40
XC318017	Dual RCA Socket	0.20
XS054	M. Screw M3xP0.5x8	0.10
DL003	2x5 LED Red	0.10
ID0001	IN4148: Silicon Diode	0.10
J102515	Sleev'n Tube L=12 D=1.5	0.10
MJ009	12 Wire Connector	0.10
MJ030	PCA-PAL 75 Ohm Cable	0.60
MR318-0-02	Rubber Stand D15x3	0.10
MR328-0-01	Rubber Keyboard (II)	1.80
PB318101-M	Silica Gel Bag Small	0.10
PC728101	Export Carton	0.50
PG728101	Gift Box	1.00
PI318123	Guarantee Card	0.10
PI728101	Instruction Manual	1.70
PP318108	Polyfoam Left	0.30
PP318109	Polyfoam Right	0.30
PP318110	Polyfoam Sheet for Mark II	0.10
PZ318102	Audio Cord PVC Bag	0.10
PZ318103	318 PVC Bag	0.10
PZ318105	Manual PVC Bag	0.10
SP728003	728 Keyboard PCB	6.40

Part No.	Description	FOB H.K. US Dollar
-----	-----	-----
TP066P	230V Supply NOR	5.20
WA003	QC Label	0.10
WA019	QC Label (White)	0.10
WA057	Burn In Label	0.10
WA062	Carton Label	0.10
WA066	Label 230V 50Hz	0.10
WA067	Colour Label (P)	0.10
WA075	SVI-728 Label (40x50 mm)	0.10
WA076	SVI-728 Label (40x100 mm)	0.10
WA079	SVI-728 Serial No. Label	0.10
WN328001	Keyboard Panel	1.00
WN328002	Key Shaft	0.10
WN328003	Space Bar Guide Plate	0.10
WN728001	Computer Top	0.90
WN728002	Computer Base	1.80
WN728003	Cartridge Door	0.10
WN728005	Keytop Set for SVI-728	6.30
XA003	Heat Insulating Fiber	0.10
XC328001	Space Bar Spring	0.10
XC328002	Space Bar Metal Bar	0.10
XC603-5	Door Spring	0.10
XC728001	Nut Plate/W M4T	0.10
XC728004	L-Type Nut Plate	0.10
XC728005	SVI-728 AL-Name Plate	0.20
XS050	S.T. Screw D3x8	0.10
XS056	S.T. Screw D3x10 "A"	0.10
XS059	M. Screw M3x5xP0.5	0.10
XS060	S.T. Screw D2x8	0.10

1.1 RECOMMENDED SPARE PART LIST FOR  
TV SYSTEM AND POWER SUPPLY

Country	Part No.	Description	FOB H.K. US Dollar
-----	-----	-----	-----
S. Africa	MM003	UM1286,U,591.25/6.0	3.50
	TP065P	240V Supply NOR	5.20
France	MM003	UM1286,U,591.25/6.0	3.50
	TP065P	240V Supply NOR	5.20
U.K.	MM003	UM1286,U,591.25/6.0	3.50
	TP065P	240V Supply NOR	5.20
Denmark	MM004	UM1286-2,U,591.25/5.5	3.50
	TP009P	Semko Standard 220V 50Hz	6.90
Finland	MM004	UM1286-2,U,591.25/5.5	3.50
	TP009P	Semko Standard 220V 50Hz	6.90
Norway	MM004	UM1286-2,U,591.25/5.5	3.50
	TP009P	Semko Standard 220V 50Hz	6.90
Sweden	MM004	Um1286-2,U,591.25/5.5	3.50
	TP009P	Semko Standard 220V 50Hz	6.90
Austria	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
S. Arabia	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
Belgium	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
W. Germany	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
Italy	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
Malta	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
Netherlands	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
Turkey	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
Yemen	MM004	UM1286-2,U,591.25/5.5	3.50
	TP066P	230V Supply NOR	5.20
Australia	MM008	UM1285E3/4,V, 55,62.7/5.5	3.60
	TP011P	240V/SAA Plastic Adaptor	6.50
New Zealand	MM008	UM1285E3/4,V, 55,62.7/5.5	3.60
	TP011P	240V/SAA Plastic Adaptor	6.50

Country	Part No.	Description	FOB H.K. US Dollar
-----	-----	-----	-----
Israel	MM008	UM1285E3/4,V, 55,627/5.5	3.60
	TP065P	240V Supply NOR	5.20
Kuwait	MM008	UM1285E3/4,V, 55,627/5.5	3.60
	TP065P	240V Supply NOR	5.20
Malaysia	MM008	UM1285E3/4,V, 55,627/5.5	3.60
	TP065P	240V Supply NOR	5.20
Greece	MM008	UM1285E3/4,V, 55,627/5.5	3.60
	TP065P	240V Supply NOR	5.20
Iceland	MM008	UM1285E3/4,V, 55,627/5.5	3.60
	TP009P	SEMKO Standard 220V 50Hz	6.90
Hong Kong	MM003	UM1285,V,591.25/6.0	3.50
	TP064P	200V Supply for HK	5.20

# ILLUSTRATED PARTS CATALOG

## DESCRIPTION

The Illustrated Parts Catalog (IPC) is provided for the users to identify parts in an assembly. The figures precede the parts listing and appear either directly above the list or on the next page. The first column of the IPC lists the referenced part in the figures. The second column refers to the Shugart part number for that part. The third column describes the name and description of the part. The fourth column describes the quantity used for that assembly.

## QUANTITY PER ASSEMBLY

The quantity listed is the quantity used on basic drive assembly as shown in the figure.

## RECOMMENDED SPARE PARTS STOCKING GUIDE

The spare parts stocking guide is broken down into three levels. These levels are: Site or Field Support Engineer (level 1), Branch Office (level 2), and Depot or Headquarters (level 3). It is assumed that the Site is replenished by the Branch immediately, and the Branch is replenished by the Depot within 30 days.

The inventories that the three levels should maintain are:

Site	1 to 20 machines
Branch	1 to 100 machines
Depot	Unlimited
Depot Parts Only	
Branch Replenishment	Same as Branch ratio

Table 2 shows the spare parts required to support the Shugart 455/465 drives in the field.

## NOTE

Some Depot parts are only unique to the depot. Stocking levels to back up Branch stocks are shown. These quantities are only a guide and may exceed or not meet each individual requirement. Requirements can vary due to usage, applications, and repair philosophies. This guide should be modified as required.

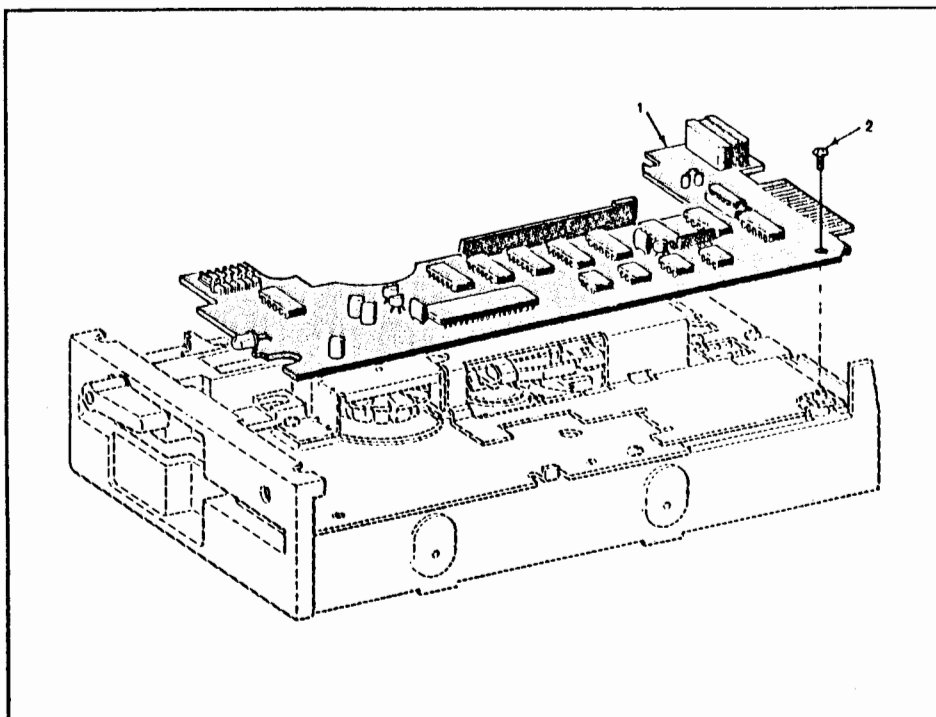


FIGURE 1. 455/465 BASIC  
DRIVE ASSEMBLY  
(SHEET 1 OF 4)



**455/465 BASIC DRIVE ASSEMBLY**

REFERENCE NUMBER	PART NUMBER	DESCRIPTION	QTY
Ref	51756	455/465 MINIFLOPPY™ DRIVE	
1	25284	455 PCB	1
	25286	465 PCB	1
2	11477	SCREW	2

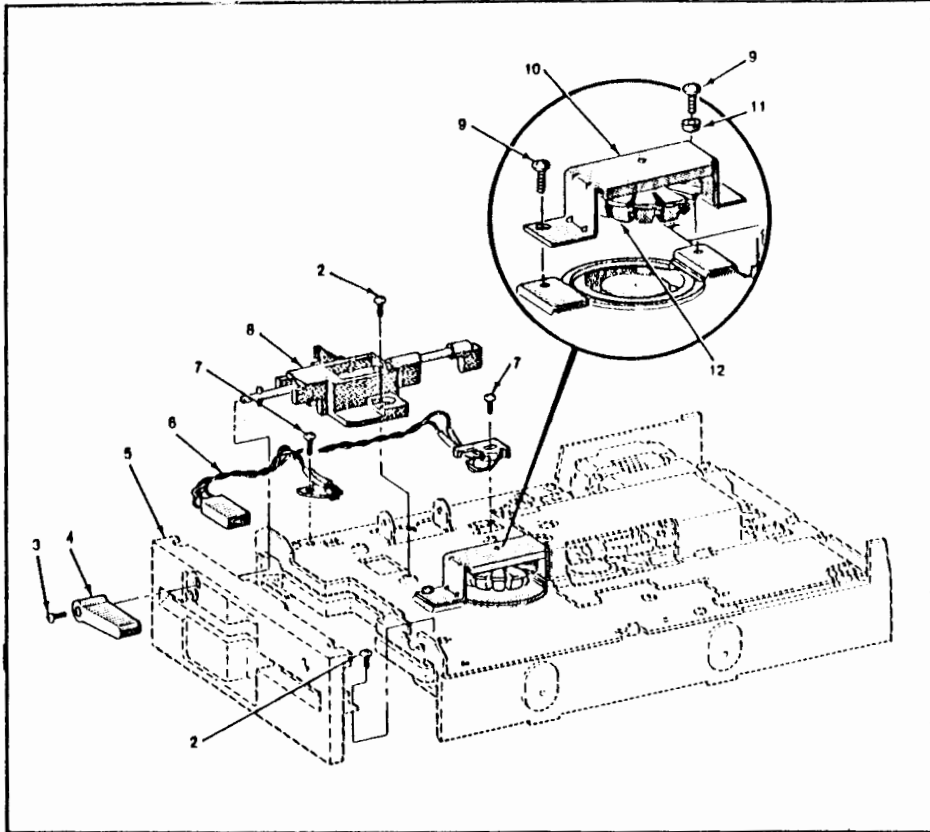


FIGURE 1. 455/465 BASIC DRIVE ASSEMBLY (SHEET 2 OF 4)

**455/465 BASIC DRIVE ASSEMBLY (CONT.)**

REFERENCE NUMBER	PART NUMBER	DESCRIPTION	QTY
3	11464	SCREW	1
4	51782	CLAMP HANDLE	1
5	52446	FACEPLATE	1
6	54758	WRITE PROTECT/INDEX DETECTOR	1
7	11475	SCREW	2
8	51896	GUIDE SHAFT ASSEMBLY	1
9	11450	SCREW	2
10	51898	CLAMP CAM ASSEMBLY	1
11	51788	SPACER	1
12	54765	COLLET	1

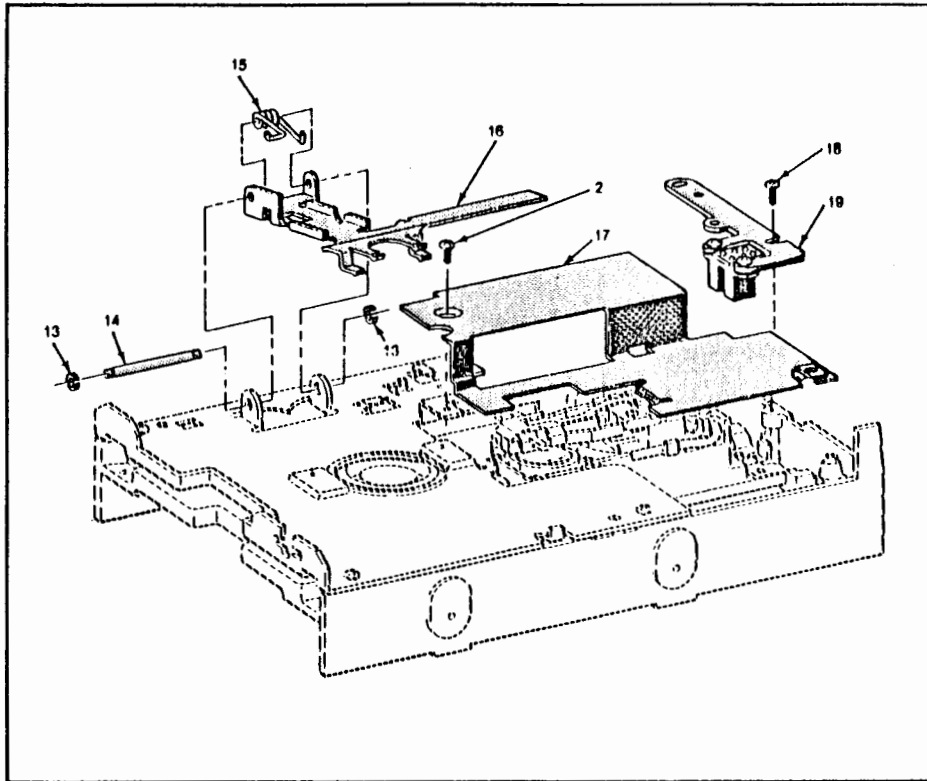


FIGURE 1. 455/465 BASIC  
DRIVE ASSEMBLY  
(SHEET 3 OF 4)

455/465 BASIC DRIVE ASSEMBLY (CONT.)			
REFERENCE NUMBER	PART NUMBER	DESCRIPTION	QTY
13	11461	E-RING	2
14	51785	LIFT SHAFT	1
15	51784	LIFT SPRING	1
16	51786	LIFTER	1
17	54768	SHIELD PLATE	1
18	11468	SCREW	1
19	54762	TRACK 00 SENSOR	1

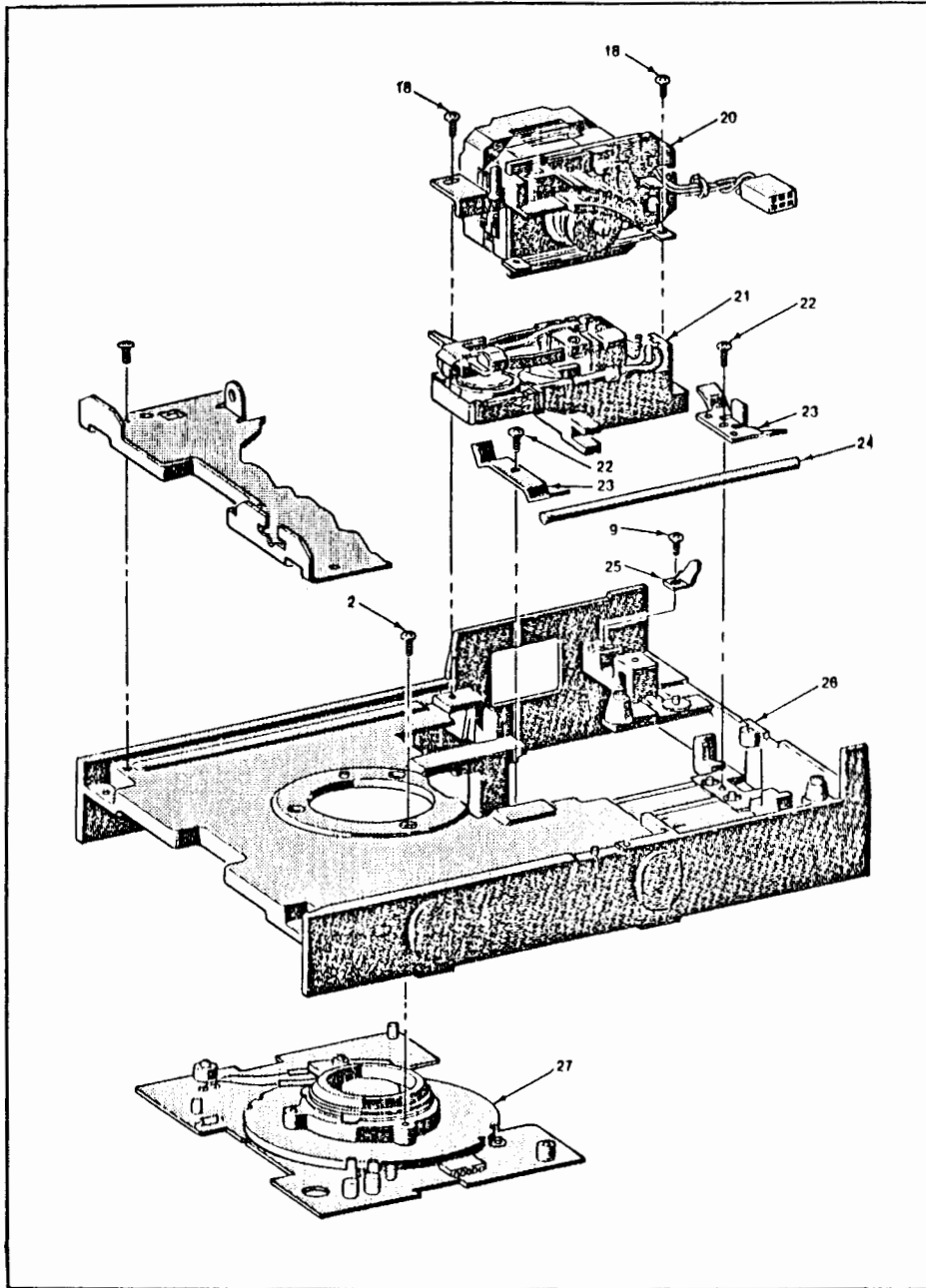


FIGURE 1. 455/465 BASIC  
DRIVE ASSEMBLY  
(SHEET 4 OF 4)

455/465 BASIC DRIVE ASSEMBLY (CONT.)

REFERENCE NUMBER	PART NUMBER	DESCRIPTION	QTY
20	52445*	STEPPER MOTOR ASSEMBLY	1
21	54773*	455 HEAD CARRIAGE ASSEMBLY	1
	54814*	465 HEAD CARRIAGE ASSEMBLY	1
22	11476	SCREW	2
23	51895	CLAMP, Guide Rod	2
24	54771	GUIDE ROD	2
25	11456	FASTEN TAB	1
26	51887	BASE	1
27	54770*	DRIVE MOTOR	1
28	54767	CARRIDGE GUIDE	1

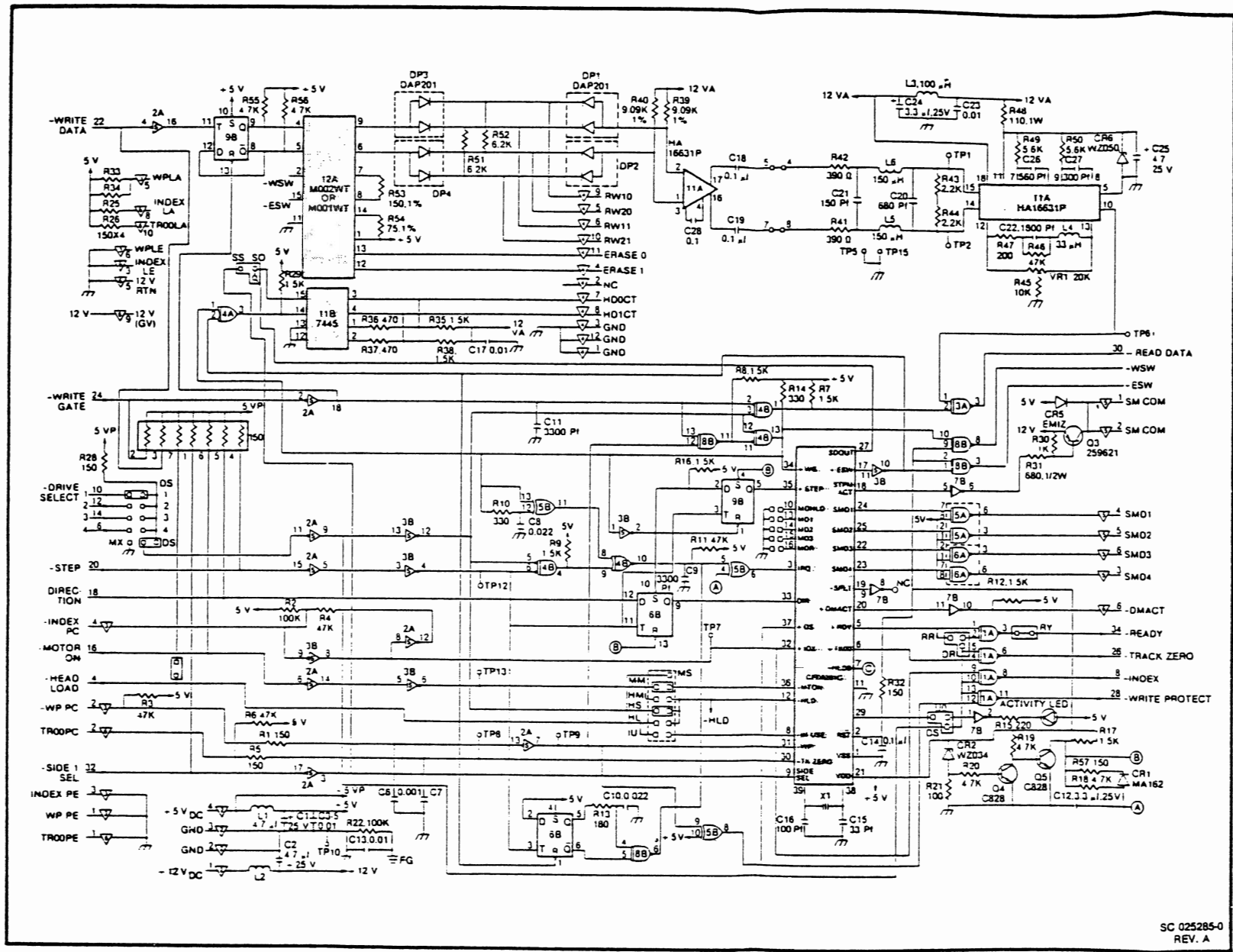
\*NOT FIELD REPLACEABLE

PART NUMBER	DESCRIPTION	QUANTITY PER LEVEL		
		SITE	BRANCH	DEPOT
25284	PCB, 455	1	3	4
25286	PCB, 465	1	3	4
51781	LOCK CAM			2
51782	CLAMP HANDLE	1	4	4
51783	LOCK SPRING	1	2	2
51784	LIFTER SPRING	1	2	2
51785	LIFTER SHAFT		2	2
51896	GUIDE SHAFT ASSEMBLY			2
51899	CLAMP BEARING			2
52448	FACEPLATE		2	3
54755	FACEPLATE (FULL HEIGHT)		2	3
54758	WRITE PROTECT/INDEX DETECTOR ASSEMBLY	1	3	3
54782	TRACK 00 ASSEMBLY	1	3	3
54785	COLLET ASSEMBLY		2	2
54768	SHIELD PLATE		1	2
54771	GUIDE ROD			2

TABLE 1. 455/465 SPARE PARTS STOCKING GUIDE

## SCHEMATIC DIAGRAMS

The following schematic diagrams are furnished as an aid to malfunction analysis of PCB's 25284 (455) and 25287 (465).

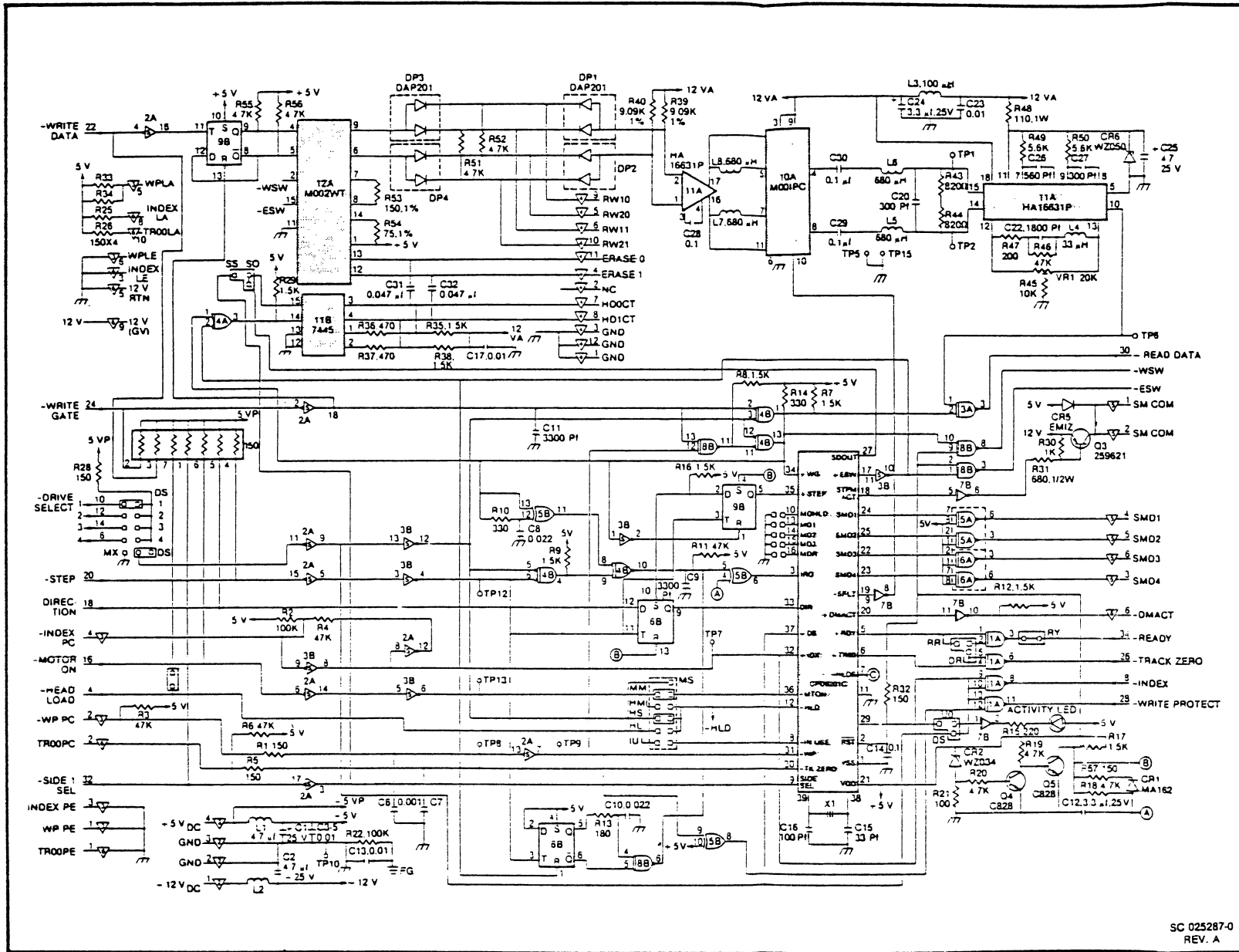


IC TYPE	REF. DES.	QUAT.
7438	1A	1
7406	7B (4/6)	1
74LS14	3B	1
74S38	8B	1
74LS33	4B	1
7445	11B	1
74LS74	6B, 9B	2
75452	3A	1
75453	4A	1
75477	5A, 6A	2
74LS240	2B	1
53286	5B (2/4)	1
HA16631P	11A	1
CFD8201C		1
MO02WT	12A	1
OR MO01WT	12A	1

MODE SELECTION	
MD1	OPEN
MD2	OPEN
MD3	SHORT
MD4	OPEN
MDHLD	SHORT

REFERENCE DESIGNATION	
LAST USED	NOT USED
R57	R41, R42, R23, R24, R27
C28	
CR6	CR3, CR4
Q5	Q1, Q2
L6	
TP15	3, 4, 11, 14

FIGURE 1. SCHEMATIC DIAGRAM (455 PCB)



IC TYPE	REF. DES.	QUAT.
7438	1A	1
7406	7B (4/8)	1
74LS14	3B	1
74LS38	8B	1
74LSJ1	4B	1
7445	11B	1
74LS74	6B,9B	2
75452	3A	1
75453	4A	1
75477	5A,6A	2
74LS240	2B	1
53286	5B (2/4)	1
HA16631P	11A	1
CFD8201C		1
M001PC	10A	1
M002WT	12A	1

MODE SELECTION	
MD1	OPEN
MD2	OPEN
MD3	SHORT
MD4	OPEN
MDHLD	SHORT

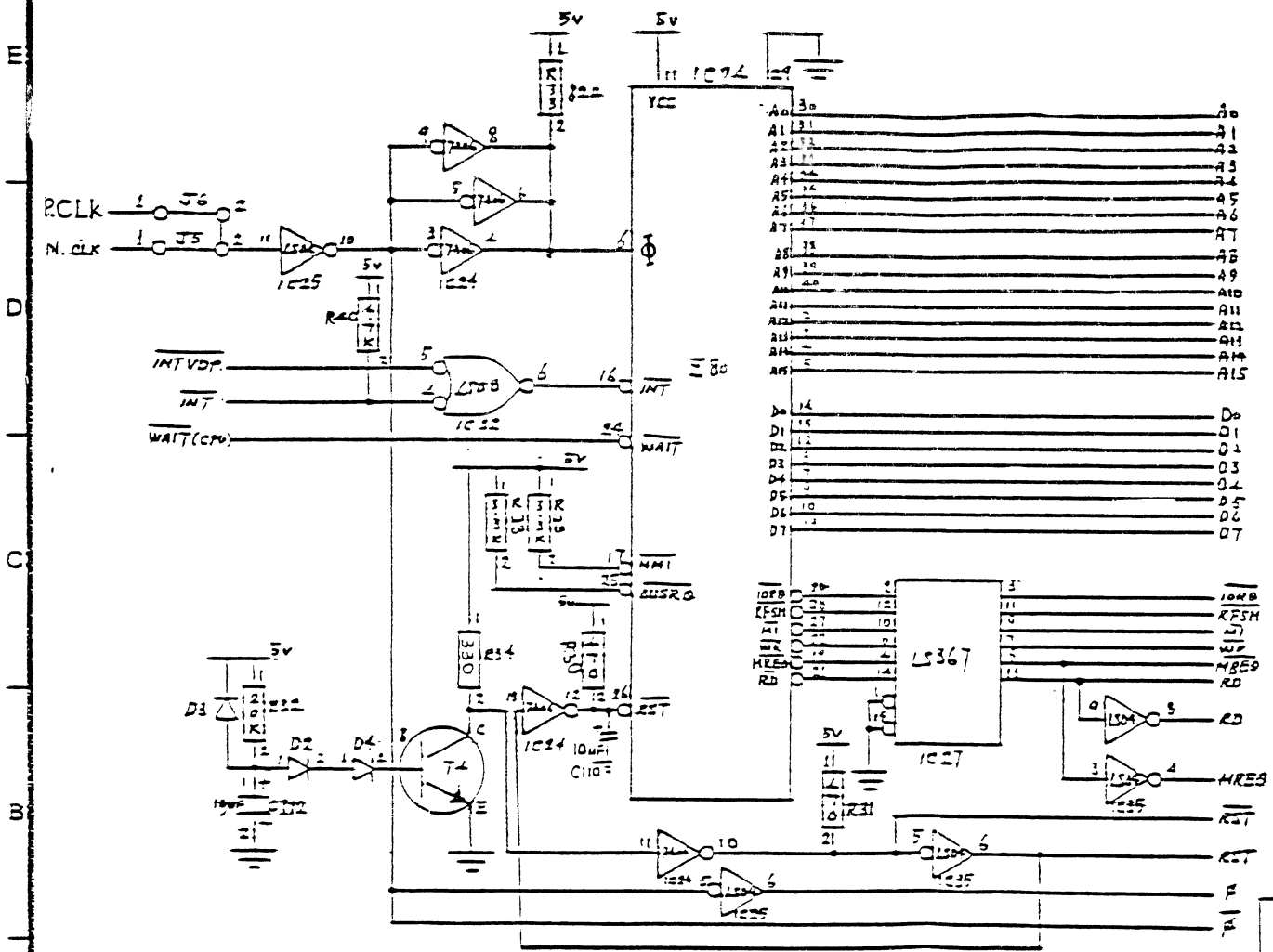
REFERENCE DESIGNATION	
LAST USED	NOT USED
R57	R41,R42,R23,R24,R27
C32	C18,C19,C21
CR6	CR3,CR4
O5	O1,O2
L8	
TP15	3,4,11,14

FIGURE 2. SCHEMATIC DIAGRAM (465 PCB)

APPENDIX B

CIRCUIT DIAGRAM

- EPROM Version(728 MSX Computer System)
- MASK ROM Version(728 MSX Computer System)
- Peripheral Drawings



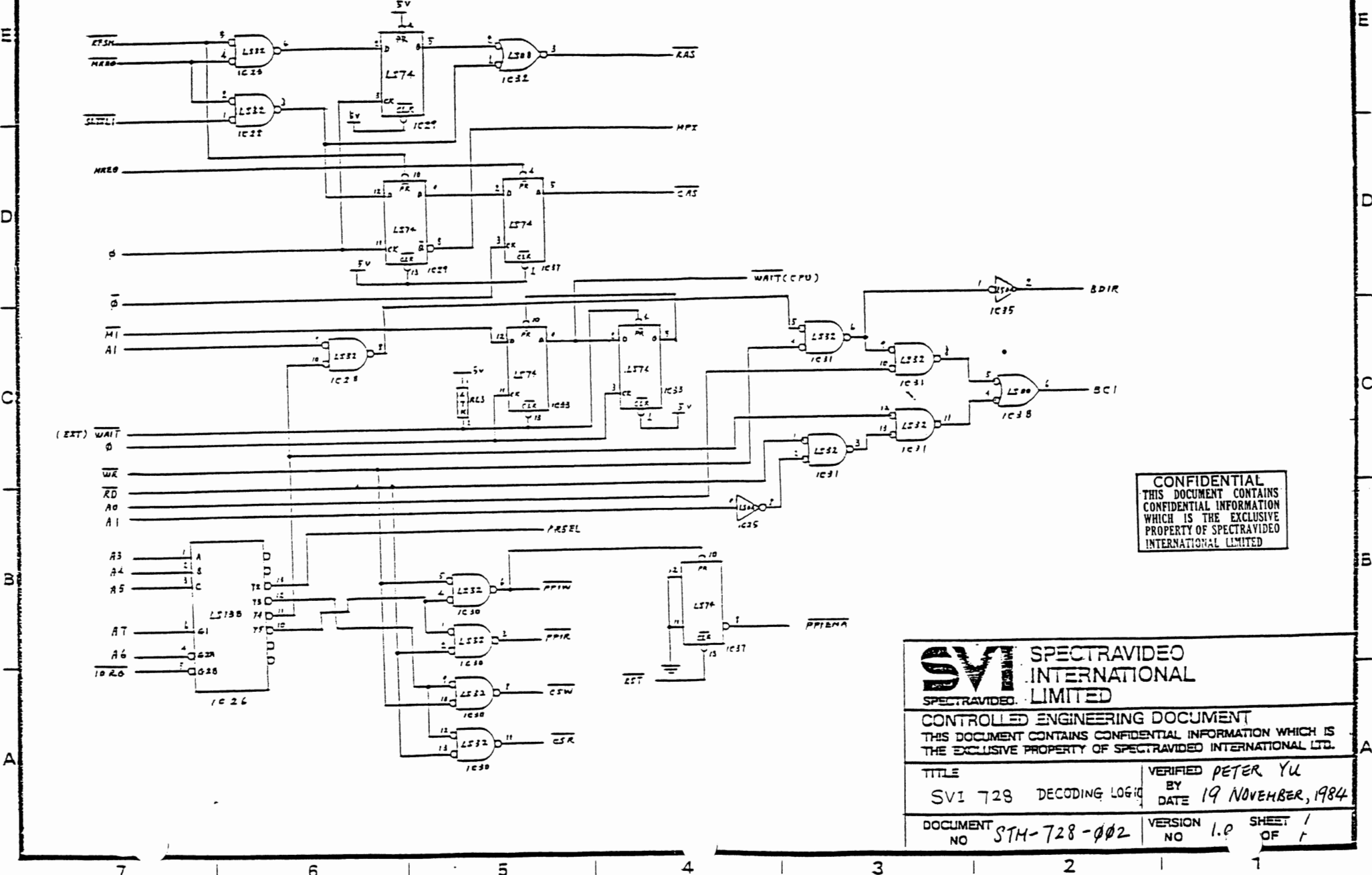
CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

CONTROLLED ENGINEERING DOCUMENT  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE	SVI 728 CPU	VERIFIED BY	PETER YU
		DATE	19 NOVEMBER, 1984
DOCUMENT NO	STM-728-001	VERSION NO	1.0
		SHEET OF	1 / 1





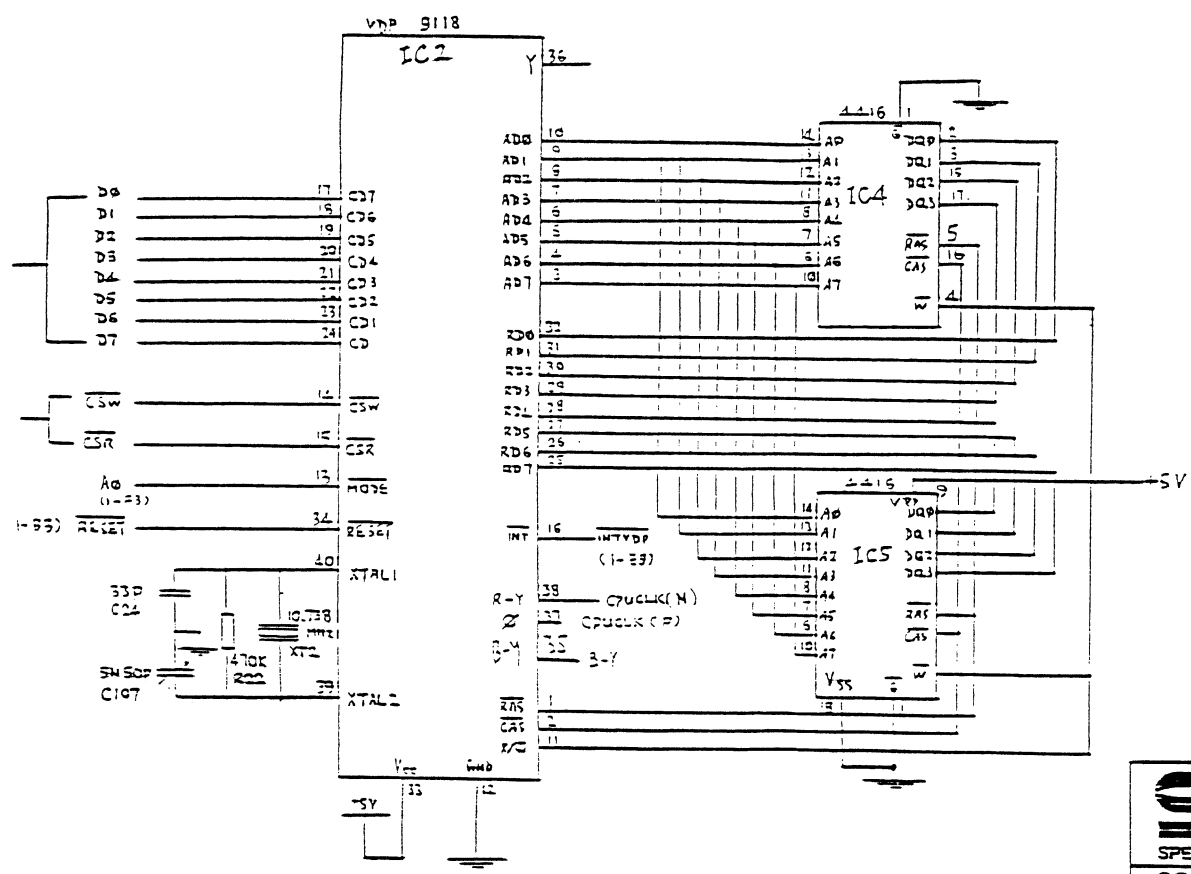
**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI SPECTRAVIDEO  
 INTERNATIONAL  
 LIMITED**

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

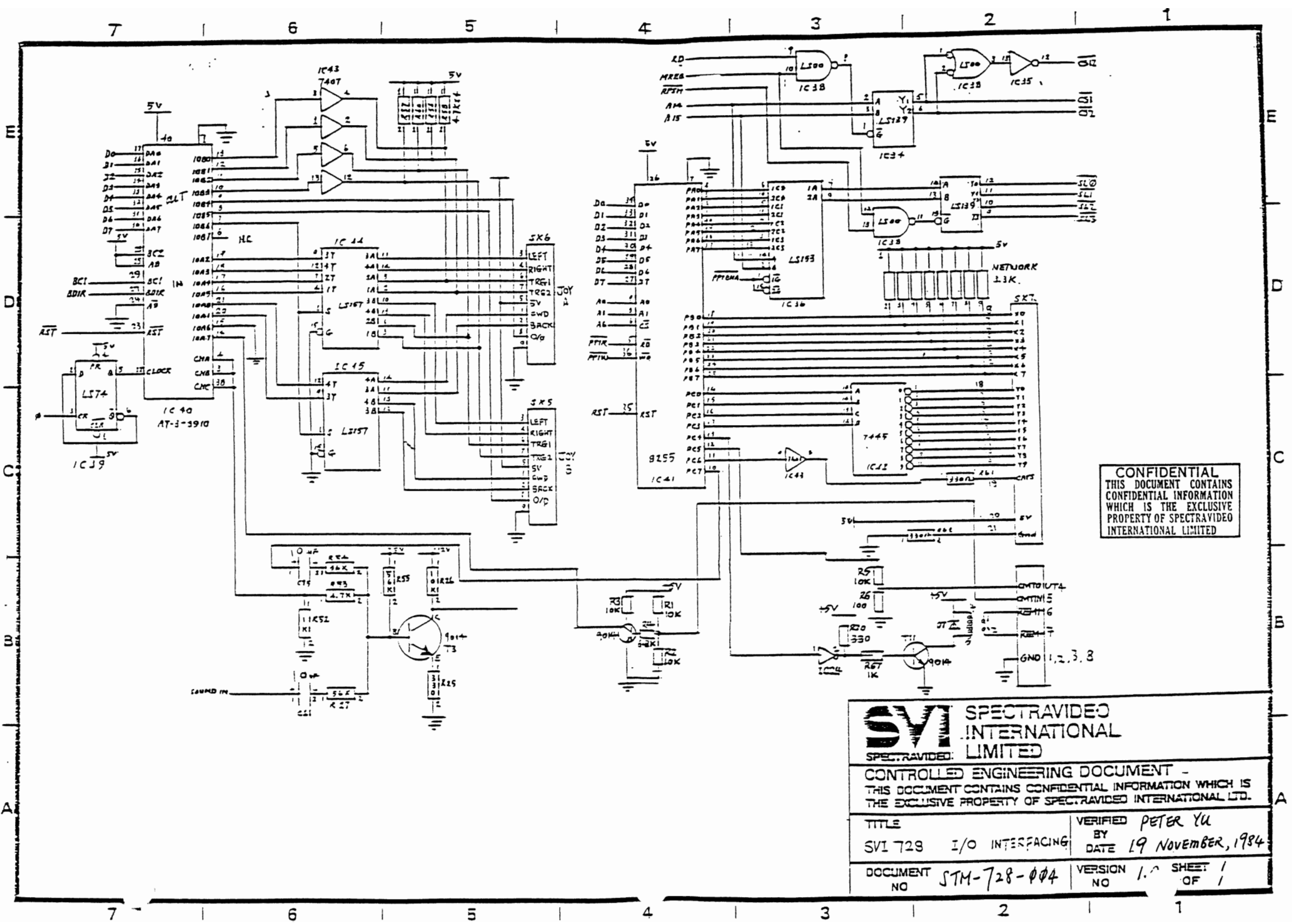
TITLE SVI 728 DECODING LOGIC  
 VERIFIED BY PETER YU  
 DATE 19 NOVEMBER, 1984

DOCUMENT NO STH-728-002  
 VERSION NO 1.0  
 SHEET 1 OF 1



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SPECTRAVIDEO INTERNATIONAL LIMITED</b>	
CONTROLLED ENGINEERING DOCUMENT THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE SVI 728 VDP & V.RAM	VERIFIED BY PETER YU DATE 19 NOVEMBER, 1984
DOCUMENT NO STM-728-003	VERSION NO 1.0 SHEET / F / 1

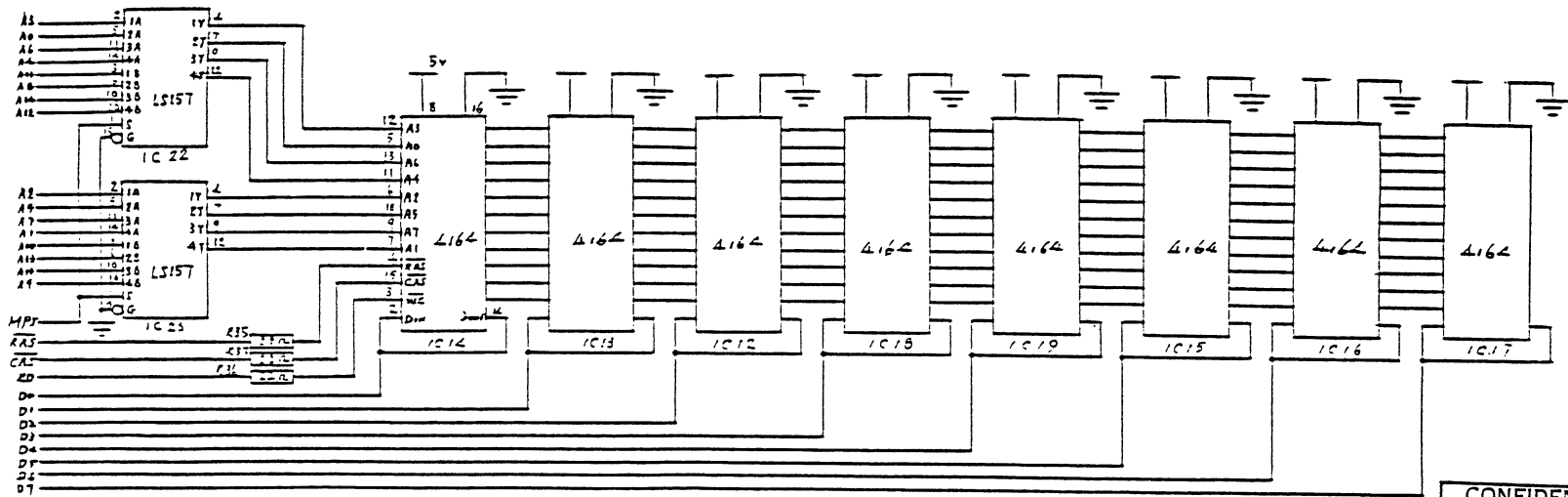


CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

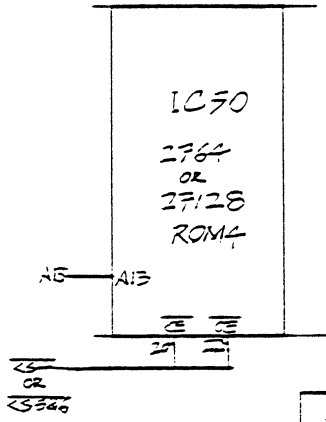
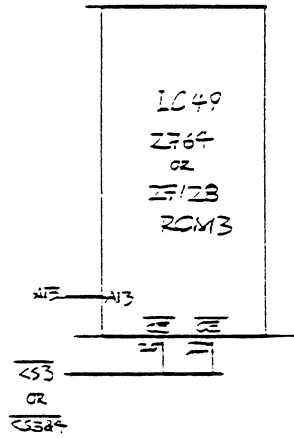
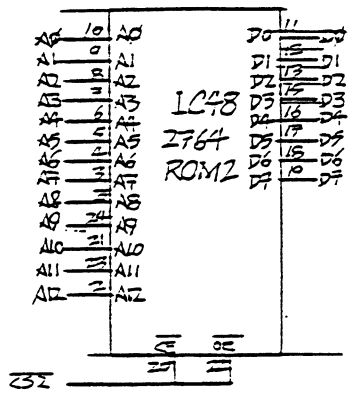
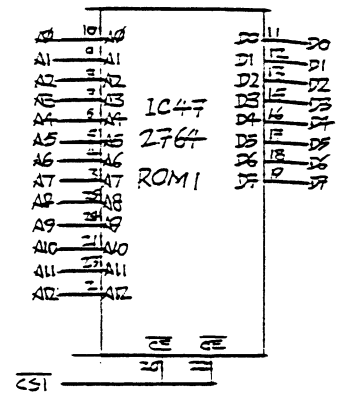
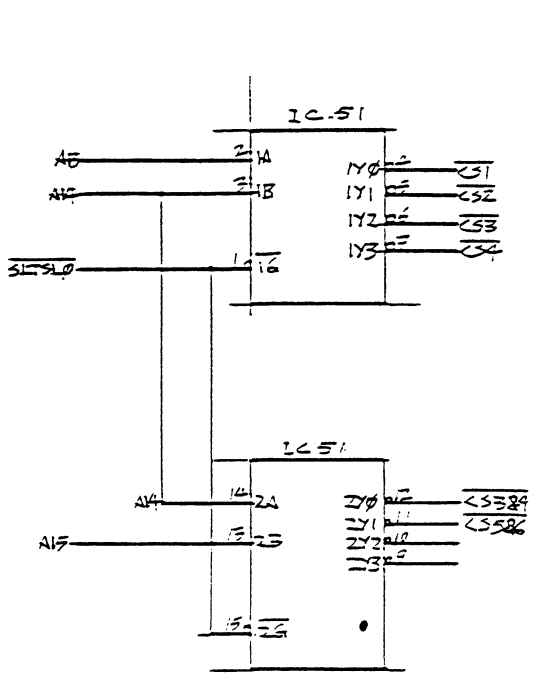
CONTROLLED ENGINEERING DOCUMENT -  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE SVI 728 I/O INTERFACING	VERIFIED BY PETER YU
DOCUMENT NO STM-728-004	DATE 19 NOVEMBER, 1984
VERSION NO	SHEET 1 OF 1



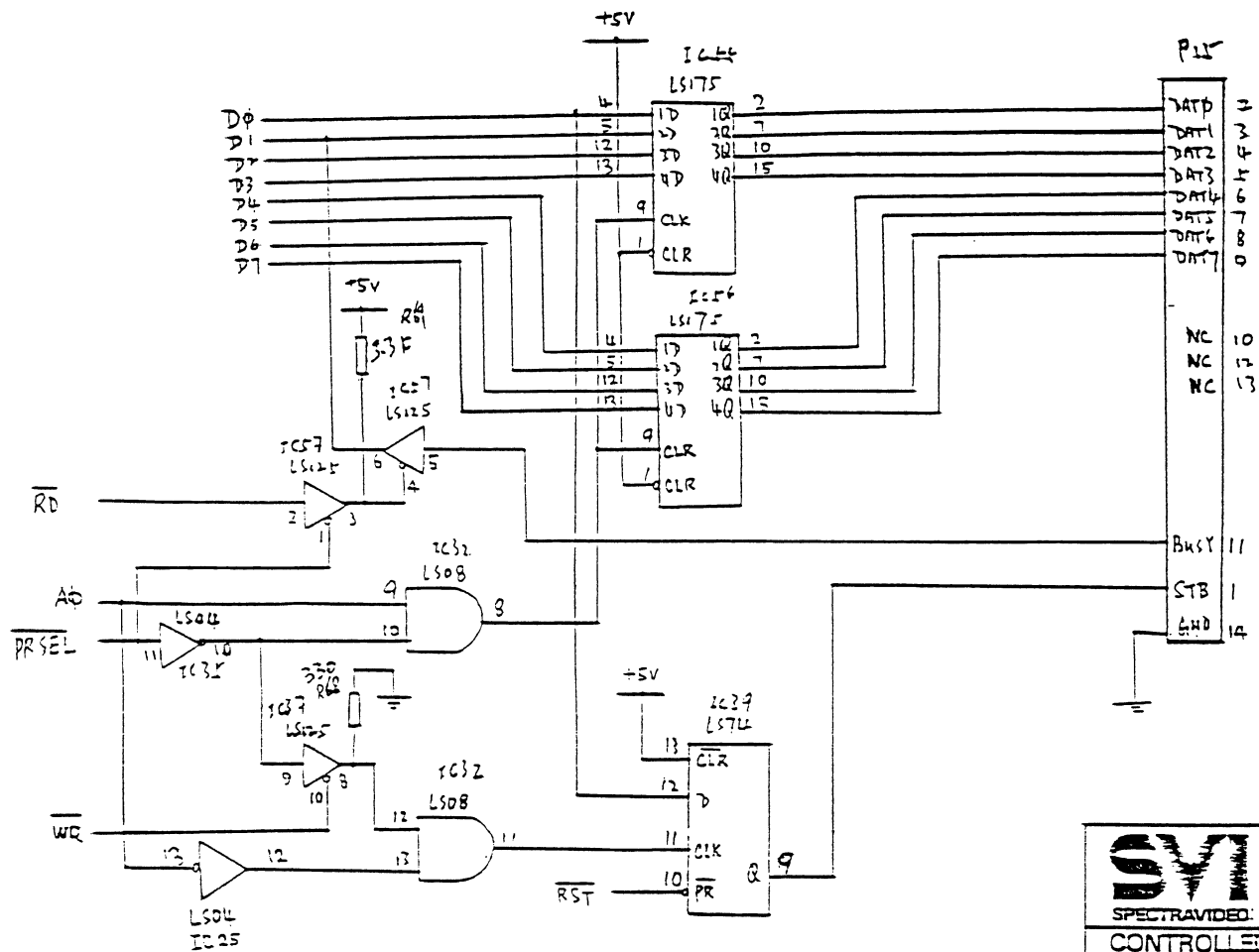
CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.		
TITLE SVI 728 RAMS	VERIFIED BY PETER YU	DATE 19 NOVEMBER, 1984
DOCUMENT NO STM-728-005	VERSION NO 1.0	SHEET OF 1 / 1



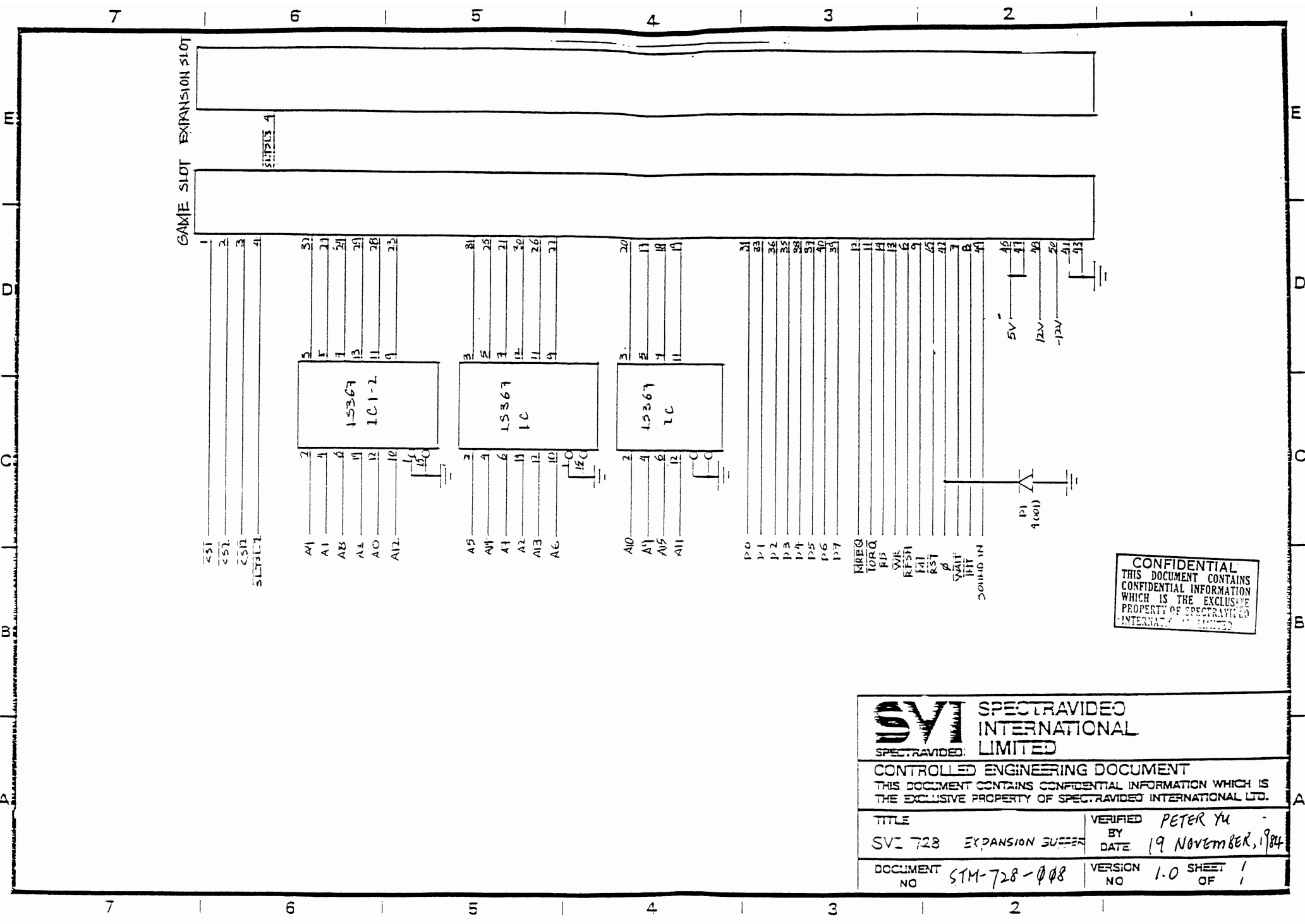
**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI</b> SPECTRAVIDEO INTERNATIONAL LIMITED	
CONTROLLED ENGINEERING DOCUMENT THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE SVI 728 ROM	VERIFIED BY PETER YU DATE 19 NOVEMBER, 1984
DOCUMENT NO. STM-728-006	VERSION NO 1.0 SHEET OF 1



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SV SPECTRAVIDEO INTERNATIONAL LIMITED</b>	
CONTROLLED ENGINEERING DOCUMENT THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE SVI 728 PRINTER INTERFACING	VERIFIED BY PETER YU DATE 19 NOVEMBER, 1984
DOCUMENT NO STM-728-007	VERSION NO 1.0 SHEET OF /



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 INTERNATIONAL  
 SPECTRAVIDEO LIMITED

CONTROLLED ENGINEERING DOCUMENT  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE	VERIFIED BY	PETER YU
SVC 728 EXPANSION BUFFER	DATE	19 November, 1984

DOCUMENT NO	STM-728-008	VERSION NO	1.0	SHEET OF	1 / 1
-------------	-------------	------------	-----	----------	-------

A W Q D M

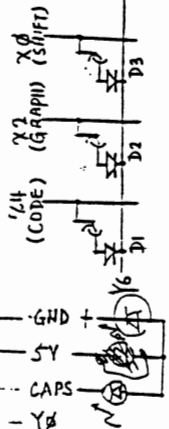
7 6 5 4 3 2

	Y7	Y6	Y5	Y4	Y3	Y2	Y1	X0
	7	6	5	4	3	2	1	0
	. : . }	u A	J	B	q	f	= +	/
	b B	i I	j J	h H	g G	e E	d D	c C
	T R	q Q	P P	o O	n N	m M	l L	k K
	Z Z	Y Y	X X	w W	v V	u U	t T	s S
Y6	F3	F2	F1	CODE	CAPS, GRPH	CTRL	SHIFT	
Y7	RETURN	SELECT	BS	STOP	TAB	ESC	F5	F4 V
Y8	→	↓	↑	←	DEL	INS	HOME	SPACE
Y9					/ (R)	* (R)	- (R)	+ (R)

( 000  
0001.  
0010 ✓  
0011 V  
0100  
0101 ✓  
0110 ✓  
0111 ✓  
1000  
1001

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21

Y6 Modification



CONFIDENTIAL  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

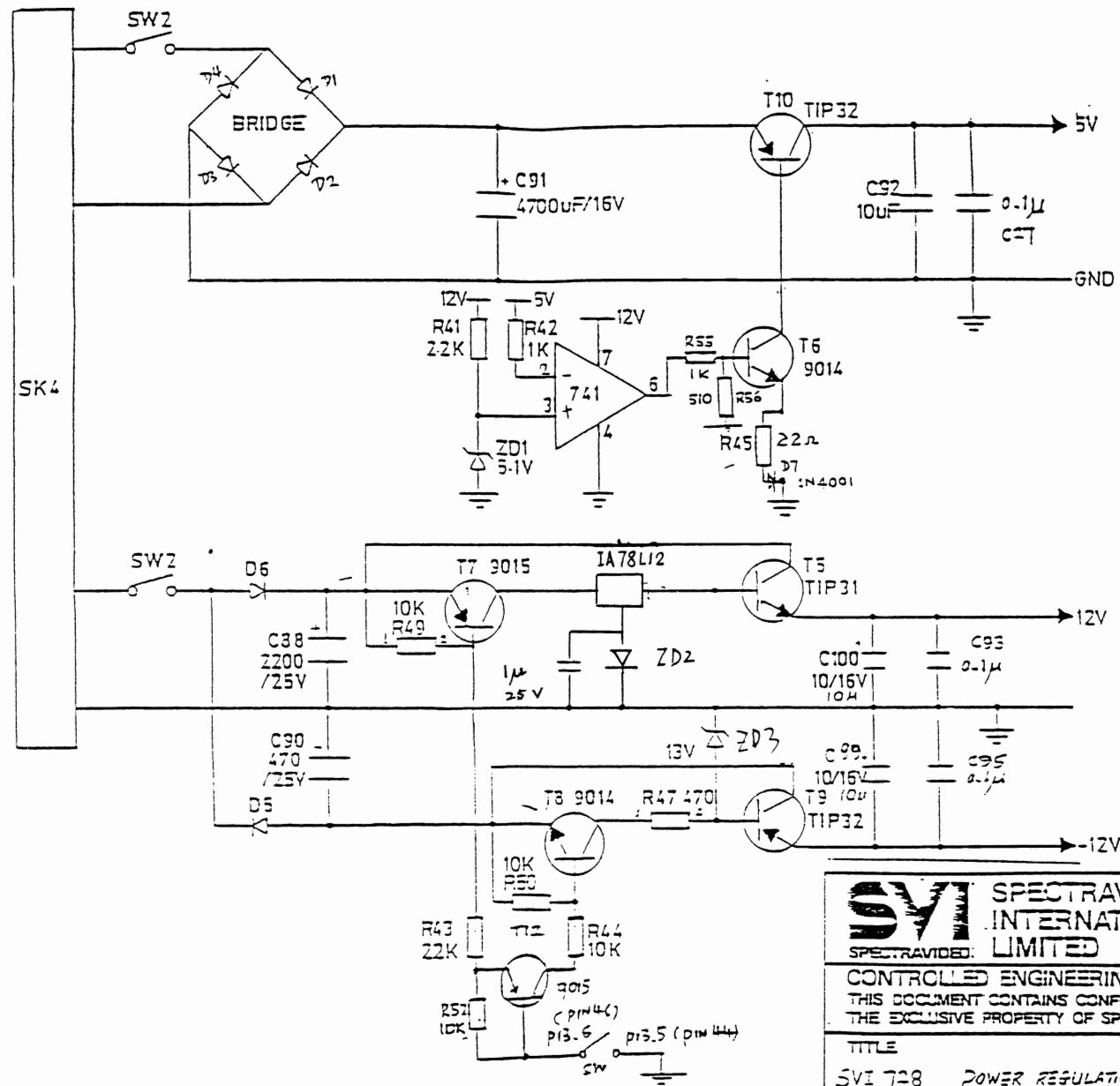
**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

CONTROLLED ENGINEERING DOCUMENT  
THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE	VERIFIED	PETER YU
SVI 728	BY	DATE
KEYBOARD MATRIX	DATE	19 November, 1984
DOCUMENT NO	VERSION NO	SHEET OF
SYM-728-009	1.0	11

7 6 5 4 3 2



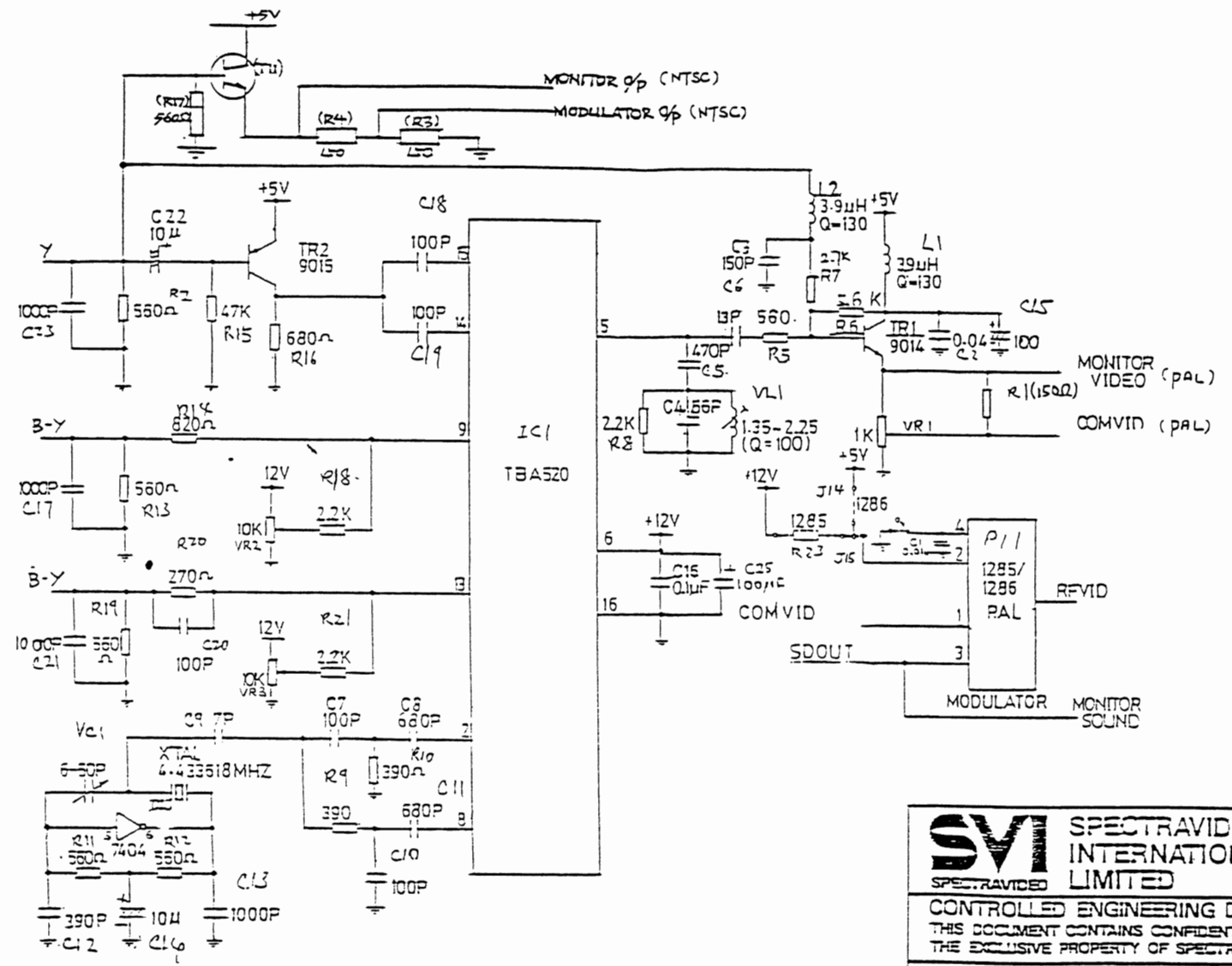


CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

CONTROLLED ENGINEERING DOCUMENT  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

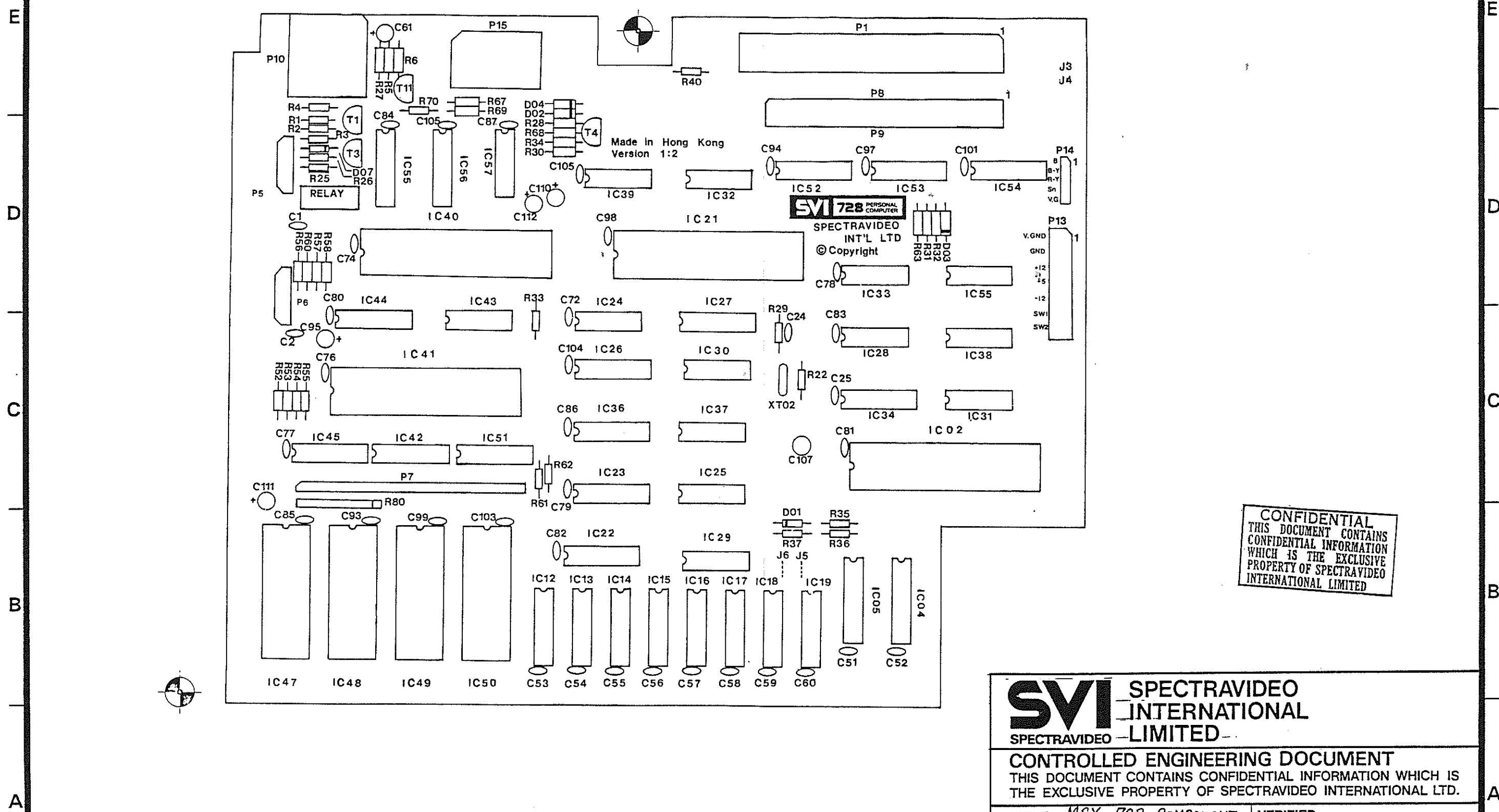
TITLE	SVI 728 POWER REGULATION	VERIFIED BY	PETER YU
DATE		DATE	19 November, 1984
DOCUMENT NO	Sim-728-010	VERSION NO	1.0
		SHEET OF	1 / 1



**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

CONTROLLED ENGINEERING DOCUMENT THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.		
TITLE	VERIFIED	PETER Yu
SVI 728 PAL ENCODING	BY	
	DATE	19 NOVEMBER, 1984
DOCUMENT NO	VERSION NO	SHEET OF
STM-728-p11	1.0	1 / 1

7 6 5 4 3 2 1

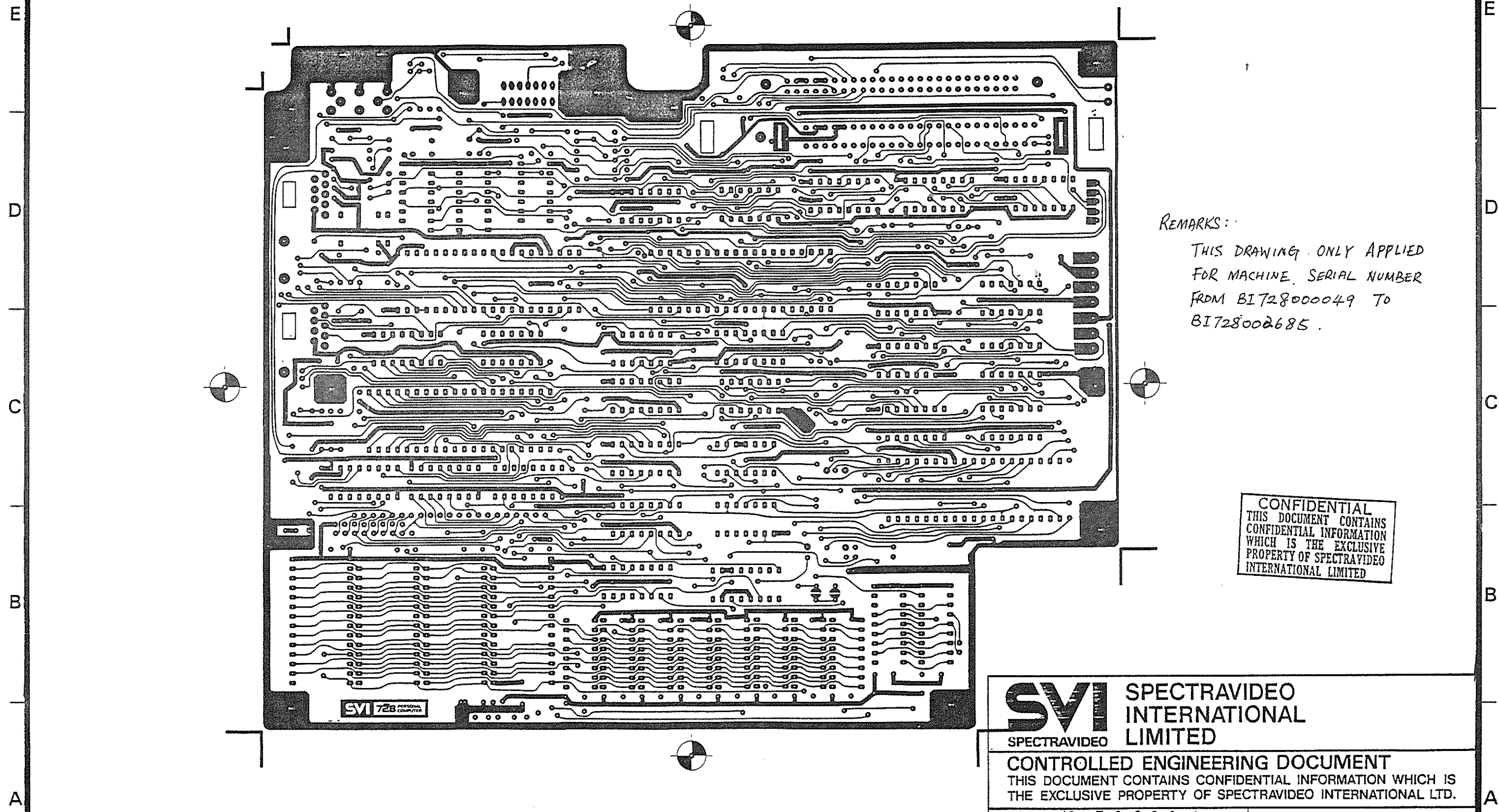


**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE—MSX 728 COMPONENT DRAWING (MAIN BOARD) — EPROM —	VERIFIED BY— KIMBA LAU DATE 27 <sup>th</sup> DEC. 1984
DOCUMENT NO STM-728-φ12	VERSION SHEET / NO 1.2 OF 1

7 6 5 4 3 2 1

7 6 5 4 3 2 1



REMARKS:  
 THIS DRAWING ONLY APPLIED  
 FOR MACHINE SERIAL NUMBER  
 FROM B1728000049 TO  
 B1728002685.

CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 SPECTRAVIDEO INTERNATIONAL  
 LIMITED

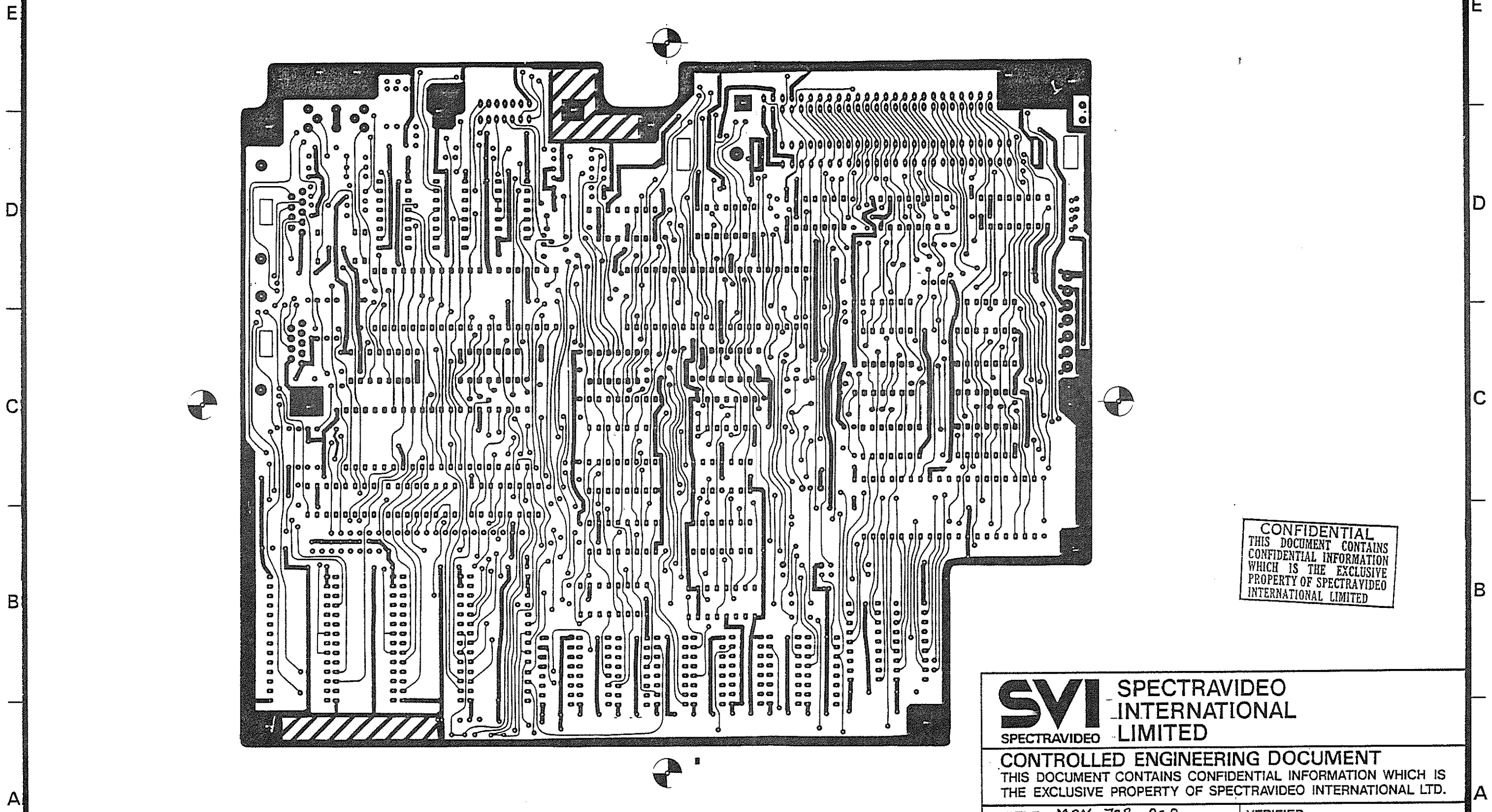
**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE <i>MSX 728 PCB COMPONENT SIDE LAYOUT (MAIN BOARD) - EPROM VERSION -</i>	VERIFIED BY <i>KIMBA LAU</i> DATE <i>27th DEC. 1984</i>
---	--

DOCUMENT NO <i>STM-728-φ13</i>	VERSION NO <i>1.2</i>	SHEET <i>1</i> OF <i>1</i>
--------------------------------	-----------------------	-------------------------------

7 6 5 4 3 2 1

7 6 5 4 3 2 1



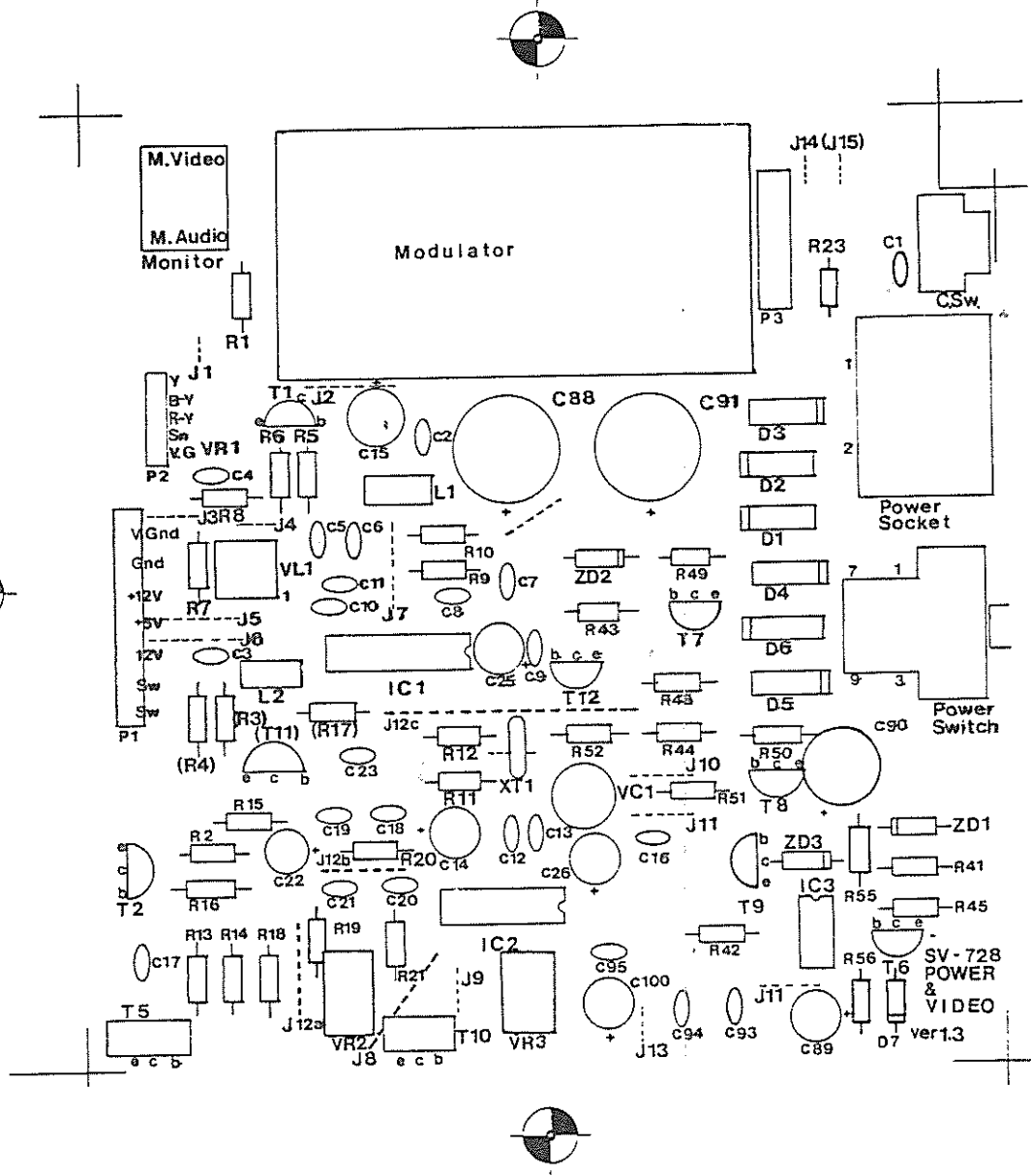
CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI</b> SPECTRAVIDEO SPECTRAVIDEO INTERNATIONAL LIMITED	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE <i>MSX 728 PCB          SOLDER SIDE LAYOUT (MAIN          BOARD) — EPROM —</i>	VERIFIED BY <i>KIMBA LAU</i> DATE <i>27th DEC. 1984</i>
DOCUMENT NO <i>STM-728-Ø14</i>	VERSION NO <i>1.2</i> SHEET <i>1</i> OF <i>1</i>

7 6 5 4 3 2 1

0 5 4 3 2 1

E  
D  
C  
B



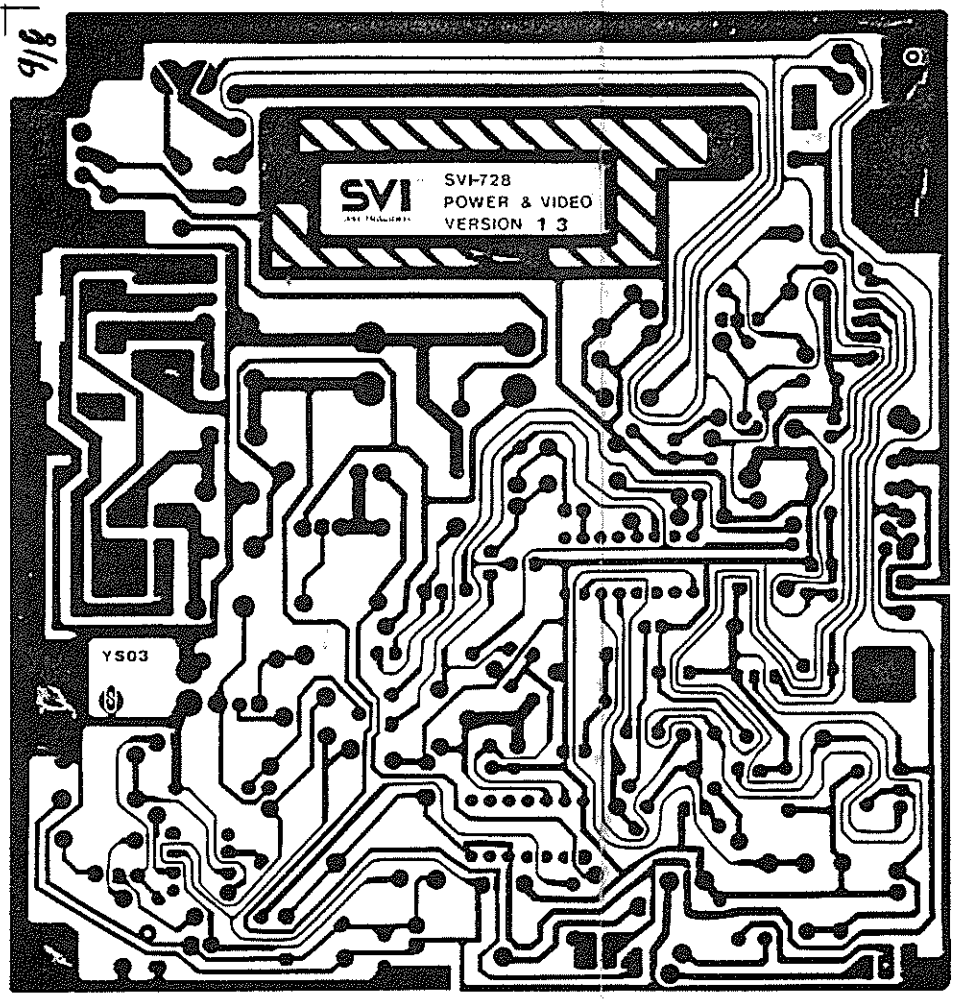
**CONFIDENTIAL**  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE <i>MSX 728 COMPONENT DRAWING (POWER AND VIDEO BOARD)</i>	VERIFIED BY <i>KIMBA LAU</i> DATE <i>27th DEC. 1984</i>
DOCUMENT NO <i>STM-728-015</i>	VERSION NO <i>1.3</i> SHEET <i>1</i> OF <i>1</i>

7 6 5 4 3 2 1

7 6 5 4 3 2 1

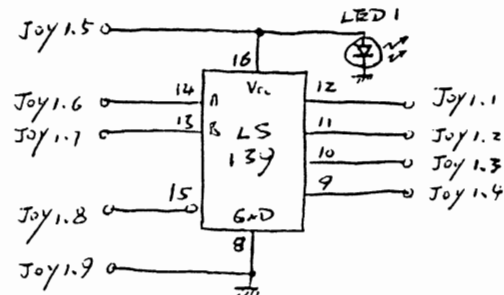
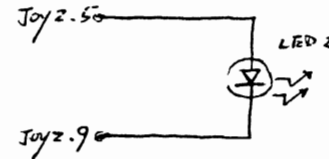
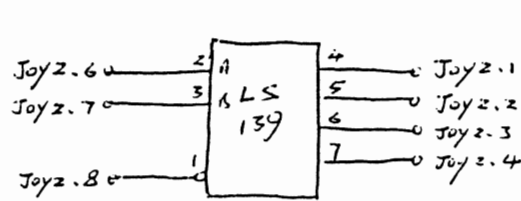
E  
D  
C  
B  
A



**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE <i>MSX 728 PCB LAYOUT (POWER AND VIDEO BOARD)</i>	VERIFIED BY <i>KIMBA LAU</i> DATE <i>27th DEC. 1984</i>
DOCUMENT NO <i>STM - 728 - Ø16</i>	VERSION NO <i>1.3</i> SHEET <i>1</i> OF <i>1</i>

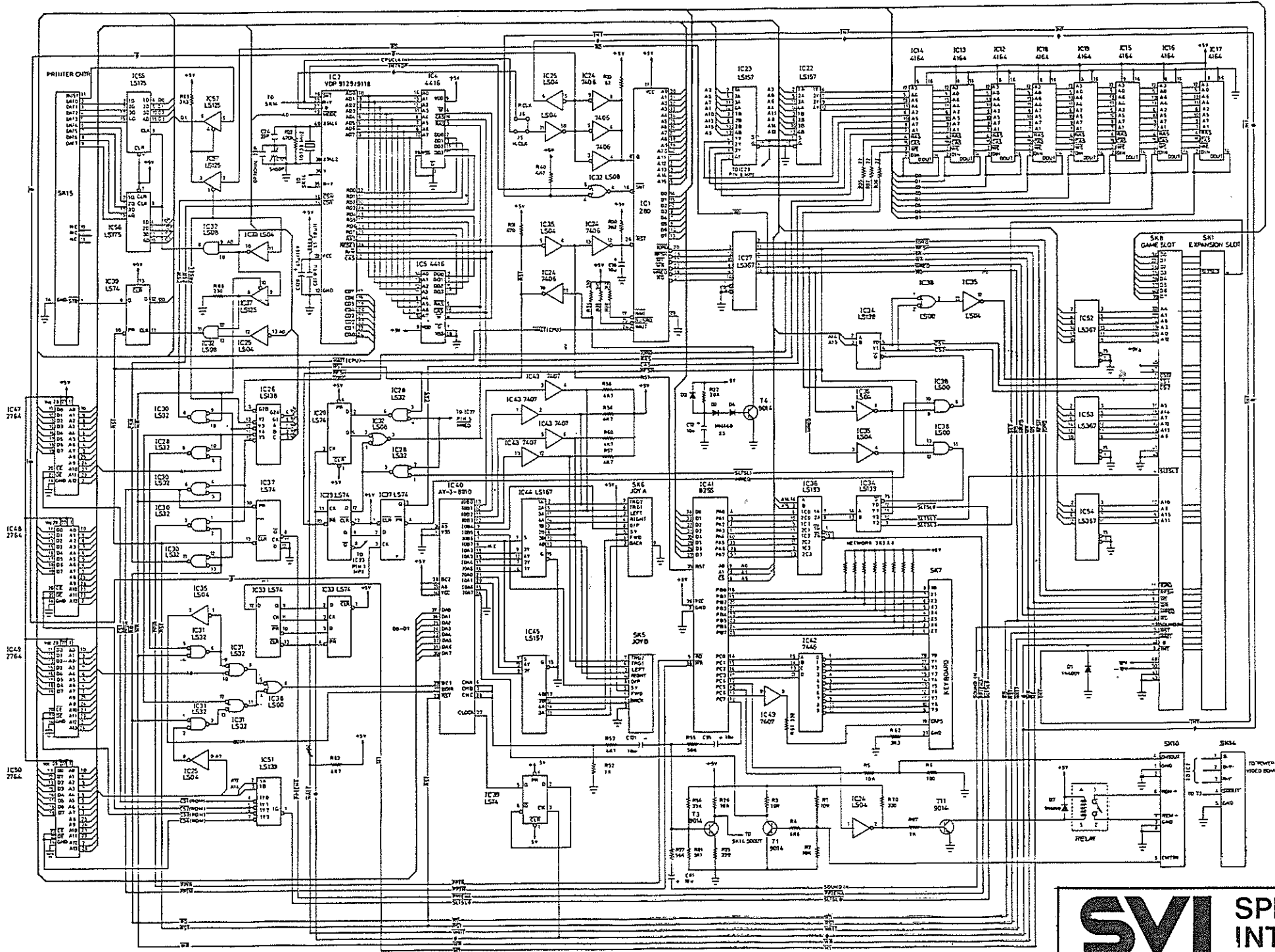
7 6 5 4 3 2 1



**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI</b> - SPECTRAVIDEO INTERNATIONAL LIMITED	
CONTROLLED ENGINEERING DOCUMENT THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE - SVI-728 TESTER JOYSTICK PLUG-IN	VERIFIED BY - KIMBA LAV DATE - 4th JAN 1985
DOCUMENT NO - STM-728-17	VERSION NO 1.0 SHEET 1 OF 1





THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

**SVI** SPECTRAVIDEO INTERNATIONAL LIMITED

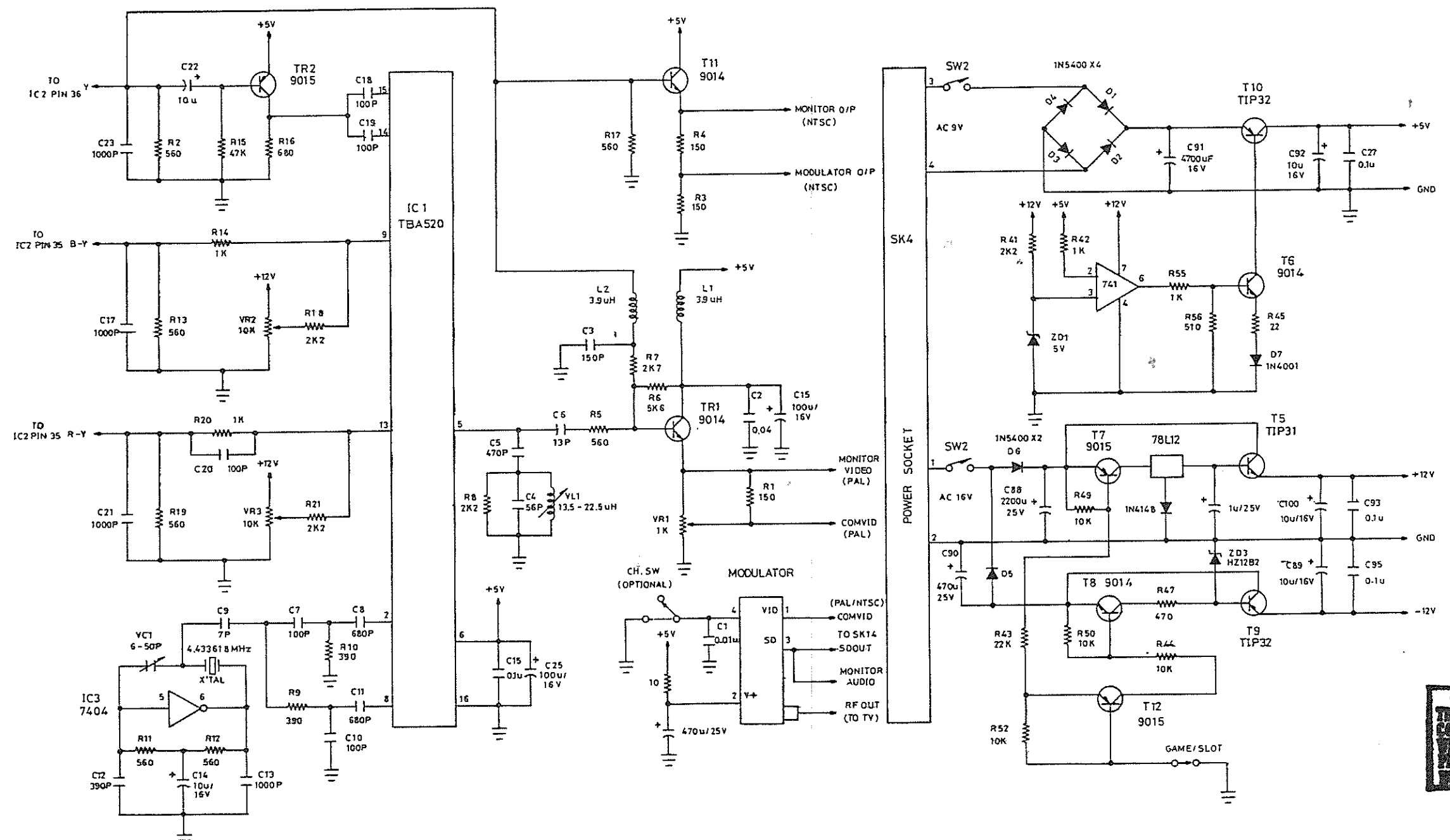
**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE *MSX 728 SCHEMATIC*  
 DIAGRAM (MAIN BOARD)  
 — EPROM VERSION —

VERIFIED BY *KIMBA LAU*  
 DATE *29-1-1985*

DOCUMENT NO *STM-728-018*

VERSION *1.0* SHEET *1*  
 NO OF *1*

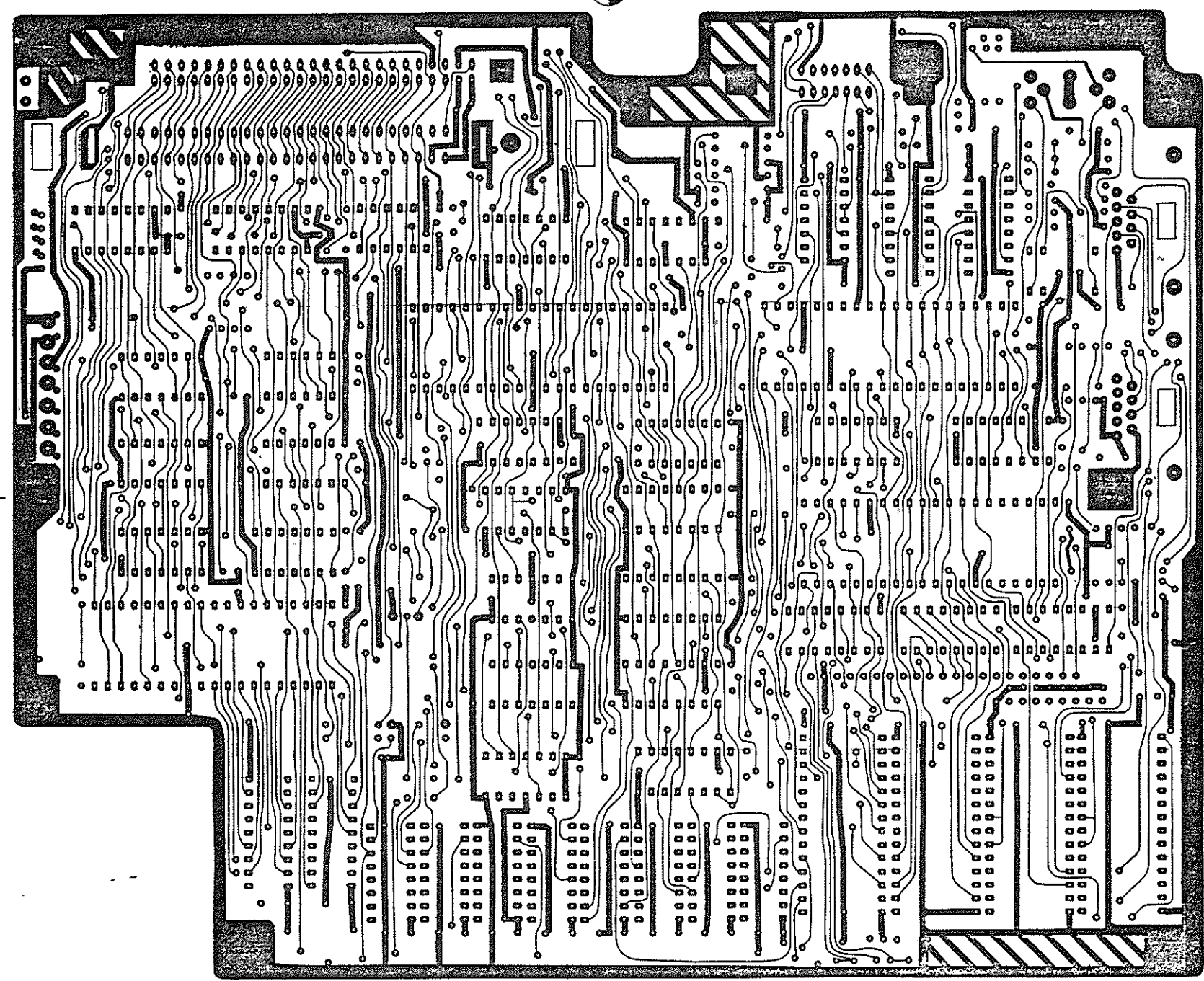


**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE <i>MSX 728 SCHEMATIC DIAGRAM (VIDEO AND POWER BOARD)</i>	VERIFIED BY <i>KIMBA LAU</i> DATE <i>29-1-85</i>
DOCUMENT NO <i>STM-728-019</i>	VERSION <i>1.0</i> SHEET <i>1</i> OF <i>1</i>

7 6 5 4 3 2 1

E  
D  
C  
B  
A



**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 SPECTRAVIDEO INTERNATIONAL  
 LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE *MSX 728 PCB*  
 LAYOUT (SOLDER SIDE)  
 — ROM VERSION —

VERIFIED BY *KIMBA LAU*  
 DATE *29-1-1985*

DOCUMENT NO *STM-728-020*

VERSION SHEET 1  
 NO *1-4* OF *1*

7 6 5 4 3 2 1

7

6

5

4

3

2

1

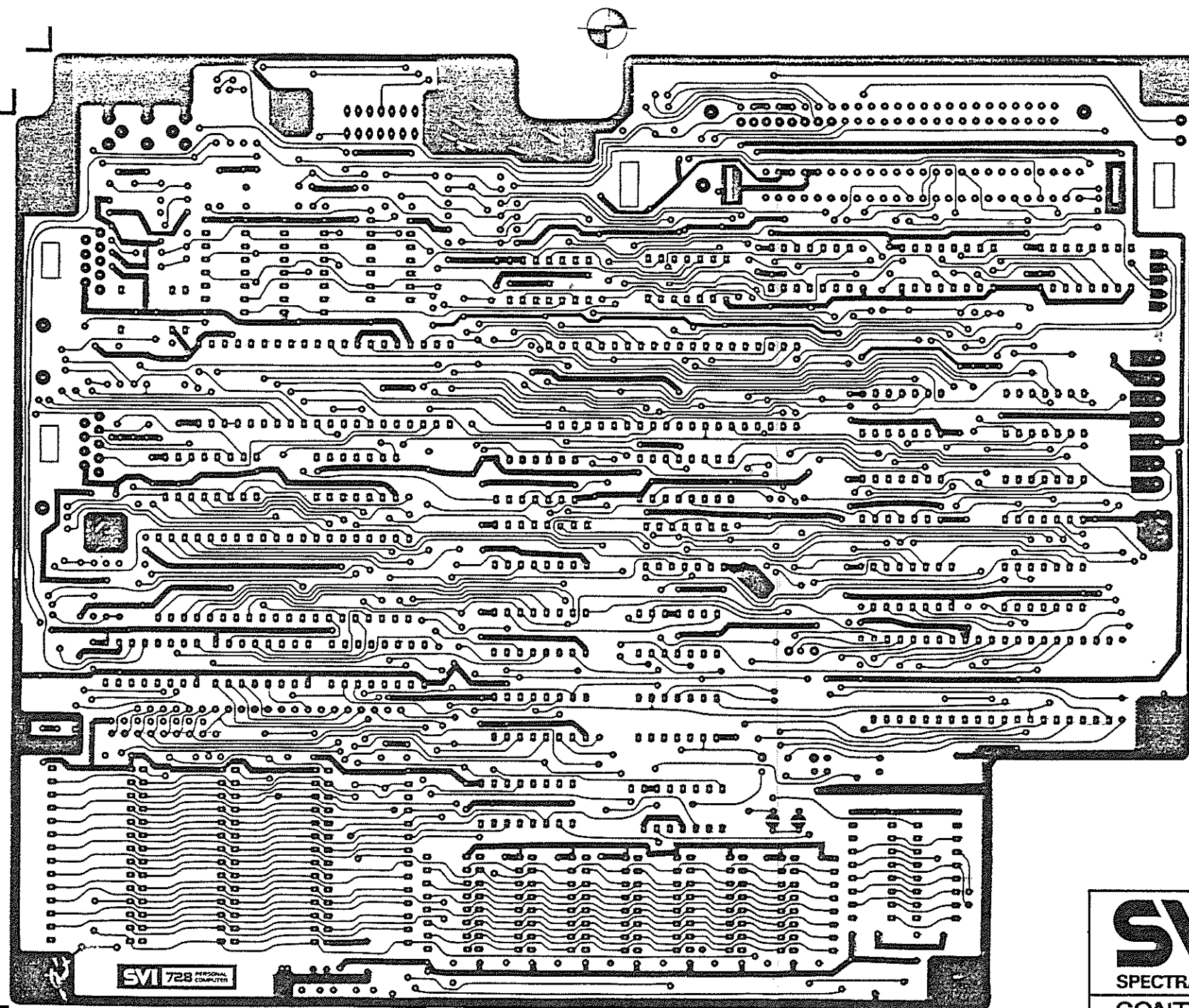
E

D

C

B

A



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 SPECTRAVIDEO INTERNATIONAL  
 LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE *MSX 728 PCB*  
*LAYOUT (COMPONENT SIDE)*  
 — ROM VERSION —

VERIFIED  
 BY *KIMBA LAU*  
 DATE \_\_\_\_\_

DOCUMENT  
 NO *STM-728-021*

VERSION SHEET /  
 NO *1.4* OF *1*

7

6

5

4

3

2

1

7

6

5

4

3

2

1

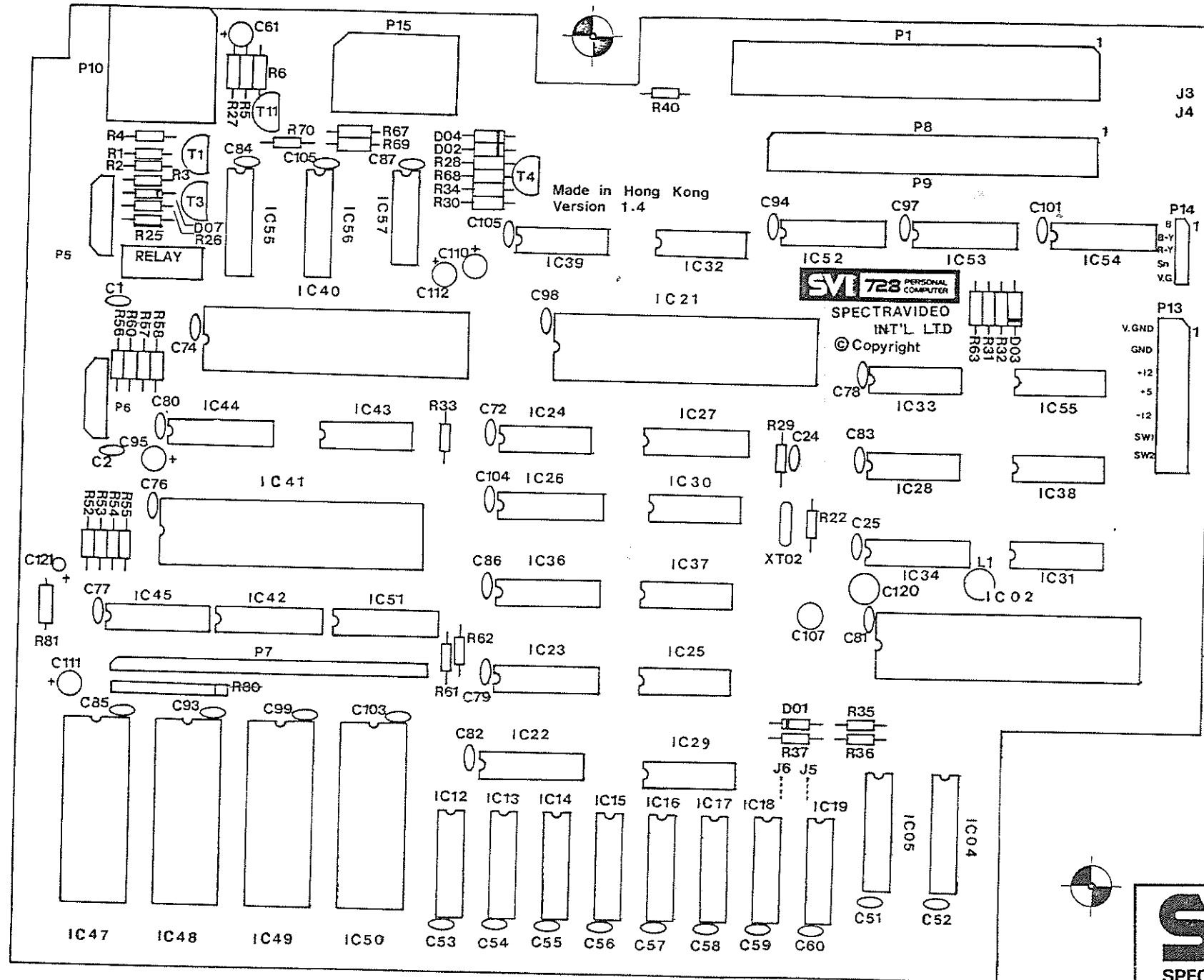
E

D

C

B

A



CONFIDENTIAL  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
INTERNATIONAL  
LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE <i>MSX 728 COMPONENT DRAWING (MAIN BOARD) — ROM VERSION —</i>	VERIFIED BY <u>KIMBA LAU</u> DATE <u>29-1-1985</u>
DOCUMENT NO <i>STM-728-022</i>	VERSION SHEET / NO <i>1.4</i> OF <i>1</i>

7

6

5

4

3

2

1

APPENDIX C

SOURCE CODE LISTING

-CP/M BIOS (2.24)

-MSX DOS

```

;*****
;
;      msx bios
;      for msx computer equipped with svi-707
;
;*****
;
;      1. support 80 column card ( auto-detected )
;      2. translate move cursor key to wordstar cursor movement
;      3. do not allow user to assign con: = ( 80-col. ) if not present
;      4. support 10 function keys:
;
;      console:
;      tty: - input keyboard,  output 80 column card
;      crt: - input keyboard,  output tv ( 39 column )
;      bat: - input rs232c,   output printer
;      ucl: - input keyboard,  output 80 column card
;
;      rdr:
;      - all input from rs232
;
;      pun:
;
;      - all output to rs232
;
;      list device:
;      tty: - output to 80 column card
;      crt: - output to tv ( 39 column )
;      lpt: - output to printer
;      ull: - output to 80 column card
;
;
;      .z80
;*****.
;      constants
;*****
;system
drive equ 0fffah ;location of drive bank
video equ 0ffffh ;location of 80 col-card bank
ram equ 0fffdh ;location of ram bank
mtime equ 0fffch ;motor timer counter
motron equ 0fffbh ;logical motor on / off
loby equ 0ffffh ;loby detected when cold boot
;console out
vdata equ 98h ;data register of vdp - 9129
vaddr equ 99h ;address latch of vdp - 9129
sndadr equ 0a0h ;address latch of sound gen.
sndwrt equ 0a1h ;data write of sound gen.
sndrd equ 0a2h ;data read of sound gen.
chaamp equ 8 ;channel a amplitude register
crtadr equ 78h ;address latch of 6845 crt controller
crtctr equ 79h ;data read / write of 6845
porta equ 0a8h ;i / o port a of 8255
portb equ 0a9h ;port b of 8255
portc equ 0aah ;port c of 8255
bell equ 7 ;sound bell
bs equ 8 ;back space
tab equ 9 ;tab.
lf equ 0ah ;line feed
chme equ 0bh ;cursor home
ff equ 0ch ;form feed
cr equ 0dh ;carriage return
esc equ 1bh ;escape
crlg equ 1ch ;cursor right
clft equ 1dh ;cursor left
cup equ 1eh ;cursor up
cdwn equ 1fh ;cursor down
curblk equ 0dbh ;cursor block code
;change to 0ffh and invert mode
;later
cxmin equ 20h ;80 column min. cursor x coordinate
cxmax equ cxmin+79 ;80 column max. cursor x coordinate
cymin equ 20h ;80 column min. cursor y coordinate
cymax equ cymin+23 ;80 column max. cursor y coordinate
cxmin4 equ cxmin ;40 column min. cursor x coordinate
cxmax4 equ cxmin+38 ;40 column max. cursor x coordinate
cymin4 equ cymin ;40 col. min. y
cymax4 equ cymin+23 ;40 col. max. y
lobyte equ 3 ;lobyte location of cp/m
screen equ 0b800h ;80 column 2k screen buffer starting
;bank should refer to register [video]
rdscn equ 0800h ;40 column 1k screen buffer for read
;note: this buffer is in video ram
wrscn equ rdscn+4000h ;40 column 1k screen buffer for write
;printer
prtpt equ 91h ;printer data output port
prtstb equ 90h ;printer data strobe port, write only
prtsor equ 90h ;printer busy read, bit 1
pbusy equ 1 ;printer busy bit
strobe equ 1 ;strobe data, i.e. bit 0

```

```

rsplst equ 0c00h ;printer spool buffer start for read
wsp1st equ rsp1st+4000h ;printer spool buffer start for write
rsplen equ 40h ;*100h = printer spool buffer end for read
wsplen equ rsplen+40h ;*100h = printer spool buffer end for write
;drive
drivea equ 9h ;on motor and sel. drive a
driveb equ 0an ;on motor and sel. drive b
onmotr equ 8 ;on motor
sidal equ 4 ;select side 1
reqpt equ 7f0ch ;data request port read only
;bit 7 = 1 for 1793 interrupt
;bit 6 = 0 for 1793 data request
selpt equ 7fbch ;drive select port
;bit 0 = 1 for select drive 0
;bit 1 = 1 for select drive 1
;bit 2 = 1 for select side 1
;bit 3 = 1 for motor on
compt equ 7fb8h ;command port of 1793 write only
sttpt equ compt ;status port of 1793 read only
trkpt equ 7fb9h ;track port of 1793
secpt equ 7fbah ;sector port of 1793
datapt equ 7fbbh ;data port of 1793
tcpm equ 7fbeh ;switch to go to cp/m
tmsx equ 7fbfh ;switch to go to msx-dos
secno equ 17 ;max sector for svi format
rserr equ 1ch ;read sector error mask
wserr equ 7ch ;write sector error mask
wterr equ 64h ;write track error mask
speed equ 0h ;speed of track to track access
;0 - 6ms step
;1 - 12ms step
;2 - 20ms step
;3 - 30ms step
restor equ 0h+speed ;command word for restore to track 0
stepin equ 54h+speed ;step in 1 track
seek equ 14h+speed ;seek required track
rsect equ 80h ;read sector command
wsect equ 0a0h ;write sector command
busy equ 0 ;disk controller processing command
drqb equ 1 ;data request
trk00 equ 2 ;track 0
skerr equ 4 ;seek record error
crcerr equ 3 ;crc check error
wprot equ 6 ;write protected
bios equ 0e600h ;bios start
ccp equ bios-1600h ;ccp start
bdos equ ccp+800h ;bdos start
tobdos equ 5 ;bdos call entry
;*****
; jump vector
;*****
jp boot
jp whoot
jp const
jp conin
jp conout
jp list
jp punch
jp reader
jp home
jp seldsk
jp settrk
jp setsec
jp setdma
jp read
jp write
jp listst
jp sectra
dw dphtbl ;point to disk parameter header table
dw dpötbl ;point to disk parameter block table
dw skytbl ;point to soft key table
db 20 ;no of sect to boot
dw bios ;loc to place bios
;
; disk parameter headers
;
dphtbl:

```



```

dpbase:
;drive A 318 format, double side
svadph:
    db      0,0          ;no sector translation table
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      dsvdpb      ;sv format double side
    dw      svcsv0     ;check vector
    dw      svalv0     ;allocation vector
;drive B 318 format, 1 or 2 side
svbdph:
    db      0,0
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      dsvdpb     ;sv format double side
    dw      svcsv1
    dw      svalv1
;drive B 328 single side
ssvdph:
    db      0,0
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      ssvdpb     ;sv format single side
    dw      ssvcsv
    dw      ssvvalv
;drive b osborne format
osbdph:
    db      0,0
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      osbdpb     ;osborne format
    dw      osbcsv
    dw      osbalv
;drive b kaypro format
kaydph:
    db      0,0
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      kaydph     ;kaypro format
    dw      kaycsv
    dw      kayalv
;drive b 12 format
s12dph:
    db      0,0
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      s12dph
    dw      s12csv
    dw      s12alv
;drive b 14 format
d14dph:
    db      0,0
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      d14dph
    dw      d14csv
    dw      d14alv
;ram disk
ramdph:
    db      0,0
    db      0,0
    db      0,0
    db      0,0
    dw      dirbuf
    dw      ramdph
    dw      ramcsv
    dw      ramalv

```

```

; 318 single side
;
; 256 bytes / sector
; 17 sector / track
; 40 track / disk
; 1 kbytes / block
; 3 reserved tracks
; 64 directory entries
;
dpbtbl:
savdpb: dw 34 ;sec/trk
        db 3 ;block shift
        db 7 ;block mask
        db 0 ;extent mask
        dw 156 ;disk size-1
        dw 63 ;directory max
        db 192 ;alloc0
        db 0 ;alloc1
        dw 16 ;check size
        dw 3 ;reserved trk
;
; msx double side
;
; 256 bytes / sector
; 17 sectors / track
; 80 tracks / disk
; 2 kbytes / block
; 3 reserved tracks
; 64 directory entries
;
dsvdpb: dw 34
        db 4
        db 15
        db 1
        dw 162
        dw 63
        db 80h
        db 0
        dw 16
        dw 3
;
; osborne disk drive
;
; 1 kbytes / sector
; 5 sector / track
; 40 tracks / disk
; 1 kbytes / block
; 2 reserved tracks
; 64 directory entries
;
osbdpb: dw 40
        db 3
        db 7
        db 0
        dw 184
        dw 63
        db 192
        db 0
        dw 16
        dw 3
;
; kaypro disk drive
;
; 512 bytes / sector
; 10 sector / track
; 40 tracks / disk
; 1 kbytes / block
; 1 reserved track
; 64 direcrory entries
;
kaydpb: dw 40
        db 3
        db 7
        db 0
        dw 194
        dw 63
        db 192
        db 0
        dw 16
        dw 1
;
; bondwell 12 format
;
; 256 bytes / sector
;
; 18 sector / track
; 40 track / disk
; 2 kbytes / block
; 2 reserved tracks
; 64 directory entries
;
s12dpb: dw 36
        db 4
        db 15
        db 1
        dw 84
        dw 127
        db 192
        db 0
        dw 32
        dw 2
;
; bondwell 14 format
;
; 256 bytes / sector
; 18 sectors / track
; 80 track / disk
; 2 kbytes / block
; 2 reserved tracks
; 64 directory entries
;
d14dpb: dw 36
        db 4
        db 15
        db 1
        dw 170
        dw 127
        db 192
        db 0
        dw 32
        dw 2
;
; ram disk
;
; 256 bytes / sector
; 16 sector / track
; 16 track / disk
; 1 kbytes / block
; 0 reserved track
; 32 directory entries
;
ramdpb: dw 32
        db 3
        db 7
        db 0
        dw 62
        dw 31
        db 80h
        db 0
        dw 0
        dw 0
;
; patch cpm to fit svi-318
;
; ds 31
;
; note: this part of program must be placed on memory
; starting from 0e715h.
; to display 40 / 80 column directries
;
patch1: push de ;see ccp.asm for details
        push bc ;save ccp's [bc]
        call getwid ;get width code either 1 or
        and b
        pop bc ;restore ccp's [bc].
        jp ccp+433h+80h ;return

```

```

;
patch2: push    af          ;see ccp.asm for details
        push    bc          ;save ccp' (bc)
        call   getwid       ;get the width code either 1 or 3
        cp     b
        pop    bc          ;restore ccp's (bc)
        jp     ccp+468h+80h ;and return
;
getwid: push    af          ;save input state
        ld     a,(ioby*e)   ;get iobyte
        and   3h           ;mask others except con:
        cp    1h           ;40 column crt: display ?
        jr    z,wexit      ;if so
        ld    a,3          ;all else are 80 column devices
wexit:  ld     b,a
        pop    af
        ret
;
hook1   equ    ccp+430h+80h ;loc jump to patch1
hook2   equ    ccp+465h+80h ;loc jump to patch2
;*****
;      soft keys
;*****
skytbl:
funkey: db     "DIR",cr,0,0,0,0
        db     0,0,0,0,0,0,0,0
        db     "STAT",0,0,0,0
        db     0,0,0,0,0,0,0,0
        db     "STAT *.*",cr
        db     0,0,0,0,0,0,0,0
        db     "LOADFKEY",cr
        db     0,0,0,0,0,0,0,0
        db     "COPY707",cr
        db     0,0,0,0,0,0,0,0,0
        db     "DIR A:",cr,0
        db     0,0,0,0,0,0,0,0,0
        db     "STAT ",cr,0,0
        db     0,0,0,0,0,0,0,0,0
        db     "STAT A:*.*",cr
        db     0,0,0,0,0
        db     "FORMAT",cr,0
        db     0,0,0,0,0,0,0,0,0
        db     "SYSGEN",cr,0
        db     0,0,0,0,0,0,0,0,0
;*****
;      interrupt routine .
;*****
intr:   push    af
        push    bc
        push    de
        push    hl
        in     a,(vaddr)   ;reset intr
; decrement counter for amplitude of bell
        ld     a,(sndamp)  ;load sound amplitude
        or    a
        jr    z,nosnd     ;skip if amplitude = 0
        dec   a           ;decrement sound amplitude
        ld    (sndamp),a  ;store sound amplitude
        ld    a,chaamp    ;point channel a amplitude
        out   (sndadr),a
        ld    a,(sndamp)  ;get channel a amplitude
        out   (sndwrt),a
nosnd:
; decrement counter for repeat key
        ld     a,(keycnt)  ;load repeat counter
        or    a
        jr    z,norpt     ;skip for no repeat key
        dec   a           ;decrement repeat counter
        ld    (keycnt),a  ;store repeat key counter
norpt:
; decrement disk drive motor counter
intakp: ld     a,(motron)  ;is motor logically turned on ?
        or    a
        jr    nz,inskp1   ;skip if logically on
        ld    a,(motime)  ;is motor physically off ?
        or    a
        jr    z,inskp1    ;skip if physically off
        dec   a           ;if not off, dec motor timer
        jr    nz,inskp2   ;skip if motor timer not equal 0
        call drion        ;if = 0 , turn on drive bank
        call offmot       ;turn off motor physically
        call drioff       ;swap off drive bank
        xor   a           ;reset motor timer
inskp2: ld     (motime),a  ;store motor timer

```

```

inaskpi:
; scan key and store in buffer
; so that conin or const can get it
    call    scan           ;scan keyboard physically
    or      a              ;any key is pressed ?
    jr      nz,press      ;branch to pressed
    ld      hl,newkey     ;no key pressed then,
    ld      de,oldkey     ;oldkey buffer = newkey buffer
    ld      bc,10         ;key buffer size = 10 bytes
    ldix
press:   ld      (prsfly),a ;if being pressed, set press flag
;check caps lock key
    ld      a,(ctlkey)
    and     08h           ;caps lock pressed ?
    jr      z,capl       ;skip if not pressed
    ld      a,(ctlky2)   ;get last caps key status
    and     08h           ;not pressed ?
    jr      nz,capl      ;skip for pressed
    ld      a,(capflg)   ;complement caps flag
    cpl
    ld      (capflg),a   ;if caps from not pressed to pressed
;show caps flag to led
capl:   ld      a,(capflg)
    or      a
    jr      nz,scap
rcap:   in      a,(portc) ;output = 1 for flag reset
    or      40h
    jr      scap1
scap:   in      a,(portc) ;output = 0 for set
    and     0bfh
scap1:  out     (portc),a
    ld      a,(ctlkey)
    ld      (ctlky2),a ;store present to last key status
;
; printer output
;
; move data from printer spooler buffer to printer
; max. move 7 characters
;
ptrout: ld      a,(msgrdy)
    or      a
    jr      z,ptrskp     ;check message in buffer
    in      a,(prtsor)
    bit    pbusy,a
    jr      nz,ptrskp    ;check printer busy
    ld      a,0ffh
    ld      (bufrdy),a   ;buffer ready for data
    ld      hl,(outptr) ;get spool out pointer
    ld      a,l
    out    (vaddr),a    ;point to buffer loc.
    ld      a,h
    out    (vaddr),a
    in      a,(vdata)   ;get message
    out    (prtpt),a
    call   dly00        ;delay for strobe data
    xor    a
    out    (prtstb),a   ;send strobe pulse
    call   dly00
    cpl
    out    (prtstb),a   ;reset strobe pulse
    inc   hl           ;inc pointer
    ld    a,h
    cp    rsplen      ;to vram end ?
    jr    nz,outrol
    ld    hl,rsplst
outrol: ld    (outptr),hl
    call  chkptr      ;all message printed ?
    ld    (msgrdy),a ;update message ready flag
    ld    a,140      ;set delay 700 usec for printer
ptrdly: dec   a
    jr    nz,ptrdly
    ld    a,(prtcnt) ;within 7 characters ?
    dec  a
    ld    (prtcnt),a
    jr    nz,ptrout   ;print another char.
ptrskp: ld    a,7
    ld    (prtcnt),a ;reset printer counter to 7
    pop  hl
    pop  de
    pop  bc
    pop  af

```

```

nop
nop
nop
ei
reti
;*****
;***** cold boot
;*****
;initial data for 40 column crt
setcrt: db 0b0h,81h,2,82h,0,84h,0f4h,87h
;initial data for sound generator
setsnd: db 0,80h,1,0,7,0feh,8,0
boot:
    di
    ld sp,ccp-2
    in a,(portc) ;off cap lamp
    or 40h
    out (portc),a
    ld a,(ram)
    out (porta),a ;64k
;
; initial sound generator for bell
;
    ld b,4 ;set counter
    ld hl,setsnd ;set pointing to init data
sets1: ld a,(hl) ;get register no
    out (sndadr),a ;output to latch
    inc hl ;inc pointer
    ld a,(hl) ;get init data
    out (sndwrt),a ;output to sound register
    inc hl ;inc pointer
    djnz sets1
;
; character buffer 0-7ff
; display buffer 800-bff
; 40 column display
;
    ld b,8 ;set counter
    ld hl,setcrt
setc1: ld a,(hl) ;set tv display format
    out (vaddr),a
    inc hl
    djnz setc1
    ld a,(loby)
    and 3h
    cp 1
    jr nz,card ;have 80 column card
;set no 80 column card flag
    ld (flag40),a ;flag 40 = 1
;
;in order to provide a 13k continous ram for printer spool buffer
;swap memory
;
    ld hl,800h ;read pattern
    ld de,4000h ;write pattern
    ld bc,800h ;counter
swapm: ld a,1 ;load charactor pattern
    out (vaddr),a ;from msx defined place
    ld a,h ;to more efficient loc.
    out (vaddr),a
    in a,(vdata)
    push af
    ld a,e ;store in new loc.
    out (vaddr),a
    ld a,d
    out (vaddr),a
    pop af
    out (vdata),a
    inc hl
    inc de
    dec bc
    ld a,b
    or c
    jr nz,swapm
;init ram card
    in a,(porta)
    ld (pbank4),a ;save present bank
    and 0c0h
    or 2ah ;switch to ram disk
    out (porta),a
;check svi - 747 present
    ld hl,07fe0h
    ld b,50 ;check 50 bytes

```

```

chkrdk: inc    hl
        dec    a
        ld    (hl),a
        cp    (hl)
        jr    nz,tocard    ;skip for no ram card
        djnz  chkrdk
        ld    a,1
        ld    (ramdok),a    ;set ram disk ok flag
;init svi - 747 directory
;fill directory with 0e5h --- no files
        ld    bc,1024    ;lk for 32 directory entries
        ld    hl,0
        ld    e,0e5h
iramdk: ld    (hl),e
        inc   hl
        dec   bc
        ld    a,b
        or    c
        jr    nz,iramdk
;copy page 4 memory access program to ram card
        ld    de,wrt41    ;get dest. pointer
        ld    hl,ovrlay    ;get source pointer
        ld    bc,endlay-ovrlay    ;set counter
        ldir
tocard: ld    a,(pbank4)
        out   (porta),a    ;recall present bank
card:   call  clrtxt    ;clear screen
;
;   after swap screen memory enable display
;
        ld    a,0f0h
        out   (vaddr),a
        ld    a,081h    ;point to display reg.
        out   (vaddr),a
;
;   check for second drive
;
        call  drion    ;switch on drive bank
        ld    a,(reqpt)    ;get request port
        and   20h    ;any 2nd drive ?
        ld    (sngdri),a    ;set single drive flag
        call  drioff    ;off drive bank
;
;   load ccp and bdos
;
        call  botdos    ;load bdos and ccp
        ld    a,(ioby)    ;default 80col and 1pt
        ld    (iobyte),a
        ld    c,9    ;display header
        ld    de,header
        call  tobdos
        ld    a,(iobyte)
        push af
        ld    a,1    ;set to crt
        ld    (iobyte),a
        ld    c,9    ;display header for tv
        ld    de,header
        call  tobdos
        pop   af
        ld    (iobyte),a    ;recall iobyte
        ld    a,0    ;select drive a
        ld    (4h),a    ;store in current drive
        ld    c,a    ;to ccp
        jp    ccp
header: db    ff
        db    'SPECTRAVIDEO CP/M-80 '
        db    'Revision 2.24',cr,lf
        db    'for SVI-707 ( release 1.0 )',cr,lf
        db    'Copyright (c) by Digital '
        db    'Research',cr,lf
        db    lf
        db    '$'
;*****
;   warm boot
;*****
wboot:  di
        ld    sp,ccp-2
        call  botdos    ;get ccp and bdos
tccp:   ld    a,(4h)    ;get current drive
        ld    c,a
        jp    ccp
;*****
;   boot bdos
;*****

```

```

botdos: ld      a,0c3h          ;interrupt handler
        ld      (038h),a
        ld      hl,intr
        ld      (039h),hl
        ld      (038h),a
        ld      a,22          ;22 sectors for ccp and bdos
        ld      (seccnt),a
        ld      a,1          ;start from track 1 sector 12
        ld      (hsttrk),a
        ld      a,12
        ld      (hstsec),a
        call    drion        ;switch to drive bank
;default to drive a:
        ld      a,(hstdsk)
        push   af
        xor    a
        ld      (hstdsk),a   ;the current drive is saved
        call   onmot        ;turn motor
        call   restr        ;restore to track 0
        call   drioff       ;off drive bank
        ld      de,ccp      ;22 sector data load to loc. statt ccp
botd1:  push   de
        call   readhst
        pop    de
        ld      hl,(hstbuf)
        ld      bc,256
        ldir
        ld      a,(seccnt)
        dec    a            ;finish ?
        ld      (seccnt),a
        jr     z,botd2
        ld      a,(hstsec)
        inc    a            ;inc sector number
        ld      (hstsec),a
        cp    17
        jr     nz,botd1
        ld      a,0
        ld      (hstsec),a
        ld      a,(hsttrk)
        inc    a            ;inc track no
        ld      (hsttrk),a
        jr     botd1
botd2:  pop    af          ;recall current drive
;
        init  jump vectors
        ld      a,0c3h      ;jump
        ld      (0h),a     ;warm boot jump
        ld      (5h),a     ;bdos jump
        ld      (hook1),a  ;cpm patch1 jump
        ld      (hook2),a  ;cpm patch2 jump
        ld      hl,bios+3
        ld      (1h),hl
        ld      hl,bdos+6
        ld      (6h),hl
        ld      hl,patch1
        ld      (hook1+1),hl
        ld      hl,patch2
        ld      (hook2+1),hl
        xor    a
        ld      (hstact),a  ;init deblocking flags
        ld      (unacnt),a
        ret
;*****
; console status
;*****
;
; con: iobyte bit 0,1
; 0 --- keyboard
; 1 --- keyboard
; 2 --- rs232
; 3 --- keyboard
;
const: di
        call   consti
        ei
        nop
        di
        ret

```

```

const1:  ld      a,(iobyte)
         and    3h          ;get con: assign
         cp    2           ;bat ?
         jp    z,batst
         ld    a,(funflg) ;displaying function key ?
         or    a
         jr    nz,keyin    ;if function key then key is ready
         ld    a,(prsf1g) ;pressed ?
         or    a
         jr    z,nokey
         call  cmpkey      ;same key ?
         cp    0ffh
         jr    nz,keyin
         ld    a,(keycnt) ;time for repeat key ?
         or    a
         jr    z,keyin
nokey:   ld    a,0
         jr    cstok
keyin:   ld    a,0ffh
cstok:   ret
batst:   ret
;*****
; console input
;*****
conin:   di
         call  conin1
         ei
         nop
         di
         ret
conin1:  ld    a,(iobyte)
         and    3h          ;get con:
         cp    2h          ;bat: input ?
         jp    z,batin
         ld    a,(funflg) ;if function key, get function key
         or    a
         jp    nz,funct1
input2:  ei                ;if no key input
         nop              ;wait until intr. scan a key
         nop
         di
         call  chkprb     ;check any key pressed
         or    a
         jr    z,input2
         call  cmpkey     ;changed ?
         cp    0ffh
         jr    nz,fkey
         ld    a,(keycnt) ;repeat key count time up ?
         or    a
         jr    nz,input2
         ld    a,6         ;reset repeat key count = 6
         ld    (keycnt),a
         ld    hl,newkey-1 ;store key
         ld    c,10
rptkey:  inc    hl
         dec    c
         ld    a,(hl)
         or    a
         jr    z,rptkey
         sla  c            ;get no of the key
         sla  c            ;in order to get ascii from table
         sla  c            ;*8
rkey1:   rr    a
         jr    c,fkey1
         inc  c
         jr    rkey1
fkey:    ld    c,a
         ld    a,40        ;reset key count to 40 for not press
         ld    (keycnt),a
fkey1:   ld    a,(ctlkey)
         and   03h
         jr    z,fkey2a
         and   02h
         jr    z,fkey3
         add  a,78         ;ctrl
fkey3:   add  a,80         ;shift
fkey2:   add  a,c

```





```

        ld      a,(newkey+3)
        ld      (ctrlkey),a      ;save ctrl key in control key buffer
        and     0e0h             ;clear ctrl keys
chkprs: ld      (newkey+3),a
        ld      hl,newkey
        ld      b,10
scan4:  ld      a,(hl)          ;check any key pressed ?
        or      a
        ret     nz              ;pressed, return with accm non-zero
        inc     hl
        djnz   scan4           ;loop checking
        ret                     ;not pressed, return with zero accm
;*****
; compare keys
; check if newkey = oldkey
;*****
cmpkey: ld      hl,newkey       ;point to newkey
        ld      de,oldkey      ;point to oldkey
        ld      c,9
cmpkyl: ld      a,(de)
        xor     (hl)           ;changed ?
        jr     z,smkey
        and     (hl)           ;which bit have change
        jr     nz,getkl        ;branch if change from 0 to 1
smkey:  inc     hl              ;check other keys
        inc     de
        dec     c
        jp     p,cmpkyl        ;loop checking
        dec     hl             ;load new keys
        dec     de             ;if reset to '0'
        ld      bc,10          ;copy newkey to old key if nokey
        lddr   a,c
        ld      c,0ffh        ;return with a = 0ffh for no key
cmpok:  ld      a,c             ;same input
        ret
getkl:  sla     c               ;get no of the key
        sla     c               ;c * 8 + log2 a
        sla     c
getkey: rr      a
        jr     c,cmpok
        inc     c
        jr     getkey
;*****
; key table
;*****
;the keyboard matrix is shown below
;small letter table
keytbl: db      '01234567'
        db      '89-=' ,5ch,'[];'
        db      '27h,60h','./ ab'
        db      'cdefghij'
        db      'klmnopqr'
        db      'stuvwxyz'
        db      ' ',128,129,130
        db      131,132,esc,tab,20h,bs,20h,cr
        db      20h,chs,20h,7fh,19,5,24,4
        db      '+-*/'
;block letter table
keysht: db      ')!@#$%' ,5ch,'&'
        db      '*(' ,5fh,'+',7ch,'=: '
        db      '"',7ch,'$%? AB'
        db      'CDEFGHIJ'
        db      'KLMNOPQR'
        db      'STUVWXYZ'
        db      ' ',133,134,135
        db      136,137,esc,tab,20h,bs,20h,cr
        db      20h,ff,20h,7fh,19,5,24,4
        db      '+-*/'
;control key table
keyctl: db      '01234567'
        db      '89-=' ,28,27,29,';'
        db      '27h,60h','./ ',1,2
        db      3,4,5,6,7,8,9,0ah
        db      0bh,0ch,0dh,0eh,0fh,10h,11h,12h
        db      13h,14h,15h,16h,17h,18h,19h,1ah
        db      ' ',128,129,130
        db      131,132,esc,tab,20h,bs,20h,cr
        db      20h,chs,20h,7fh,19,5,24,4
        db      '+-*/'

```

```

;*****
; console output routine
;*****
;
; iobyte 0,1
; 0 --- 80 column
; 1 --- 40 column
; 2 --- list device
; 3 --- 80 column
;
conout:
di
ld (stkbuf),sp
ld sp,blostk
call coutl
ld sp,(stkbuf)
ei
nop
di
ret

;
coutl: ld a,(iobyte) ;check iobyte for 40/80 column
and 3h
cp 1h
jp z,dsp40 ;=1 for 40 col, others are 80 col
cp 2h
jp z,listl ;to list device
ld a,(flag40) ;80 column card installed ?
or a
jp z,dsp80

dsp40: ;40 column for no 80 column card
;erase cursor
call xytoc4 ;change x-y coordinate to buffer pointer
ld de,wrscn ;point to write screen
add hl,de
ld a,l
out (vaddr),a
ld a,h
out (vaddr),a
ld a,(curbuf) ;recall the hided character
out (vdata),a
ld a,(escflg) ;in escape sequence ?
or a
jp nz,escse4
ld a,c
cp ' ' ;= control keys ?
jr c,ctlse4
call outvi4 ;output if normal ascii code
call curfw4 ;advance cursor

dspok4:
nodsp4: xor a ;reset all reset status and flags
ld (escflg),a
ld (esccnt),a

dspok4: call xytoc4 ;write cursor back
ld de,rdscn
add hl,de
ld a,l
out (vaddr),a
ld a,h
out (vaddr),a
in a,(vdata)
ld (curbuf),a ;save the character behind the cursor
ld de,wrscn-rdscn
add hl,de
ld a,l
out (vaddr),a
ld a,h
out (vaddr),a ;write the cursor block
ld a,curblk
out (vdata),a
ret

;*****
; control sequence 40 col
;*****
ctlse4: cp bell ;sound bell
jp z,bell40
cp bs ;back space
jr z,bs40
cp tab ;tab
jr z,tab40
cp lf ;line feed
jr z,lf40
cp chme ;cursor home
jr z,chme40
cp ff ;form feed

```

```

jr      z,ff40
cp      cr
jr      z,cr40      ;carriage return
cp      esc
jr      z,esc40    ;enter escape esquence
cp      crig
jp      z,crig40   ;cursor right
cp      clft
jp      z,clft40  ;cursor left
cp      cup
jr      z,cup40   ;cursor up
cp      cdwn
jp      z,cdwn40 ;cursor down
jr      dpsk4

;*****
;      bell
;*****
bell40: call  sndbel      ;sound the bell
jr      nodsp4
;sound the bell routine
sndbel: ld   a,l0h
ld      (sndamp),a
ret

;*****
;      back space
;*****
bs40:   call  curbs4
ld      c,' '
call    outvi4
jr      tds40

;*****
;      tab
;*****
tab40:  call  curfw4
ld      a,(curx)
and     7
jr      nz,tab40
jr      tds40

;*****
;      line feed
;*****
lf40:   ld   a,(cury)
cp      cymax4
jr      nz,lf41
call    scrol4
jr      tds40
lf41:   inc  a
ld      (cury),a
jr      tds40

;*****
;      cursor home
;*****
chme40: ld   a,cymin4
ld      (cury),a

;*****
;      carriage return
;*****
cr40:   ld   a,cxmin4
ld      (curx),a
jr      tds40

;*****
;      form feed
;*****
ff40:   call  clrtxt
jr      chme40

;*****
;      clear screen
;*****
clrtxt: ld   de,wrscon
ld      bc,40*25
clr1:  ld   a,e
out     (vaddr),a
ld      a,d
out     (vaddr),a
ld      a,' '
out     (vdata),a
inc     de
dec     bc
ld      a,b
or      c
jr      nz,clr1
ret

```

```

;*****
; enter esc seq
;*****
esc40: ld a,1
      ld (escflg),a
      jp dspk4
;*****
; cursor up
;*****
cup40: ld a,(cury)
      cp cymin4
      jr z,tdspo4
      dec a
      jr curdn4
;*****
; cursor down
;*****
cdwn40: ld a,(cury)
      cp cymax4
      jr z,tdspo4
      inc a
      jr curdn4
;*****
; cursor left
;*****
clft40: ld a,(curx)
      cp cxmin4
      jr z,tdspo4
      dec a
      jr curig4
;*****
; cursor right
;*****
crig40: ld a,(curx)
      cp cxmax4
      jr z,tdspo4
      inc a
      jr curig4
;*****
; cursor backspace
;*****
curbs4: ld a,(curx)
      cp cxmin4
      jr z,curb41
      dec a
      jr curb42
;*****
curb41: ld a,(cury)
      cp cymin4
      ret z
      dec a
      ld (cury),a
      ld a,cxmax4
;*****
curb42: ld (curx),a
      ret
;*****
; cursor forward
;*****
curfw4: ld a,(curx)
      cp cxmax4
      jr z,curf41
      inc a
      jr curf42
;*****
curf41: ld a,(cury)
      cp cymax4
      jr z,curf43
      inc a
      ld (cury),a
      ld a,cxmin4
;*****
curf42: ld (curx),a
      ret
;*****
curf43: call scrol4
      ld a,cxmin4
      ld (curx),a
      ret
;*****
; scroll
;*****
scrol4: ld hl,rdscn+40
      ld de,wrscln
      ld bc,40*24
;*****
scro41: ld a,1
      out (vaddr),a
      ld a,h
      out (vaddr),a
      in a,(98h)
      push af
      ld a,e
      out (vaddr),a
      ld a,d
      out (vaddr),a
      pop af
      out (vdata),a
      inc hl
      inc de
      dec bc
      ld a,b
      or c
      jr nz,scro41
      ret
;*****
; output to video
; the character in c reg.
; is output screen pointed by x-y
; where x-y is in curx and cury
;*****
outvi4: call xytoc4
      ld de,wrscln
      add hl,de
      ld a,1
      out (vaddr),a
      ld a,h
      out (vaddr),a
      ld a,c
      out (vdata),a
      ret
;*****
; x-y to continue
; the coordinate stored in
; curx, cury is converted
; to a pointer pointing to
; the corresponding screen
; location
;*****
xytoc4: ld a,(cury)
      sub cymin4
      ld hl,0
      ld de,40
      cp 0
      jr z,xyto42
;*****
xyto41: add hl,de
      dec a
      jr nz,xyto41
;*****
xyto42: ld a,(curx)
      sub cxmin4-1
      ld e,a
      ld d,0
      add hl,de
      ret
;*****
; esc sequence
;*****
escse4: ld a,(escclnt) ;check escape status
      cp 1 ;set y coordinate
      jp z,saty4
      cp 2 ;set x coordinate
      jp z,setw4
      ld a,c
      cp 'j' ;form feed
      jp z,ff40
      cp 'E' ;form feed
      jp z,ff40
      cp 'y' ;start locating cursor
      jp z,locu40
      cp 'A' ;cursor up
      jp z,cup40
      cp 'B' ;cursor down
      jp z,cdwn40
      cp 'C' ;cursor right
      jp z,crig40
      cp 'D' ;cursor left
      jp z,clft40
      cp 'H' ;cursor home
      jp z,chme40
      jp dspok4

```

```

;*****
; locate cursor
;*****
locu40: ld      a,1
        ld      (escCnt),a
        jr      tDpsk4
sety4:  ld      a,c
        cp      cymin4
        jr      nc,sety41
        ld      a,cymin4
sety41: cp      cymax4+1
        jr      c,sety42
        ld      a,cymax4
sety42: ld      (cury),a
        ld      a,2
        ld      (escCnt),a
tDpsk4: jp      dSpSk4
setx4:  ld      a,c
        cp      cxmin4
        jr      nc,setx41
        ld      a,cxmin4
setx41: cp      cxmax4+1
        jr      c,setx42
        ld      a,cxmax4
setx42: ld      (curx),a
        jp      dSpSk4
;*****
; 80 column display
;*****
dSpSk0: ld      a,(modflg)
        or      a
        jr      nz,srmode
        ld      a,(escflg)
        or      a
        jp      nz,escseq      ;esc sequence
        ld      a,c
        cp      ' '
        jr      c,ctlseq
        ld      a,(invert)
        or      a
        jr      z,ninv
        ld      a,c
        or      80h
        ld      c,a
ninv:   call    outvid
        call    curfw
dSpSk:  call    poscur
nodSp:  xor     a
        ld      (escflg),a
        ld      (escCnt),a
dSpSkp: ret
srmode: xor     a
        ld      (modflg),a
        ret
;*****
; control sequence
;*****
ctlSeq: cp      bell      ;sound bell
        jr      z,bell0
        cp      bs        ;back space
        jr      z,bs0
        cp      tab       ;tab
        jr      z,tab0
        cp      lf        ;line feed
        jr      z,lf0
        cp      chme      ;cursor home
        jr      z,chme0
        cp      ff        ;form feed
        jr      z,ff0
        cp      cr        ;carriage return
        jr      z,cr0
        cp      esc       ;enter escape sequence
        jr      z,esc0
        cp      crlg      ;cursor right
        jr      z,crlg0
        cp      clft      ;cursor left
        jr      z,clft0
        cp      cup       ;cursor up
        jr      z,cup0
        cp      cdwn      ;cursor down
        jr      z,cdwn0
        jr      dSpSkp

```

```

;*****
; bell
;*****
bell0:  call    sndbel      ;sound the bell
        jr      nodsp
;*****
; back space
;*****
bs0:    call    curbs
        ld      c,' '
        call    outvid
        jr      tDpsk4
;*****
; tab
;*****
tab0:   call    curfw
        ld      a,(curx)
        and    7
        jr      nz,tab0
        jr      tDpsk4
;*****
; line feed
;*****
lf0:    ld      a,(cury)
        cp      cymax
        jr      nz,lf1
        call    scroll
        jr      tDpsk4
lf1:    inc     a
        ld      (cury),a
        jr      tDpsk4
;*****
; cursor home
;*****
chme0:  ld      a,cymin
        ld      (cury),a
;*****
; carriage return
;*****
cr0:    ld      a,cxmin
        ld      (curx),a
        jr      tDpsk4
;*****
; form feed
;*****
ff0:    call    scnOn
        ld      a,' '
        ld      (screen),a
        ld      hl,screen
        ld      de,screen+1
        ld      bc,80*25
        ldLr
        call    scnOff
        jr      chme0
;*****
; enter esc seq
;*****
esc0:   ld      a,1
        ld      (escflg),a
        jp      dSpSkp
;*****
; cursor up

```

```

;*****
cup0: ld a,(cury)
      cp cymin
      jr z,tdspok
      dec a
      jr curdn1
;*****
; cursor down
;*****
cdwn0: ld a,(cury)
      cp cymax
      jr z,tdspok
      inc a
curdn1: ld a,(cury),a
      jr tdspok
;*****
; cursor left
;*****
clft0: ld a,(curx)
      cp cxmin
      jr z,tdspok
      dec a
      jr curigl
;*****
; cursor right
;*****
orig0: ld a,(curx)
      cp cxmax
      jr z,tdspok
      inc a
curigl: ld a,(curx),a
      jr dspok
;*****
; cursor backspace
;*****
curbs: ld a,(curx)
      cp cxmin
      jr z,curbs1
      dec a
      jr curbs2
curbs1: ld a,(cury)
      cp cymin
      ret z
      dec a
      ld a,(cury),a
      ld a,cxmax
curbs2: ld a,(curx),a
      ret
;*****
; cursor forward
;*****
curfw: ld a,(curx)
      cp cxmax
      jr z,curfw1
      inc a
      jr curfw2
curfw1: ld a,(cury)
      cp cymax
      jr z,curfw3
      inc a
      ld a,(cury),a
      ld a,cxmin
curfw2: ld a,(curx),a
      ret
curfw3: call scroll
      ld a,cxmin
      ld a,(curx),a
      ret
;*****
; scroll
;*****
;*****
scroll: call scnon ;turn on screen bank
        ld hl,screen+80 ;set pointers for scroll
        ld de,screen
        ld bc,80*24 ;scroll whole page
        ldir
        call scnoff ;turn off screen bank
        ret
;*****
; screen on
; turn on screen bank
;*****
scnon: in a,(porta)
        ld (pbank1),a ;save present bank
        and 0cfh
        ld b,a
        ld a,(video) ;get video bank
        and 30h
        or b
        out (porta),a
        ret
;*****
; screen off
;*****
scnoff: ld a,(pbank1) ;get saved memory bank
        out (porta),a
        ret
;*****
; output to video
; data in c reg. to loc pointed by curx cury
;*****
outvid: call scnon ;turn on screen bank
        call xytoc ;point to x-y coordinate pointed loc
        ld de,screen
        add hl,de
        ld (hl),c ;put data
        call scnoff ;turn off screen
        ret
;*****
; x-y to continue
; convert x-y ( curx cury )
; to pointer pointing screen
; stored in hl
;*****
xytoc: ld a,(cury)
        sub cymin
        ld hl,0
        ld de,80
        cp 0
        jr z,xytoc2
        jr z,xytoc1
xytoc1: add hl,de
        dec a
        jr nz,xytoc1
xytoc2: ld a,(curx)
        sub cxmin
        ld e,a
        ld d,0
        add hl,de
        ret
;*****
; position cursor
; move cursor to position
; specified by curx cury
;*****
poscur: call xytoc
        ld a,14
        out (crtadr),a
        ld a,h
        out (crtctr),a
        ld a,15
        out (crtadr),a
        ld a,l
        out (crtctr),a
        ret
;*****
; escape sequence
;*****

```

```

escseq: ld      a,(esccht)
        cp      1
        jp      z,sety      ;set y coordinate
        cp      2
        jp      z,setx      ;set x coordinate
        ld      a,c
        cp      'j'        ;form feed
        jp      z,ff0
        cp      'E'        ;form feed
        jp      z,ff0
        cp      'K'        ;erase to end of line
        jr      z,etoel0
        cp      'J'        ;erase to end of page
        jr      z,etoep0
        cp      'l'        ;erase present line
        jp      z,erall0
        cp      'L'        ;add in one line
        jp      z,addl10
        cp      'M'        ;delete one line
        jp      z,dell10
        cp      'Y'        ;locate cursor
        jp      z,locur0
        cp      'A'        ;cursor up
        jp      z,cup0
        cp      'B'        ;cursor down
        jp      z,cdwn0
        cp      'C'        ;cursor right
        jp      z,crig0
        cp      'D'        ;cursor left
        jp      z,clft0
        cp      'H'        ;cursor home
        jp      z,chme0
        cp      'p'        ;set reverse display mode
        jp      z,srev0
        cp      'q'        ;reset reverse display mode
        jp      z,rsrev0
        cp      'x'        ;set block cursor
        jp      z,smode0
        cp      'y'        ;set line cursor
        jp      z,rmode0
        jr      todspk

;*****
;          erase to end of line
;*****
etoel0: ld      a,(curx)
        push    af          ;save cursor x-coordinate
        ld      c,' '      ;load blank
etoel1: call    outvid      ;output to screen
        ld      a,(curx)
        inc     a
        cp     cxmax+1
        ld     (curx),a
        jr     nz,etoel1    ;loop until end of line
        pop    af          ;recall cursor x-coordinate
        ld     (curx),a
        jr     todspk

;*****
;          erase to end of page
;*****
etoep0: call    scnon       ;turn on screen
        call    xytoc
        ld     de,screen
        add    hl,de
        ld     c,' '
etoep1: ld     (hl),c      ;blank to screen
        inc    hl
        ld     a,h
        cp     0bfh
        jr     nz,etoep1    ;loop until end of page
        ld     a,l
        cp     080h
        jr     nz,etoep1
        call   scnoff      ;off screen
        jr     todspk

;*****
;          erase entire line
;*****
erall0: ld     a,(curx)
        push    af
        ld     a,cxmin
        ld     (curx),a
        call   scnon

```



```

erall2: call    xytoc
        ld     de,screen
        add   hl,de
        ld   c,
        ld   b,80
erall1: ld     (hl),c
        inc  hl
        djnz erall1
erall3: call    scnofff
        pop  af
        ld  (curx),a
todspk: jp     dspok
;*****
; locate cursor
;*****
locur0: ld     a,1
        ld     (escCnt),a
        jr    tdpskp
sety:   ld     a,c
        cp    cymIn
        jr    nc,sety1
        ld   a,cymIn
        cp    cymax+1
        jr    c,sety2
        ld   a,cymax
sety2:  ld     (cury),a
        ld   a,2
        ld     (escCnt),a
tdpskp: jp     dspok
setx:   ld     a,c
        cp    cxmIn
        jr    nc,setx1
        ld   a,cxmIn
        cp    cxmax+1
        jr    c,setx2
        ld   a,cxmax
setx2:  ld     (curx),a
        jr    todspk
;*****
; set invert
;*****
srav0:  ld     a,1
        jr    rsrev1
;*****
; reset invert
;*****
rsrev0: ld     a,0
rsrev1: ld     (invert),a
        jr    todspk
;*****
; set mode
;*****
smode0: ld     a,10
        out   (crtadr),a
        ld   a,060h
        jr    rmode1
;*****
; reset mode
;*****
rmode0: ld     a,10
        out   (crtadr),a
        ld   a,068h
rmode1: out   (crtctr),a
        ld   a,1
        ld   (modfig),a
        jr    todspk
;*****
; insert one blank line
; lines above inserted line
; scroll up
;*****
add110: ld     a,(curx)
        push  af
        ld   a,cxmin
        ld   (curx),a
        call scnon
        ld   a,(cury)
        cp   cymax
        jp   z,erall2
        call xytoc
        push hl
        pop  bc
        scf
        ld   hl,780h-79
        sbc  hl,bc
        push hl
        pop  bc
        ld   hl,screen+780h-81
        ld   de,screen+780h-1
        lddr
        jp   erall2
;*****
; delete one line
; lines below deleted line
; scroll up
;*****
del110: ld     a,(curx)
        push  af
        ld   a,cxmin
        ld   (curx),a
        call scnon
        ld   a,(cury)
        cp   cymax
        jp   z,erall3
        call xytoc
        ld   bc,screen
        add  hl,bc
        push hl
        ld   bc,80
        add  hl,bc
        push hl
        push hl
        pop  bc
        ld   hl,screen+780h+81
        scf
        sbc  hl,bc
        push hl
        pop  bc
        pop  hl
        pop  de
        ldir
        jp   erall3
;*****
; list program
;*****
;
; iobyte bit 6,7
; 00h --- 80col
; 40h --- 40col
; 80h --- printer
; 0c0h --- rs232 output
;
list:
list1: ld     a,(iobyte)
        and   0c0h
        jp   z,dsp80
        cp   40h
        jp   z,dsp40
        cp   0c0h
        jp   z,rs232o
print:
ptrstr: push  bc
        ei
        nop
        nop
        oi
        pop  bc
        ld   a,(bufrdy)
        or   a
        jr   z,ptrstr
;save start pos.
;carry set
;save counter
;save present pos.
;save new pos.
;check iobyte
;list to 80 col
;list to tv
;list to rs232
;save char.
;wait for interrupt
;to print char. in buffer
;if it is full
;get char.
;check buffer ready
;branch for buffer full

```

```

ld      a,0ffh          ;buffer filled sign
ld      (msgrdy),a     ;message is ready for print
ld      hl,(inptr)     ;get spool buffer pointer
ld      a,l
out     (vaddr),a
ld      a,h
add     a,40h          ;set bit 6 for write data
out     (vaddr),a     ;store char. in spool buffer
ld      a,c
out     (vdata),a
inc     hl              ;inc pointer
ld      a,h
cp      rsplen         ;end of spool buffer ?
jr      nz,inrol       ;branch to not end
ld      hl,rsplst      ;reset pointer to start loc.
inrol:  ld      (inptr),hl ;store pointer
        call    chkptr   ;spool buffer full ?
        ld      (bufrdy),a ;store buffer ready flag 1 for not full
        ret

;
; temp
rs232o: ret
;
; check spool pointer routine
;
chkptr: ld      hl,(inptr) ;get push in pointer
        ld      b,h
        ld      c,l
        ld      hl,(outptr) ;get pop out pointer
        or      a           ;reset carry flag
        sbc     hl,bc       ;are they equal ?
        ld      a,h
        or      l           ;if equal a = 0 , buffer not ready
        ret

;*****
; list status
;*****
listst: ld      a,(iobyte)
        and     0c0h
        cp      80h        ;checking printer ?
        jr      z,listst1  ;yes, get buffer ready flag
        xor     a           ;always ready for screen and rs232
        ret
listst1: ld     a,(bufrdy)
        ret

;*****
; punch device
;*****
punch:  ret

;*****
; reader
;*****
reader: batin:
        xor     a
        ret

;*****
; disk routines
;*****
;
; cp/m to host disk constants
;
iobyte equ     0003h      ;i/o byte
cdisk  equ     0004h      ;current disk no.
; 0=a , 1,2,3,4,5,6=b , 7=ram disk

;
; bdos constants on entry to write
; stored in register c
;
wrall  equ     0          ;write to allocated
wrdir  equ     1          ;write to directory
wrnal  equ     2          ;write to unallocated
;
;
; the bdos entry points given below show the
; code which is relevant to deblocking only.
;
;*****
; home the selected disk
;*****

```

```

HOME:
homeCO: ld      a, (hstwr)      ;check for pending write
        or
        jr      nz, homeD
        ld      (hstact), a    ;clear host active flag
homeD:  ret
;*****
;***** select disk
;*****
seldsk: ld      a, c            ;get selected disk
        ld      (savdsk), a    ;save selected disk
        ld      a, (hstdsk)    ;load current selected disk drive
        cp      c              ; = new disk drive ?
        jr      z, seldb      ;branch for new = current
        ld      a, (hstwr)    ;check for pending write
        or
        jr      nz, clrwd     ;clear pending write
seldb:  ld      a, (savdsk)    ;recall selected disk
        ld      c, a
        or
        jr      nz, seld1     ;0 for drive a
seld9:  ld      a, c
        ld      (sekdisk), a
        ret
; clear pending read
clrwd:  ld      a, (sektrk)    ;save selected track
        ld      (savtrk), a
        xor     a
        ld      (sektrk), a    ;set selected track = 0
        call    read          ;read to remove pending write
        ld      a, (savtrk)    ;recall original track
        ld      (sektrk), a
        jr      seldb
seld1:  cp      ramdisk       ;select ram disk ?
        jr      nz, selda
        ld      a, (ramdok)    ;is there any ram disk
        or
        jr      z, nodisk
        ld      hl, ramdph
        jr
selda: ;
        ; notation of drives
        ;
        ; 0 = msx drive a
        ; 1 = b = msx drive b
        ; 2 = c = msx drive b, osborne format
        ; 3 = d = msx drive b, kaypro format
        ; 4 = e = msx drive b, 12/14 single side
        ; 5 = f = msx drive b, 12/14 double side
        ; 6 = g = SVI - 328 single side
        ; 7 = h = ram disk svi - 747
msxa   equ     0
msxb   equ     1
osb    equ     2
kay    equ     3
sl2    equ     4
dl4    equ     5
ssv    equ     6
ramdisk equ    7
;
        ld      a, c
        cp      1
        jr      nz, seld2
        ld      hl, svbdph
        jr      seld9
        ld      (sekdisk), a
        ret
seld2: cp      osb           ;osborne drive
        jr      nz, seld3
        ld      hl, osbdph
        jr      seld9
seld3: cp      kay           ;kaypro drive
        jr      nz, seld4
        ld      hl, kaydph
        jr      seld9
seld4: cp      sl2           ;bondwell 12 drive
        jr      nz, seld5
        ld      hl, sl2dph
        jr      seld9
seld5: cp      dl4           ;bondwell 14 drive

```

```

        jr      nz,seld6
        ld      hl,d14dph
        jr      seld9
seld6:  cp      ssv          ;svi-318 single side
        jr      nz,nodisk
        ld      hl,ssvdph
        jr      seld9
nodisk: ld      hl,0          ;no physical or logical drive
        ret
;*****
;          set track given by registers bc
;*****
settrk: ld      h,b
        ld      l,c
        ld      (sektrk),hl  ;track to seek
        ret
;*****
;          set sector given by register c
;*****
setsec: inc     bc          ;inc for logical sector start at 1
        ld      a,c
        ld      (seksec),a  ;sector to seek
        ret
;*****
;          set dma address given by bc
;*****
setdma: ld      (dmaadr),bc
        ret
;*****
;          translate sector number bc
;*****
sectra: ld      h,b
        ld      l,c
        inc     hl          ;inc for logical sector start at 1
        ret
;*****
;          read the selected cp/m sector ---
;          the read entry point takes the place of
;          the previous bios definition for read
;*****
read:   di
        call   readrt
        ei
        nop
        di
        ret
readrt: call   chgdsk       ;change to suitable disk format
        xor    a
        ld    (unacnt),a
        ld    a,l
        ld    (readop),a  ;read operation
        ld    (rsflag),a  ;must read data
        ld    a,wrua1
        ld    (wrtype),a  ;treat as unalloc
        jp    rwoper      ;to perform the read
;*****
;          write the selected cp/m sector ---
;          the write entry point takes the place of
;          the previous bios definition for write.
;*****
write:  di
        call   writrt
        ei
        nop
        di
        ret
writrt: call   chgdsk       ;change to suitable disk format
        xor    a           ;0 to accumulator
        ld    (readop),a  ;not a read operation
        ld    a,c         ;write type in c
        ld    (wrtype),a
        cp    wrua1       ;write unallocated?
        jr    nz,chkuna   ;check for unalloc
;          write to unallocated, set parameters
;          ld    a,recpbk  ;next unalloc recs
;          db    03eh     ;ld a,

```

```

recpbk: db      0
        ld      (unacnt),a
        ld      a,(sekdisk)      ;disk to seek
        ld      (unadsk),a      ;unadsk = sekdisk
        ld      hl,(sectrk)
        ld      (unatrck),hl    ;unstrk = sectrk
        ld      a,(seksec)
        dec     a                ;sector start from 0
        dec     a                ;for blocking and deblocking
        ld      (unasec),a      ;unasec = seksec
;
shkuna: ;check for write to unallocated sector
        ld      a,(unacnt)      ;any unalloc remain?
        or      a
        jr      z,alloc        ;skip if not
;
; more unallocated records remain
        dec     a                ;unacnt = unacnt - 1
        ld      (unacnt),a
        ld      a,(sekdisk)      ;same disk?
        ld      hl,unadsk
        cp      (hl)            ;sekdisk = unadsk?
        jr      nz,alloc        ;skip if not
;
; disks are the same
        ld      hl,unatrck
        call   sektrckcmp        ;sectrk = unatrck?
        jr      nz,alloc        ;skip if not
;
; tracks are the same
        ld      a,(seksec)      ;same sector?
        dec     a
        dec     a                ;sector start from 0
        ld      hl,unasec
        cp      (hl)            ;seksec = unasec?
        jr      nz,alloc        ;skip if not
;
; match, move to next sector for future ref
        inc    (hl)              ;unasec = unasec + 1
        ld      a,(hl)          ;end of track?
;
; cpmspt
        cp      cpmspt          ;count cp/m sectors
        db      0feh            ;cp in machine code
cpmspt: db      0
        jr      c,noovf         ;skip if no overflow
;
; overflow to next track
        ld      (hl),0          ;unasec = 0
        ld      hl,(unatrck)
        inc    hl
        ld      (unatrck),hl    ;unatrck = unatrck + 1
;
noovf: ;match found, mark as unnecessary read
        xor     a                ;0 to accumulator
        ld      (rsflag),a      ;rsflag = 0
        jp      rwooper         ;to perform the write
;
alloc: ;not an unallocated record, require pre-read
        xor     a                ;0 to accum
        ld      (unacnt),a      ;unacnt = 0
        inc    a                ;1 to accum
        ld      (rsflag),a      ;rsflag = 1
;
;
;
; common code for read and write follows
;
;
rwooper: ;enter here to perform the read/write
        xor     a                ;0 to accum
        ld      (erflag),a      ;no errors (yet)
        ld      a,(seksec)      ;compute host sector
        dec     a                ;start from 0
        dec     a
        push   bc
;
; load no of shift
        ld      b,secshf
        db      06h            ;ld b,
secshf: db      0
getssn: or      a                ;carry = 0
        rra                ;shift right
        djnz   getssh          ;shift until count = 0
        pop    bc

```

```

    ld      (sekfst),a      ;host sector to seek
)
) active host sector?
ld      hl,hstact      ;host active flag
ld      a,(hl)
ld      (hl),1      ;always becomes 1
or      a      ;was it already?
jr      z,filhst      ;fill host if not
)
) host buffer active, same as seek buffer?
ld      a,(sekdst)
ld      hl,hstdsk      ;same disk?
cp      (hl)      ;sekdst = hstdsk?
jr      nz,nomatch
)
) same disk, same track?
ld      hl,hsttrk
call    sektrkcmp      ;sektrk = hsttrk?
jr      nz,nomatch
)
) same disk, same track, same buffer?
ld      a,(sekfst)
ld      hl,hstsec      ;sekfst = hstsec?
cp      (hl)
jr      z,match      ;skip if match
)
) nomatch:
;proper disk, but not correct sector
ld      a,(hstwrtr)      ;host written?
or      a
call    nz,writest      ;clear host buff
)
) filhst:
;may have to fill the host buffer
ld      a,(sekdst)
ld      (hstdsk),a
ld      hl,(sektrk)
ld      (hsttrk),hl
ld      a,(sekfst)
ld      (hstsec),a
ld      a,(rsflag)      ;need to read?
or      a
call    nz,readhst      ;yes, if 1
xor     a      ;0 to accum
ld      (hstwrtr),a      ;no pending write
)
) match:
;copy data to or from buffer
ld      a,(seksec)      ;mask buffer number
dec     a
dec     a
) and     secmsk      ;least signif bits
db      0e6h
) secmsk: db      0
ld      l,a      ;ready to shift
ld      h,0      ;double count
rept    7      ;shift left 7
add     hl,hl
) hl gas relative host buffer address
) memory mapped version
ld      de,(hstbuf)
add     hl,de      ;hl = host address
ld      de,(dmaadr)      ;get/put cp/m data
ld      bc,128      ;length of move
ld      a,(readop)      ;which way?
or      a
jr      nz,rwmove      ;skip if read
)
) write operation, mark and switch direction
ld      a,1
ld      (hstwrtr),a      ;hstwrtr = 1
ex      de,hl      ;source/dest swap
)
) rwmove:
;c initially 128, de is source, hl is dest
ldir
)
) data has been moved to/from host buffer
ld      a,(wrtype)      ;write type
cp      wrdir      ;to directory?
ld      a,(erflag)      ;in case of errors
ret     nz      ;no further processing

```

```

)
)      clear host buffer for directory write
)      or      a      ;errors?
)      ret     nz      ;skip if so
)      xor     a      ;0 to accum
)      ld      (hstwrtr),a      ;buffer written
)      call    writenst
)      ld      a,(erflag)
)      ret
)
)
)      utility subroutine for 16-bit compare
)
)
)
)      sektrkcmp:
)      ;hl = .unatrkr or .hsttrkr, compare with sektrk
)      ex      de,hl
)      ld      hl,sektrk
)      ld      a,(de)      ;low byte compare
)      cp      (hl)      ;same?
)      ret     nz      ;return if not
)      ;low bytes equal, test high 1s
)      inc     de
)      inc     hl
)      ld      a,(de)
)      cp      (hl)      ;sets flags
)      ret
)change disk routine
)chgdsdsk: push  af
)           push  de
)           push  hl
)           ld    a,(sekdsdsk)
)           add   a,a      ;shift to suitable bias
)           add   a,a
)           ld    e,a
)           ld    d,0      ;16 bit add
)           ld    hl,drvtbl ;load drive table pointer
)           add   hl,de
)           ld    a,(hl)   ;get record per block
)           ld    (recpbk),a
)           inc   hl
)           ld    a,(hl)   ;load logical sector per track
)           ld    (cpmspt),a
)           inc   hl
)           ld    a,(hl)   ;load record shift
)           ld    (secmsk),a
)           inc   hl
)           ld    a,(hl)   ;load sector mask
)           ld    (secshf),a
)           pop   hl
)           pop   de
)           pop   af
)           ret
)drive characteristics table
)      recpbk = no of cp/m record per block
)      cpmspt = no of cp/m record per track
)      secmsk = ( physical sector size ) / 128 - 1
)      secshf = log2 [ ( physical sector size ) / 128 ]
)      db      recpbk,cpmspt,secmsk,secshf
)drvtbl: db      16,34,1,1      ;a msx
)         db      16,34,1,1      ;b msx
)         db      8,40,7,3      ;c Osborne
)         db      8,40,3,2      ;d Kaypro
)         db      16,36,1,1      ;e bw12
)         db      16,36,1,1      ;f bw14
)         db      8,34,1,1      ;g svi-328
)         db      8,32,1,1      ;h ram disk
)
)*****
)      hstdsk=host disk #
)      hstsec=host sect # 0x16
)      hsttrk=host track # 0x79
)      erflag non-zero if error
)*****
)writenst:
)         ld    a,(hstdsk)      ;get selected disk
)         cp    ramdisk      ;ram disk ?
)         jp    z,wrtram      ;write ram disk
)         ld    a,l

```

```

        ld      (motron),a      ;set motor on flag
        call   drion           ;turn on drive bank
        call   onmot          ;turn on motor
        ld     a,(motime)      ;motor physically on ?
        or     a
        jr     nz,wsec0       ;branch for physically on
        ld     hl,40000        ;delay 0.5sec
        call   delay0
wsec0:  ld     a,10            ;set error counter
        ld     (errcnt),a
wsec4:  ld     a,(gsttrk)      ;get selected track
        call   seek0         ;seek track
        ld     hl,buffer
        ld     (hstbuf),hl
wsec1:  call   sbias          ;load physical sector transform
        ld     a,(gstsec)
        add    a,b            ;add bias factor start 0 or 1
        ld     (sect),a
        ld     a,wsect       ;get write sector command word
        ld     (compt),a     ;out to command port
        ld     bc,reqpt
        ld     hl,(hstbuf)
        ld     de,0          ;set timer to check no disk
wschk:  ld     a,(bc)
        add    a,a
        jp    p,wsdata
        dec   e              ;decrement timer
        jp    nz,wschk
        dec   d
        jp    nz,wschk
        call  offmot        ;off motor for no disk
        jp    rwrri
wsdata: ld     de,datapt     ;get data from data port
        jp
wsdrq:  ld     a,(bc)        ;read request port
        add    a,a           ;shift 1 bit
        jr    c,wsdone     ;interrupt ?
        jp    m,wsdrq      ;data request ?
wsdat:  ld     a,(hl)        ;get data from buffer
        ld     (de),a       ;write to data port
        inc   hl            ;inc pointer
        jp    wsdrg
wsdone: ld     a,(sttpt)    ;command finished ?
        bit   busy,a
        jr    nz,wsdone    ;wait for finish
        and  wserr         ;any write error
        jp    z,fini       ;branch for ok
;save error flags for display
        ld     b,a
        ld     a,(errcnt)   ;error 10 times ?
        dec   a
        ld     (errcnt),a
        jr    nz,wsec4     ;branch to retry
        jp    rwrri
;
;*****
;
;   hstdsk=host disk #
;   hsttrk=host track # 0479
;   hstsec=host sect # 0416
;   read "hstsiz" bytes
;   erflag non-zero if error
;*****
readhst:
        ld     a,(hstdsk)   ;ram disk ?
        cp    ramdisk
        jp    z,rdram      ;read ram disk
        ld     a,l
        ld     (motron),a  ;set motor on flag
        call   drion
        call   onmot       ;turn on motor
        ld     a,(motime)  ;motor physically on ?
        or     a
        jr     nz,rsec0    ;delay 0.5 sec
        ld     hl,40000
        call   delay0
rsec0:  ld     a,10         ;set error counter
        ld     (errcnt),a
rsec4:  ld     a,(hsttrk)   ;seek required track
        call   seek0
        ld     hl,buffer
        ld     (hstbuf),hl

```



```

rnecl: call sbias ;get physical sector bias
        ld a,(hstsec)
        add a,b ;add physical sector bias
        ld (secpt),a
        ld a,rsect ;load read sector command word
        ld (compt),a
        ld bc,reqpt
        ld hl,(hstbuf)
        ld de,0 ;set timer for no disk check
ruchk: ld a,(bc)
        add a,a
        jp p,rsdata
        dec e
        jp nz,rschk
        dec d ;dec no disk timer
        jp nz,rschk
        call offmot ;off motor for no disk
        jp rwerr1
rndata: ld de,datapt ;read data
        jp rsdat
rsdrq: ld a,(bc) ;check request port
        add a,a ;shift 1 bit
        jr c,rsdone ;branch for interrupt
        jp m,rsdrq ;branch for no data
rmdat: ld a,(de) ;read data
        ld (hl),a ;store in memory
        inc hl ;inc pointer
        jp rsdrq
rsdone: ld a,(sttpt)
        bit busy,a ;finish ?
        jr nz,rsdone
        and rserr ;any read sector error ?
        jp z,fini
;save error flag for display
        ld b,a
        ld a,(errcnt) ;retry 10 times ?
        dec a
        ld (errcnt),a
        jr nz,rsec4.
rwerr1: ; display disk off line error
        ld a,0d0h ;disable intr. code
        out (vaddr),a
        ld a,81h ;point to register 1
        out (vaddr),a
        ld de,msg08 ;disk offline
        ld c,9
        call tobdos ;print buffer
        jp rwerr2
rwerr: ; this part of program display the disk i/o error
; display interrupt first
        .comment $
        ld a,0d0h ;disable intr. code
        out (vaddr),a
        ld a,81h ;point to register 1
        out (vaddr),a
        ld a,b ;recall error flag
        add a,a
        ld (erflag),a
        ld hl,0
        push hl
        ld hl,msg02
        push hl
        ld hl,msg03
        push hl
        ld hl,msg04
        push hl
        ld hl,msg05
        push hl
        ld hl,msg06
        push hl
dsperr: pop de
        ld a,e
        or d
        jr z,dspeak
        ld c,9
        ld a,(erflag)
        add a,a
        ld (erflag),a
        call c,tobdos
        jr dsperr

```

```

dapeok: ld      a, (hstdsk)
        call    hexnum
        ld      a, c
        add    a, 11h          ;bias to display alphabet
        ld      (msg00+8), a
        ld      a, (hsttrk)
        call    hexnum
        ld      a, b
        ld      (msg00+18), a
        ld      a, c
        ld      (msg00+19), a
        ld      a, (hstsec)
        call    hexnum
        ld      a, b
        ld      (msg00+30), a
        ld      a, c
        ld      (msg00+31), a
        ld      de, msg00
        ld      c, 9
        call    tobdos

rwerr2:
;enable interrupt
        ld      a, 0F0h          ;enable intr. code
        out    (vaddr), a
        ld      a, 081h          ;point to register 1
        out    (vaddr), a
        ;
        ld      a, 1            ;set error flag
        ld      (erflag), a
        jr     rwmoff
fini:   ld      a, 0            ;reset error flag
        ld      (erflag), a
rwmoff: ld      a, 120          ;set timer for motor physically off
        ld      (motime), a
        ld      a, 0            ;set motor physically off
        ld      (motron), a
        call    drioff          ;back to cp/m bank
        ld      a, (erflag)
        ret

;
; ram disk read write operation
;
wtram:  ld      a, 1            ;set write operation
        jr     rwraml
rdram:  xor     a
        ;set read operation
rwrmi:  ld      (phywrt), a      ;indication of read/write
        di
        in     a, (porta)
        ld      (pbank4), a     ;save present bank
        and   0c0h
        or    2ah              ;switch ram disk bank
        out    (porta), a
        ld      a, (hsttrk)     ;get track
        add    a, a
        add    a, a
        add    a, a
        add    a, a            ; * 16
        ld      b, a
        ld      a, (hstsec)
        add    a, b            ;add and get mem. page pointing
        cp    rmdbuf          ;on 4th page ?
        jr     nc, page4       ;branch for 4 th page
        call   rwrmi          ;get data in ram disk
rwrmi4: ld      a, (pbank4)     ;restore cp/m bank
        out    (porta), a
        xor    a
        ld      (erflag), a     ;always no error
        ret

; read / write from buffer to page stored in a
rwrmi3: ld      d, a
        ld      e, 0            ;point to ram disk addr
        ld      hl, buffer      ;point to buffer loc
        ld      a, (phywrt)
        or    a
        jr     nz, rwrmi2
        ex    de, hl          ;swap source and dest. for read
rwrmi2: ld      bc, 256
        ldir
        ret

;
; page 4 read / write operation
;

```

```

page4:  add    a,3           ;shift three 256 byte page for workspace
        ld     (locbak),a   ;save the location
        ld     a,(phywrt)   ;identify read / write
        or     a
        jr     nz,wrt4
rdp4:   ld     a,(locbak)    ;get stored location
        ld     d,a         ;save in suitable register
        call  rdp41        ;read data
        ld     a,rndbuf     ;move from ram disk: buffer to buffer
        call  rwr4m3       ;move data
        jr     rwr4m4
wrt4:   ld     a,rndbuf     ;for write operation write to buf. first
        call  rwr4m3       ;move from buffer to ram disk buffer
        ld     a,(locbak)   ;get dest. addr
        ld     d,a
        call  wrtp41       ;write from r. d. buffer to disk
        jr     rwr4m4
rndbuf  equ    0bch        ;location for ram disk buffer
wrtp41  equ    (rndbuf+1)*256 ;entry of page 4 write
rdp41   equ    wrtp41+2    ;entry of page 4 read
;
; this part of program is copied to ram disk bank
; should be stored in large buffer to reduce ram size
;
ovrlay:
wrt42:  jr     wrtp43       ;jump to write
rdp42:  ld     a,0aah        ;64k all ram disk
        out    (porta),a   ;switch page
        ld     e,0         ;source loc. pointer
        ld     l,0
        ld     h,rndbuf    ;read from buffer
        ex     de,hl
        ld     bc,256     ;set counter
        ldir
rdp43:  ld     a,06ah        ;back to bios page
        out    (porta),a
        nop
        nop
        nop
        nop
        ret
wrt43:  ld     a,0aah        ;64k all ram disk
        out    (porta),a
        ld     e,0
        ld     l,0
        ld     h,rndbuf    ;write to buffer
        ld     bc,256     ;set counter
        ldir
        jr     rdp43
endlay:
;
sbias:  ld     hl,sbias     ;get sector bias table
        ld     b,0
        ld     a,(hstdsk)
        ld     c,a
        add   hl,bc        ;point to selected disk bias
        ld     b,(hl)     ;get bias
        ret
sbias:  db     1,1,1,0,0,1  ;1 for start from 1
;
onmot:  ld     a,(sngdri)   ;single drive installed ?
        or     a
        ld     a,(hstdsk)  ;preload selected logical drive
        jr     nz,don4     ;branch for always drive a:
        or     a
        jr     nz,don1     ;branch for drive b
don4:   ld     b,a         ;save hstdsk in b
        ld     a,(dabak)   ;get last logical drive assign to a: physical
        cp     b         ;last logical drive * present logical drive
        ld     a,b
        ld     (dabak),a  ;save present logical drive
;instruction added to disable the
;change disk warning
        jr     don5
        jr     z,don5     ;branch for no change
        ld     de,msg20    ;print 'insert '
        call  prtmsg      ;output to screen
        ld     a,(dabak)  ;get logical drive assigned
        cp     ssv        ;svi single side format ?
        jr     nz,don6
        ld     de,msg26   ;print 'svi-single side'
        jr     don7

```

```

don6:  cp      d14          ;bondwell 14 format ?
      jr      nz,don8
      ld      de,msg25
      jr      don7
don8:  cp      s12         ;bondwell 12 format ?
      jr      nz,don9
      ld      de,msg24
      jr      don7
don9:  cp      kay         ;kaypro format ?
      jr      nz,dona
      ld      de,msg23
      jr      don7
dona:  cp      osb         ;osborne format ?
      jr      nz,donb
      ld      de,msg22
      jr      don7
donb:  ld      de,msg21    ;print 'svi-double side'
don7:  call   prtmsg
      ld      de,msg27    ;print 'format disk and type any key'
      call   prtmsg
      call   conin       ;wait for key
don5:  ld      a,(selreg)
      and    0fch
      or     1           ;drive a on
      jr     don2
don1:  ld      a,(selreg)
      and    0fch
      or     2           ;drive b on
don2:  or     8           ;motor on
      ld      (selpt),a
      ld      (selreg),a
      ld      a,(dskbak)
      ld      b,a
      ld      a,(hstdsk)
      cp     b
      jr     z,don3
      ld      (dskbak),a
      call   dly00
      call   restr
don3:  ret
;
;      print message routine
;
prtmsg: push   de
prtml: pop    de
      ld      a,(de)      ;get a character
      cp     '$'         ;end ?
      ret     z           ;return for end
      inc    de
      push   de
      ld      c,a
      call   conout      ;print character
      jr     prtml       ;loop until end
;
msg20: db     cr,lf,'Insert $'
msg21: db     'SVI-double side$'
msg22: db     'Osborne$'
msg23: db     'Kaypro$'
msg24: db     'Bondwell 12$'
msg25: db     'Bondwell 14$'
msg26: db     'SVI-single side$'
msg27: db     ' format disk and type any key',cr,lf,'$'
;
; off motor
;
offmot: ld     a,0        ;physically turn off motor
      ld     (selpt),a   ;output to i/o port
      ld     (selreg),a
      ld     (motime),a
      ret
;
; restore to track 0
restr: ld     a,restor
      ld     (compt),a
      call   dly00
      call   ready0
      ld     a,(compt)   ;test trk00
      bit   trk00,a
      jr    z,restr
      ret
;
;      delay routine
;

```

```

delay0: dec    hl
        ei
        nop
        di
        ld     a,h
        or     l
        jr     nz,delay0
        ret

; stepper motor stepin one step
;
stepin0: ld     a,stepin
        ld     (compt),a
        call  dly00
        call  ready0
        ret

; seek a required track
; track stored in a
;
seek0:  ld     c,a
        cp     40
        jr     c,seek6
        ld     a,(selreg)
        or     4h                ;select side 1
        ld     (selpt),a
        ld     (selreg),a
        ld     a,c
        sub    40                ;side 1 ?
        jr     seek3
seek6:  ld     a,(selreg)
        and    0fbh              ;select side 0
        ld     (selpt),a
        ld     (selreg),a
        ld     a,c
        jr     seek3
seek00: call    restr              ;if seek error retry
        call    stepin0           ;restore and step in
        jr     seek2
seek3:  ld     b,a
        ld     a,(trkpt)
        cp     b                ;already on required track ?
        ret     z                ;return for equal
seek2:  ld     a,b
        or     a
        jr     z,seek4
        ld     (datapt),a        ;load track to seek
        ld     a,seek
        ld     (compt),a        ;load seek command
        call  dly00             ;delay
        call  ready0            ;
seek1:  ld     a,(compt)         ;ok ?
        bit    busy,a
        jr     nz,seek1
        bit    skerr,a         ;seek error ?
        jr     nz,seek0
seek5:  ld     hl,1200           ;delay 20 msec
        call  delay0
        ret
seek4:  call    restr
        jr     seek5

;
dly00:  nop
        nop
        nop
        nop
        nop
        nop
        ret

;
ready0: call    dly00
ready1: ld     a,(compt)
        bit    busy,a
        jr     nz,ready1
        ret

;
drion:  ld     a,(drive)
        and    0ch
        ld     b,a
        in     a,(porta)

```

```

        ld      (pbank3),a      ;save present bank
        and    0f3h
        or     b                ;get drive bank
        out    (porta),a
        ret
)
drioff: ld      a,(pbank3)      ;off drive bank
        out    (porta),a
        ret
)
;this part of program to display disk r/w fault
)
;convert a one byte data to two byte display
;input from a out put to bc
        .comment %
hexnum: ld      c,a
        srl    a
        srl    a
        srl    a
        srl    a
        ld     e,a
        ld     d,0
        ld     hl,numtbl
        add    hl,de
        ld     a,(hl)
        ld     b,a
        ld     a,c
        and    0fh
        ld     e,a
        ld     d,0
        ld     hl,numtbl
        add    hl,de
        ld     a,(hl)
        ld     c,a
        ret
)
;number table
)
numtbl: db      30h,31h,32h,33h,34h,35h,36h,37h
        db      38h,39h,41h,42h,43h,44h,45h,46h
)
;error message
)
msg00: db      cr,lf,'drive x: track 0xxh sector 0xxh $'
msg04: db      cr,lf,'record not found $'
msg03: db      cr,lf,'crc error $'
msg02: db      cr,lf,'lost data $'
msg06: db      cr,lf,'write protected $'
msg05: db      cr,lf,'write fault $'
msg03: db      cr,lf,'disk offline $'
%
)
;*****
)
***** storages
)
*****
;conin
tmpkey: db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 ;buffer for key debounce
newkey: db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 ;buffer for new key
oldkey: db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 ;buffer for old key
temp0:  db      0 ;temp storage
temp1:  db      0
prsflg: db      0 ;non-zero for key being pressed
ctlkey: db      0 ;control key buffer
ctlky2: db      0 ;control key backup
koycnt: db      0 ;timer for auto repeat key
funflg: db      0 ;non-zero for function key pressed
funadr: dw      0 ;pointer pointing present fkey location
;console out
stkbuf: db      0,0 ;stack buffer
        db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        db      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
biostk: db      0 ;stack for bios
flag40: db      0 ;non zero for no 80 col card.
sndamp: db      0 ;sound amplitude register
escflg: db      0 ;escape flag
esccnt: db      0 ;escape status count
invert: db      0 ;invert display flag

```

```

capflg: db      0      ;capital lock flag
modflg: db      0      ;mode flag
curx:   db     20h     ;screen cursor x pointer
cury:   db     20h     ;screen cursor y pointer
curbuf: db     20h     ;buffer for char. behind cursor
;printer
inptr:  dw     rsplst  ;pointer for spooler input
outptr: dw     rsplst  ;pointer for spooler output
magrdy: db      0      ;spooler ready for output flag
bufrdy: db     0ffh    ;spooler ready for input flag
prtcnt: db      7      ;max no. of char. output per intr.
;drive
dabak:  db      0      ;last logical drive for drive a:
sngdri: db      0      ;single drive system flag
locbak: db      0      ;backup for ram disk sector pointer
phywrt: db      0      ;write operation flag for ram disk
ramdok: db      0      ;ram disk present flag
hstbuf: dw     buffer  ;pointer pointing dma address
errcnt: db      0      ;floppy error counter
savdsk: db      0      ;backup buffer for select disk operation
savtrk: db      0      ;backup buffer for select disk operation
sekdisk: db      0      ;logically selected disk
sektrk: db     0,0     ;logically selected track
seksec:  db      0      ;logically selected sector
hstdsk: db      0      ;physically selected disk
dakbak: db      0      ;backup reg. to show change of logical disk
hstsec:  db      0      ;physically selected sector
hsttrk: db     0,0     ;physically selected track
gstsec  equ     hstsec  ;physically selected sector
gsttrk  equ     hsttrk  ;physically selected track
sekhst: db      0      ;host sector
hstact: db      0      ;host allocation
hntwrt: db      0      ;pending write flag
selreg: db      0      ;backup reg. for disk select output port
unacnt: db      0      ;unallocated counter
unadsk: db      0      ;unallocated drive
unatrck: db     0,0    ;unallocated track
unasec:  db      0      ;unallocated sector
erflag: db      0      ;read write error flag
rnflag: db      0      ;
;
rnadop: db      0      ;read operation flag for floppy
wrtype: db      0      ;write type storage reg.
dmaadr:  db     0,0    ;dma addr storage
scccnt:  db      0      ;sector loading counter
phank1: db      0      ;reg. for saving present bank
phank2: db      0      ;reg. for saving present bank
phank3: db      0      ;reg. for saving present bank
phank4: db      0      ;reg. for saving present bank
;buffers
svcsv0: ds      16     ;check location for a:
svcsv1: ds      16     ;check location for b:
ssvcsv: ds      16     ;check location for g:
osbcsv: ds      16     ;check location for c:
kaycsv: ds      16     ;check location for d:
sl2csv: ds      32     ;check location for f:
ramcsv: ds       8     ;check location for h:
dl4csv: ds      32     ;check location for e:
svalv0: ds      21     ;allo. vector for a:
svalv1: ds      21     ;allo. vector for b:
ssvalv: ds      21     ;allo. vector for g:
oshalv: ds      24     ;allo. vector for c:
kayalv: ds      25     ;allo. vector for d:
sl2alv: ds      11     ;allo. vector for f:
ramalv: ds       8     ;allo. vector for h:
dl4alv: ds      22     ;allo. vector for e:
buffer:  ds     1024   ;read or write buffer
dirbuf:  ds      128   ;directory buffer
prjend:  ds       0      ;program end
;*****
;          end
;*****
end

```

```

TITLE   DSKDRV - MSXDOS disk driver for 2 drive hardware -
;
; Following disk driver is for WD179X series floppy disk
; controllers or equivalent. Smart enough to detect 'drive not
; ready' even if the hardware does not support it. Can transfer
; 128 Kbytes in 6 seconds because the overhead is set to the
; minimum possible time.
;
; For hardware which has images of FDC to help transfer to/from
; RAM at 4000H..7FFFH.
;
; This module is placed somewhere between 6000H and 7FFFH.
;
; By Hitoshi Suzuki (ASCII Microsoft)
;
; As of Mar 22nd, 1984
;
; .Z80
;
; MSX system specific constants
;
CR      EQU    0DH
LF      EQU    0AH
;
; external symbols
;
EXTRN  PROMPT
EXTRN  GETSLOT      ;get slot address (i.e., where I am)
;in [A]
EXTRN  GETWRK      ;get base of work area in [IX] and
;[HL]
EXTRN  DIV16      ;[DC] = [BC]/[DE], remainder in [HL]
EXTRN  XFER       ;block transfer routine
EXTRN  $SECBUF    ;pointer to sector buffer for physical
;driver
EXTRN  RAHAD0,RAHAD1,RAHAD2,RAHAD3
EXTRN  ENASLT
EXTRN  RAWFLG     ;read after write flag
EXTRN  SETINT,PRVINT
EXTRN  DISINT,ENAIINT
;
; definition of my own work area (offset from top)
;
MOTCNT EQU    0      ;motor stop counter
LAST0  EQU    1      ;last access counter for drive 0
LAST1  EQU    2      ;last access counter for drive 1
CURDRV EQU    3      ;currently selected drive (physical)
TRACK0 EQU    4      ;head position for drive 0
TRACK1 EQU    5      ;head position for drive 1
PRVDRV EQU    6      ;previously specified drive
PHYDRV EQU    7      ;number of physical drives
;
WRKLEN EQU    7+1
;
; symbols internally defined
;
PUBLIC MYSIZE
MYSIZE EQU    WRKLEN      ;work area size for this cartridge
;
PUBLIC SECLN
SECLN EQU    512         ;maximum sector size supported by this
;cartridge
;
$SDRIVE:
;
; drive parameters supported by this cartridge
;
DPBTBL:
;
; single side, 9 sectors
;
DPBFCL:  DEFB    0FCH      ;MEDIA
         DEFW    512      ;SECSIZ
         DEFB    00001111B ;DIRMSK
         DEFB    4        ;DIRSHFT
         DEFB    0        ;CLUSMSK
         DEFB    1        ;CLUSSHFT
         DEFW    1        ;FIRFAT
         DEFB    2        ;FATCNT
         DEFB    64       ;MAXENT
         DEFW    9        ;FIRREC
         DEFW    352      ;MAXCLUS
         DEFB    2        ;FATSIZ
         DEFW    5        ;FIRDIR

```



```

;
; double side, 9 sectors
;
DPBFD: DEFB 0FDH ;MEDIA
        DEFW 512 ;SECSIZ
        DEFB 00001111B ;DIRMSK
        DEFB 4 ;DIRSHFT
        DEFB 1 ;CLUSMSK
        DEFB 2 ;CLUSSHFT
        DEFW 1 ;FIRFAT
        DEFB 2 ;FATCNT
        DEFB 112 ;MAXENT
        DEFW 12 ;FIRREC
        DEFW 355 ;MAXCLUS
        DEFB 2 ;FATSIZ
        DEFW 5 ;FIRDIR
;
; single side, 8 sectors
;
DPBFE: DEFB 0FEH ;MEDIA
        DEFW 512 ;SECSIZ
        DEFB 00001111B ;DIRMSK
        DEFB 4 ;DIRSHFT
        DEFB 0 ;CLUSMSK
        DEFB 1 ;CLUSSHFT
        DEFW 1 ;FIRFAT
        DEFB 2 ;FATCNT
        DEFB 64 ;MAXENT
        DEFW 7 ;FIRREC
        DEFW 314 ;MAXCLUS
        DEFB 1 ;FATSIZ
        DEFW 3 ;FIRDIR
;
; double side, 8 sectors
;
DPBFF: DEFB 0FFH ;MEDIA
        DEFW 512 ;SECSIZ
        DEFB 00001111B ;DIRMSK
        DEFB 4 ;DIRSHFT
        DEFB 1 ;CLUSMSK
        DEFB 2 ;CLUSSHFT
        DEFW 1 ;FIRFAT
        DEFB 2 ;FATCNT
        DEFB 112 ;MAXENT
        DEFW 10 ;FIRREC
        DEFW 316 ;MAXCLUS
        DEFB 1 ;FATSIZ
        DEFW 3 ;FIRDIR
;
; default drive parameter for this cartridge
;
PUBLIC DEFDPB
DEFDPB EQU DPBFC
;
; drive capability definition
;
STEP_RATE EQU 00B ;stepping motor rate
MOTOR_TIME EQU 4*60 ;time to stop the motor
USER_TIME EQU 2*60 ;time within which the user may not
;be able to change disks
RETRIES EQU 21 ;number of retries (better be odd)
;
; Command definition
;
FORCE_CMD EQU 11010000B ;force ready command
RESTORE_CMD EQU 00000000B+STEP_RATE ;restore command
STEPIN_CMD EQU 01010000B+STEP_RATE ;step-in command
SEEK_CMD EQU 00010000B+STEP_RATE ;seek command
READ_CMD EQU 10000000B ;sector read command
WRITE_CMD EQU 10100000B ;sector write command
;
; I/O port definition
;
FD_STT EQU 7FB3H ;FDC status
FD_CMD EQU 7FB0H ;FDC command
FD_TRK EQU 7FB9H ;FDC track register
FD_SEC EQU 7FBAH ;FDC sector register
FD_DAT EQU 7FB0H ;FDC data register
FD_SEL EQU 7FBCH ;motor, side select, drive select register
FD_REQ EQU 7FBC2H ;DRQ and IRQ status
;

```

```

; Error condition definition
;
; write sector
;
; NPWRCLxx
; | | | | |
; | | | | |----- lost data
; | | | | |----- CRC error
; | | | | |----- record not found
; | | | | |----- write fault
; | | | | |----- protected
; | | | | |----- not ready
;
; read sector
;
; NxxRCLxx
; | | | | |
; | | | | |----- lost data
; | | | | |----- CRC error
; | | | | |----- record not found
; | | | | |----- not ready
;
; FD_SEL definition
;
; xxxxMS10
; | | | | |
; | | | | |----- enable drive 0
; | | | | |----- enable drive 1
; | | | | |----- select side
; | | | | |----- motor on
;
; FD_REQ definition
;
; IDxxxxxx
; | | | | |
; | | | | |----- 0 when data ready/requested
; | | | | |----- 1 when interrupt requested
;
;
; SUBTTL write
;
; DSKIO entry
;
PUBLIC DSKIO
DSKIO: JP NC,READ ;if no carry, we should read
;
; Write a sector
;
; inputs:
; [A] = drive number ( 0 based )
; [B] = sector count to write
; [C] = media descriptor
;
; OFEH 8 sectors, single side
; OFCH 9 sectors, single side
; OFFH 8 sectors, double side
; OFDH 9 sectors, double side
;
; [DE] = logical sector number
; [HL] = transfer address
;
; outputs:
; if successful, return with carry flag reset
; otherwise, return with carry flag set,
; error code in [A],
; number of sectors remaining in [B]
;
; AF, BC, DE and HL can be modified
;
CALL WRMMAIN ;perform write operation

```

```

RWEND:  PUSH  AF                ;save returned status
        LD    C,USER_TIME      ;assume no errors detected
        JR    NC,NOERR         ;good assumption
        LD    C,0              ;error detected, do not take
                                ;this as 'disk access'
NOERR:  LD    A,FORCE_CMD      ;always force FDC to ready state
        LD    (FD_CMD),A
        EX   (SP),HL
        EX   (SP),HL
        EX   (SP),HL
        EX   (SP),HL
        LD    A,(FD_DAT)       ;clear possible DRQ or INTR
        LD    A,(FD_STT)
        LD    (IX+MOTCNT),MOTOR_TIME ;tell interrupt handler to
                                ;stop the motor after MOTOR_TIME
                                ;which drive has been accessed?
        LD    A,(IX+CURDRV)
        AND  A
        JR    NZ,RWEND1       ;drive 1
        LD    (IX+LAST0),C    ;say disk access just occurred
                                ;on drive 0
        POP  AF                ;restore returned status
        RET

RWEND1: LD    (IX+LAST1),C    ;say disk access just occurred
                                ;on drive 1
        POP  AF                ;restore returned status
        RET

;
WRMAIN: CALL  SELDRV          ;Select drive, track and sector
        RET  C
;
; register usage:
;
; [B] sector counter
; [C] current track
; [D] current status of FD_SEL in lower 4 bits
; bit 7 set if 8 sectors, reset if 9 sectors
; bit 6 set if double side, reset if single side
; [HL] target address
;
        LD    A,H
        AND  A                ;write from RAM above 8000H?
        JP  M,WRITE_DIRECT    ;yes, write directly
        CALL WR_RELOCATE      ;relocate for write operation
        CALL JP_SSECBUF      ;call relocated code
        RET  C                ;write error
        INC  B                ;done all sectors?
        DEC  B
        RET  Z                ;yes
        LD    A,H
        AND  A                ;write from RAM above 8000H?
        JP  M,WRITE_DIRECT    ;yes, write directly
;
; Here when we cannot write directly
;
        PUSH  HL                ;save real target address
        PUSH  DE                ;save FD_SEL content
        PUSH  BC                ;save sector count and media type
        LD    DE,(SSECBUF)     ;move target to SSECBUF
        PUSH  DE
        LD    BC,512
        CALL  XFER
        POP  HL                ;restore address of SSECBUF
        POP  BC                ;restore parameters
        POP  DE
        CALL  WRSECT          ;write a sector from SSECBUF
        POP  HL                ;restore real target address
        JP   MORE_WRITE
;
; Here when we can write directly from the source RAM
;
WRITE_DIRECT:
        CALL  WRSECT
MORE_WRITE:
        RET  C                ;write error
        DEC  B                ;done with all sectors specified?
        RET  Z                ;yes, return with carry flag reset
        CALL  UPDATE          ;prepare for next write
        JP   WRITE_DIRECT     ;go and write next sector

```

```

;
; Routine which actually performs write operation
;
; (E) retry counter
;
WRSECT: LD E,RETRIES ;set retry count
WRRTRY: CALL BSYCHK ;make sure FDC is ready for command
LD A,WRITE_CMD ;sector write command
BIT 6,D ;single sided disk?
JR Z,WRITE2 ;yes
OR 0000010B ;enable side compare flag
BIT 2,D ;which side are we dealing with?
JR Z,WRITE2 ;side 0
OR 00001000B ;compare for side 1
WRITE2: PUSH HL ;save target address
PUSH DE ;save FD_SEL content and retry count
PUSH BC ;save sector count and target track
LD DE,WRDONE ;set where to jump when done
PUSH DE
CALL DISINT
DI
LD (FD_CMD),A ;issue write command
LD BC,FD_REQ
LD DE,FD_DAT
WRLOOP: LD A,(BC) ;read data request port
ADD A,A
RET C ;write confirmed
JP M,WRLOOP ;no data requested
LD A,(HL) ;get from target address
LD (DE),A ;write to FDC's data register
INC HL ;bump address pointer
JP WRLOOP

WRDONE: POP BC ;restore sector count and target track
POP DE ;restore current FD_SEL and retry count
POP HL ;restore target address
EI
CALL ENAINT
LD A,(FD_STT)
AND 1111100B ;not ready or
;write protect or
;write fault or
;record not found or
;CRC error or
;lost data ?
RET Z ;no, return with carry flag reset
JP M,WRITE_NOTRDY ;not ready, do not retry (this is not
;likely to happen)
BIT 6,A ;write protect?
JR NZ,WRRTRY ;yes, do not retry
;
; here when retry might be necessary
;
PUSH AF ;save error code
CALL RESEEK ;restore and re-seek head if necessary
POP AF ;restore error code
DEC E ;should retry?
JR NZ,WRRTRY ;yes
;
; no more retries, report errors to the caller
;
SCF ;set carry to indicate error occurred
LD E,A ;move error status to (E)
BIT 5,E ;write fault?
LD A,10
RET NZ ;yes
BIT 4,E ;record not found?
LD A,0
RET NZ ;yes
BIT 3,E ;CRC error?
LD A,4
RET NZ ;yes
LD A,12 ;treat as 'other error'
RET
;
; here when 'write protected' error occurred. But we are not
; yet sure this is really 'write protected' error because when
; no disk is inserted, the FDC reports the same result.

```

```

WRTprt: LD      A, FORCE_CMD      ; terminate write command
        LD      (FD_CMD), A
        EX     (SP), HL
        EX     (SP), HL
        LD      A, READ_CMD     ; issue read command
        LD      (FD_CMD), A
        EX     (SP), HL        ; kill time
        EX     (SP), HL
        LD      HL, FD_REQ
        LD      DE, 0
WRPRT1: LD      A, (HL)         ; get requests
        ADD     A, A
        JR     C, DSKRDY       ; read confirmed, disk is ready
        JP     P, DSKRDY       ; data ready, disk is ready
        DEC     DE              ; need to wait more?
        LD      A, E
        OR     D
        JP     NZ, WRPRT1      ; yes
;
; here when 'not-ready' error occurred
;
WRITE NOTRDY:
        LD      A, 2           ; ID for 'drive not ready'
        SCF
        RET
;
; FDC responded to read command, disk is surely inserted
;
DSKRDY: XOR     A              ; ID for 'disk write protected'
        SCF
        RET
        SUBTTL read
;
; Read sectors
;
; Inputs:
; {A} = drive number ( 0 based )
; {B} = sector count to read
; {C} = media descriptor
;
;         OFEH  8 sectors, single side
;         OFCH  9 sectors, single side
;         OFFH  8 sectors, double side
;         OFDH  9 sectors, double side
;
; {DE} = logical sector number
; {HL} = transfer address
;
; outputs:
; if successful, return with carry flag reset
; otherwise, return with carry flag set,
; error code in {A},
; number of sectors remaining in {B}
;
; AF, BC, DE and HL can be modified
;
READ:   CALL    RDMAIN         ; perform read operation
        JP     RWEHD         ; force FDC to ready state, etc.
;
RDMAIN: CALL    SELDRV        ; select drive, track and sector
        RET     C
;
; register usage:
;
; {B} sector counter
; {C} current track
; {D} current status of FD_SEL in lower 4 bits
; bit 7 set if 5 sectors, reset if 9 sectors
; bit 6 set if double side, reset if single side
; {HL} target address
;
        LD      A, H
        AND     A              ; read to RAM above 8000H?
        JP     M, READ_DIRECT ; yes, read directly to destination
        CALL    RD_RELOCATE    ; relocate for read operation
        CALL    JP_$SECBUF    ; call relocated code
        RET     C              ; read error
        INC     B              ; done all sectors?
        DEC     B
        RET     Z              ; yes
        LD      A, H
        AND     A              ; read to RAM above 8000H?
        JP     M, READ_DIRECT ; yes, read directly to destination

```

```

;
; Here when we cannot read directly to the destination RAM
;
    PUSH    HL                ;save real target address
    LD      HL,($SECBUF)
    CALL    RDSECT            ;read a sector to $SECBUF
    POP     HL                ;restore real target address
    RET     C                  ;read error
    PUSH    HL
    PUSH    DE
    PUSH    BC
    EX      DE,HL             ;move target address to [DE]
    LD      HL,($SECBUF)
    LD      BC,512
    CALL    XFER
    POP     BC                ;restore parameters
    POP     DE
    POP     HL
    AND     A                  ;make sure carry is cleared
    JP     MORE_READ
;
; Here when we can read directly to the destination RAM
;
READ_DIRECT:
    CALL    RDSECT
    RET     C                  ;read error
MORE_READ:
    DEC     B                  ;done with all sectors specified?
    RET     Z                  ;yes, return with carry flag reset
    CALL    UPDATE
    JP     READ_DIRECT        ;go and read next sector
;
; Routine which actually performs read operation
;
; [E] retry counter
;
RDSECT: LD     A,E,RETRIES      ;set retry count
RDRTRY: CALL   BSYCHK          ;make sure FDC is ready for command
        LD     A,READ_CND      ;sector read command
        BIT   6,D              ;single sided disk?
        JR    Z,READ2         ;yes
        OR    0000010B         ;enable side compare flag
        BIT   2,D              ;which side are we dealing with?
        JR    Z,READ2         ;side 0
        OR    0001100B         ;compare for side 1
READ2:  PUSH   HL              ;save target address
        PUSH   DE              ;save FD_SEL content and retry count
        PUSH   BC              ;save sector count and target track
        LD     BC,FD_REQ       ;load address of IRQ and DRQ
        LD     DE,RDDONE       ;set where to jump when done
        PUSH   DE
        CALL   DISINT
        UI
        LD     (FD_CND),A      ;issue read command
;
; Check first response from FDC. We do this because our hardware
; cannot sense if media is inserted or not
;
WAIT:   LD     DE,0            ;set counter to 65536
        LD     A,(BC)
        ADD   A,A
        JP    P,DISKOK        ;got response, media is inserted
        RET   C                ;read confirmed
        DEC   E
        JP    NZ,WAIT
        DEC   D
        JP    NZ,WAIT
        POP   BC              ;discard jump address
        POP   BC              ;get remaining sector count
        POP   DE              ;discard retry count
        POP   HL              ;discard target address
        JR    READ_NOTRDY     ;drive not ready, no retries
;
DISKOK: LD     DE,FD_DAT       ;load address of FDC data register
        JP    RDLOP1          ;data ready, get it
;
RDLOOP: LD     A,(BC)          ;read data request port
        ADD   A,A
        RET   C                ;read confirmed
        JP    M,RDLOOP        ;no data ready
RDLOP1: LD     A,(DE)         ;read FDC's data register
        LD     (HL),A         ;write to transfer address
        INC   HL              ;bump address pointer
        JP    RDLOOP

```

```

;
RDDONE: POP    BC          ;restore sector count and target track
        POP    DE          ;restore current FD SEL and retry count
        POP    HL          ;restore target address
        EI
        CALL   ENAHIT
        LD     A,(FD_STT)
        AND   1001100B     ;not ready      or
                          ;record not found or
                          ;CRC error      or
                          ;lost data     ?
        RET    Z           ;no, return with carry flag reset
        JP    H,READ_NOTRDY ;not ready, do not retry (this is not
                          ;likely to happen)
;
; here when retry might be necessary
;
        PUSH   AF          ;save error code
        CALL  RESEEK      ;restore and re-seek head if necessary
        POP   AF          ;restore error code
        DEC   E           ;should retry?
        JR    NZ,RDRTRY   ;yes
;
; no more retries, report errors to the caller with carry flag
; set and [B] containing # of sectors remaining
;
        SCF              ;set carry to indicate error occurred
        LD    E,A         ;move error status to [E]
        BIT  4,E         ;record not found?
        LD   A,B
        RET  NZ          ;yes
        BIT  3,E         ;CRC error?
        LD   A,4
        RET  NZ          ;yes
        LD   A,12        ;treat as 'other error'
        RET
;
; here when 'not-ready' error occurred during read
;
READ_NOTRDY:
        LD    A,2         ;ID for 'drive not ready'
        SCF
        RET
;
        SUBTTL  select proper drive
;
; Select drive, track and sector
;
SELDRV: PUSH   AF
        PUSH   BC
        PUSH   HL
        CALL  GETWRK     ;get base of work area in [IX]
        POP   HL
        POP   BC
        POP   AF
        CP    2          ;is drive number in proper range?
        JR    C,SLDRV1   ;yes
BADPRM: LD     A,12      ;maybe internal malfunction
        SCF
        RET
;
SLDRV1: PUSH   AF
        LD    A,C         ;get media descriptor byte
        CP    0FCH       ;within proper range? ( 0FCH .. 0FFH )
        JR    NC,SLDRV2  ;yes
        POP   AF
        JR    BADPRM     ;bad parameter specified
;
SLDRV2: EX    (SP),HL    ;save transfer address
        PUSH  HL         ;save drive number specified
        PUSH  BC         ;save sector count and media type
        CALL BSYCHK     ;make sure FDC is ready
        BIT  1,C         ;8 sectors or 9 sectors per track?
        LD   C,E         ;move logical sector number to [BC]
        LD   B,D         ;for DIV16
        LD   DE,8        ;assume 8 sectors
        JR   NZ,SELAS   ;good assumption
        INC  DE         ;9 sectors per track

```

```

SEL8S: CALL    DIV16      ;[BC] = [BC] / [DE], [HL] = remainder
        LD      A,L      ;now [BC] = track, [HL] = sector - 1
        INC     A
        LD      (FD_SEC),A ;set sector register
        LD      L,C      ;move track to [L]
        POP     BC      ;restore sector count and media type
        POP     AF      ;restore drive number
        LD      H,A      ;remember specified drive
        LD      A,(IX+PHYDRV) ;how many drives do we have?
        DEC     A
        JR      Z,FAKE1  ;only 1 drive, always select drive 0
        LD      A,H      ;restore specified drive

FAKE1: ADD     A,1001B    ;map 0,1 to 1,2 and motor on
        BIT     0,C      ;single side?
        JR      Z,SIDE0  ;yes, always select side 0
        SRL    L        ;divide track by 2 to determine the side
        JR      NC,SIDE0 ;side 0 specified
        OR     0100B    ;side 1 specified

SIDE0: LD      D,A      ;get media descriptor byte
        LD      A,C
        RRCA
        RRCA
        AND    11000000B ;leave only significant bits
        OR     D
        LD      D,A      ;remember what is currently output
        DI
        LD      (FD_SEL),A
        LD      A,(IX+MOTCNT) ;get current motor status
        AND    A        ;see if motor was spinning or not
        LD      (IX+MOTCNT),OFFH ;tell interrupt handler not
                                ;to stop the motor
        EI
        JR      NZ,NOWAIT ;was spinning, no delay required
        PUSH   HL       ;was stopping, wait some time
        LD     HL,0
MOTWAT: DEC    HL
        LD     A,L
        OR    H
        JR    NZ,MOTWAT
        POP   HL
NOWAIT: LD     C,L      ;remember target track
        LD     A,(IX+PHYDRV) ;get number of physical drives
        DEC   A
        JR    Z,FAKE2  ;we have only 1 drive
        LD     A,(IX+CURDRV) ;get currently specified drive
        CP    H        ;same as before?
        JR    Z,SAMDRV  ;yes
        XOR   L        ;flip LSB to switch drives
        LD     (IX+CURDRV),A
        LD     A,(FD_TRK) ;read from FDC's track register
        JR    Z,WAS1   ;from drive 1 to drive 0
;
;   drive is switched from 0 to 1
;
        LD     (IX+TRACK0),A ;remember this
        LD     A,(IX+TRACK1) ;get head position for the new drive
        JR    WAS0
;
;   drive is switched from 1 to 0
;
WAS1: LD     (IX+TRACK1),A ;remember this
        LD     A,(IX+TRACK0) ;get head position for the new drive
WAS0: LD     (FD_TRK),A
        EX    (SP),HL    ;kill time
        EX    (SP),HL
        JR    CHKTRK    ;see if same track as before
;
FAKE2: LD     A,H      ;get specified drive number
        CP    (IX+PRVDRV) ;same as before?
        LD     (IX+PRVDRV),A
        JR    Z,SAMDRV  ;yes
        PUSH  IX       ;save base of work area
        PUSH  DE       ;save FD_SEL status
        PUSH  BC       ;save sector count and current track
        CALL  PROMPT
        POP   BC
        POP   DE
        POP   IX

```



```

SAMDRV: LD      A,(FD_TRK)      ;get track number FDC currently has
CHKTRK: CP      C                ;same with specified track?
        CALL    NZ,SEEKC        ;if not, seek to that track
        POP     HL              ;restore target address
        RET     ;carry flag is always cleared

        SUBTTL  update disk parameters
;
; Update sector, side, and track for multi sector read/write
;
UPDATE: CALL    BSYCHK          ;Make sure FDC is ready
        INC     H                ;advance memory address by 512 bytes
        INC     H
        LD      A,(FD_SEC)      ;load FDC's sector register
        INC     A
        LD      (FD_SEC),A      ;assume within range
        BIT     7,D              ;8 or 9 sectors?
        JR      NZ,UPD8S        ;8 sectors
        CP      9+1             ;need to wrap to next side or track?
        RET     C                ;no
UPD8S:  CP      0+1             ;need to wrap to next side or track?
        RET     C                ;no
        LD      A,1              ;start from sector 1
        LD      (FD_SEC),A
        BIT     6,D              ;single sided drive?
        JR      Z,NXTTRK        ;yes, always go to next track
        BIT     2,D              ;which side are we in now?
        JR      NZ,NXTTRK       ;side 1, have to go to next track
        SET     2,D              ;side 0, have to go to the other side
        LD      A,D
        LD      (FD_SEL),A      ;select side 1
        RET

;
; NXTTRK: RES    2,D              ;go to the other side
        LD      A,D
        LD      (FD_SEL),A      ;select side 0
        INC     C                ;advance target sector
        CALL    BSYCHK          ;make sure FDC is ready
        LD      A,STEPIN_CMD     ;issue step-in command
        LD      (FD_CMD),A
        EX     (SP),HL          ;kill time
        EX     (SP),HL
        JR      BSYCHK          ;wait for termination

        SUBTTL  Miscellaneous procedures
;
; Re-calibrate then re-seek
;
RESEEK: BIT     0,E              ;re-seek only on even retry counts
        RET     NZ
        CALL    RESTOR          ;restore head of current drive
;
; Seek track specified by (C)
;
SEEKC:  LD      A,C              ;get target track
        LD      (FD_DAT),A      ;set destination
        EX     (SP),HL          ;kill time
        EX     (SP),HL
        LD      A,SEEK_CMD      ;issue seek command
CHDWAT: LD      (FD_CMD),A
        EX     (SP),HL          ;kill time
        EX     (SP),HL
;
; Wait until FDC is ready for new command
;
BSYCHK: LD      A,(FD_STT)      ;get FDC status
        RRA                    ;look at busy bit (LSD)
        JR      C,BSYCHK        ;FDC still busy, wait
        RET
;
RESTOR: CALL    BSYCHK
        LD      A,RESTORE_CMD   ;issue restore command
        JR      CHDWAT
;
        SUBTTL  initialization
;
; Initialize hardware only
;
; All registers may be affected
;
PUBLIC  INIHRD

```

```

INIHRD: LD      A,FORCE_CHD      ;force FDC to ready state
        LD      (FD_CMD),A
        EX      (SP),HL         ;kill time
        EX      (SP),HL
        LD      A,0001B         ;restore drive 0
        CALL    INIHRD1
        LD      A,0010B         ;restore drive 1
        CALL    INIHRD1
        XOR     A                ;disable both drives
        LD      (FD_SEL),A
        RET

;
INIHRD1: LD      (FD_SEL),A
        CALL    BSYCHK
        LD      A,RESTORE_CMD   ;issue restore command
        LD      (FD_CMD),A
        EX      (SP),HL         ;kill time
        EX      (SP),HL
        LD      HL,0            ;set loop counter
INIHRD2: LD      A,(FD_STT)      ;get FDC status
        RRA                    ;look at busy bit [LSB]
        RET     NC              ;FDC ready
        DEC     HL
        LD      A,L
        OR      H
        JR      NZ,INIHRD2      ;FDC still busy, wait more
        RET

;
;   get number of drives to [L]
;
;   [HL] can be modified
;
PUBLIC  DRIVES
DRIVES: PUSH    BC
        PUSH    AF
        CALL    GETWRK          ;get work area base to [IX]
        LD      A,0010B         ;enable drive 1
        LD      (FD_SEL),A
        CALL    BSYCHK
        LD      A,RESTORE_CMD   ;issue restore command
        LD      (FD_CMD),A
        EX      (SP),HL         ;kill time
        EX      (SP),HL
        LD      HL,0            ;set loop counter
CHKDR2: LD      A,(FD_STT)      ;get FDC status
        RRA                    ;look at busy bit [LSB]
        JR      NC,HAVTWO       ;FDC ready, we have two drives
        DEC     HL
        LD      A,L
        OR      H
        JR      NZ,CHKDR2      ;FDC still busy, wait more
        INC     L                ;load 1 to [L]
        DEFB    0CAH            ;'JP Z' instruction
HAVTWO: LD      L,2
        LD      (IX+PHYDRV),L   ;remember actual number of drives
        XOR     A                ;disable both drives
        LD      (FD_SEL),A
        POP     AF
        JR      Z,FORCE1        ;ok even if only 1 drive
        LD      L,2             ;pretend we have 2 drives
FORCE1: POP     BC
        RET

;
;   Initialize environments
;
;   All registers may be affected
;
PUBLIC  INIENV
INIENV: CALL    GETWRK          ;get base of work area to [IX] and [HL]
        XOR     A                ;clear out all variables except PHYDRV
        LD      B,WRKLEN-1
INILOP: LD      (HL),A
        INC     HL
        DJNZ   INILOP
        LD      HL,INTRPT
        JP      SETINT          ;save previous hook and set my own

SUBTTL  interrupt handler

```

```

;
;      interrupt handler
;
INTRPT: PUSH    AF                ;[A] must be preserved for 'ON SPRITE
;GOSUB'
        CALL    GETWRK           ;get base of work area in [HL]
        LD      A,(HL)           ;get current motor count
        AND     A                ;stopping?
        JR      Z,INTR1          ;yes, do nothing
        CP      OFFH             ;currently working with disks?
        JR      Z,INTR1          ;yes, do nothing
        DEC     A                ;update it
        LD      (HL),A
        JR      NZ,INTR1         ;motor time not elapsed
        LD      (FD_SEL),A       ;stop the motor
INTR1:  INC     HL
        LD      A,(HL)           ;get 'last disk access' counter for
;drive 0
        AND     A                ;already 0?
        JR      Z,INTR2          ;yes, do nothing
        DEC     (HL)             ;update it
INTR2:  INC     HL
        LD      A,(HL)           ;get 'last disk access' counter for
;drive 1
        AND     A                ;already 0?
        JR      Z,INTR3          ;yes, do nothing
        DEC     (HL)             ;update it
INTR3:  POP     AF
        JP      PRVINT           ;jump to previous interrupt hook

```

SUBTTL disk change handlers

```

;
;      See if disk has been possibly changed
;
; inputs:
; [A] = drive number ( 0 based )
; [B] = 0
; [C] = media descriptor
; [HL] = base address of DPB
;
PUBLIC DSKCHG
DSKCHG: PUSH    HL                ;save address of DPB
        PUSH    BC                ;save media ID
        PUSH    AF                ;save drive number
        CALL    GETWRK           ;get base of work area in [IX]
        POP     AF                ;restore drive number
        POP     BC                ;restore media ID
        POP     HL                ;restore address of DPB
        LD      B,(IX+PHYDRV)     ;do we have two drives?
        DEC     B
        JR      Z,DSKCHG_IDRIVE ;no
        AND     A                ;status for which drive?
        AND     A                ;status for which drive?
        LD      D,(IX+LAST1)     ;assume drive 1
        JR      NZ,DSKCH1        ;good assumption
DSKCHG_IDRIVE:
        LD      B,(IX+LAST0)     ;drive 0
DSKCH1: AND     A                ;clear carry
        INC     B                ;has there been a disk access within
;USR TIME?
        LD      B,1              ;assume so
        RET     NZ               ;good assumption, disk must not have
;been changed, or a SUPERMAN may be
;using the system
        PUSH    BC                ;save passed media ID
        PUSH    HL                ;save base of DPB
        LD      DE,1             ;read first sector of FAT to SSECBUF
        LD      HL,($SSECBUF)
        CALL    READ
        JR      C,DSKCHG_ERROR ;read error
        LD      HL,($SSECBUF)
        LD      B,(HL)           ;get first byte of FAT
        POP     HL                ;restore base of DPB
        PUSH    BC                ;save first byte of FAT
        CALL    GETDPB           ;build a new DPB for the current media
        LD      A,12             ;assume 'other error'
        JR      C,DSKCHG_ERROR ;good assumption
        POP     AF                ;restore first byte of FAT to [A]
        POP     BC                ;restore passed media ID in [C]
        CP      C                ;same with passed ID?
        SCF                    ;clear carry flag only
        CCF
        LD      B,-1             ;assume not
        RET     NZ               ;disk has been definitely changed
        INC     B                ;have no idea whether the disk has

```

```

                                ;been changed or not
                                RET
;
DSKCHG_ERROR:
    POP     DE
    POP     DE
    RET
;
; Build a new DPB for the specified drive. Since our hardware
; is assumed to be 5inch IBM disk compatible, all we have to
; do is to check the first byte of FAT passed from the caller
; and return one of the possible 4 DPBs.
;
    PUBLIC  GETDPB
GETDPB:  EX    DE,HL                ;move DPB address to [DE]
        INC    DE                    ;skip DRVNUM entry
        LD     A,B
        SUB    OFCH                ;get rid of offset
        RET    C                    ;something is wrong
        LD     L,A                  ;[HL] = [HL] * 18
        LD     H,0
        ADD    HL,HL
        LD     C,L
        LD     B,H
        ADD    HL,HL
        ADD    HL,HL
        ADD    HL,HL
        ADD    HL,BC
        LD     BC,DPBTBL
        ADD    HL,BC
        LD     BC,18
        LDIR
        RET
;
        SUBTTL  disk, formatter
    PUBLIC  CHOICE
;CHOICE: LD     HL,CHOMSG
CHOICE: LD     HL,0
        RET
;
;CHOMSG: DEFB    CR,LF
;        DEFB    '1 - 8 sectors',CR,LF
;        DEFB    '2 - 9 sectors',CR,LF
;        DEFB    CR,LF
;        DEFB    0
;
    PUBLIC  DSKFMT
;DSKFMT: CP     2+1                ;good choice?
;        JR     NC,FMTERR          ;no
;
;
; Your format routine comes here
;
;
; AND     A
; RET
;
;FMTERR:
; LD     A,12                    ;bad parameter
; SCF
; RET
;
        SUBTTL  expanded statements for OEM
    PUBLIC  OEMSTATEMENT
OEMSTATEMENT:
    SCF                            ;no expanded statements
    RET
;
; .280

WRITRK_CMD EQU    11110000B
;
INDEX EQU    00000010B
;
HEAD EQU    0
TRACK EQU    1
SECTOR EQU    2
RETRY EQU    3

```

```

;
DSKFHT: CALL  FORMAT
        PUSH  AF
        LD    A,0000B
        LD    (FD_SEL),A
        POP  AF
        EI
        RET
;
FORMAT: DI
        LD    A,D
        CP    1
        JR    NC,EROR12
        PUSH HL
        LD    HL,6500h
        SBC  HL,BC
        POP  HL
        JR    NC,EROR14
        PUSH HL
INITFD: LD    A,FORCE_CHD
        LD    (FD_CHD),A
        EX   (SP),HL
        EX   (SP),HL
        LD    A,1001B
        LD    (FD_SEL),A
        CALL BSCHK
        LD    A,RESTORE_CHD
        LD    (FD_CHD),A
        EX   (SP),HL
        EX   (SP),HL
        LD    HL,0
INIFD1: LD    A,(FD_STT)
        RRA
        JR    NC,WINDEX
        DEC  HL
        LD    A,H
        OR   L
        JR    NZ,INIFD1
        JR    EROR02
WINDEX: LD    HL,0
WINHIL: LD    A,(FD_STT)
        AND  INDEX
        JR    Z,WINLOW
        DEC  HL
        LD    A,H
        OR   L
        JR    NZ,WINHIL
        JR    WINERR
WINLOW: LD    HL,0
WINLOL: LD    A,(FD_STT)
        AND  INDEX
        JR    NZ,FRMAT
        DEC  HL
        LD    A,H
        OR   L
        JR    NZ,WINLOL
EROR02: SCF
WINERR: POP  HL
        LD    A,2
        RET
;
EROR12: SCF
        LD    A,12
        RET
;
EROR14: SCF
        LD    A,14
        RET
;
FRMAT:  POP  HL
        PUSH IX
        PUSH HL
        EX   (SP),HL
        EX   (SP),IX
        EX   (SP),HL
        POP  HL
        INC  HL
        INC  HL
        INC  HL
        INC  HL
;
        LD    A,1001B
        LD    (FD_SEL),A
        CALL  DLAFL
        XOR  A
        LD    (IX+HEAD),A
        DEC  A
        LD    (IX+RETRY),A
        LD    (IX+TRACK),A
TRKLOP: INC  (IX+TRACK)
TRKLP1: INC  (IX+RETRY)
        CALL SPRDBL
        CALL TRKWRT
        CALL VERIFY
        JR    NC,CONT1
        LD    A,9
        CP   (IX+RETRY)
        JR    NZ,TRKLP1
        LD    A,16
        SCF
        POP  IX
CONT1:  LD    A,0FFh
        LD    (IX+RETRY),A
        LD    A,1101B
        LD    (FD_SEL),A
        INC  (IX+HEAD)
TRKLP2: INC  (IX+RETRY)
        CALL SPRDBL
        CALL TRKWRT
        CALL VERIFY
        JR    NC,CONT2
        LD    A,9
        CP   (IX+RETRY)
        JR    NZ,TRKLP2
        LD    A,16
        SCF
        POP  IX
CONT2:  DEC  (IX+HEAD)
        LD    A,39
        CP   (IX+TRACK)
        JR    Z,FHTDUN
        CALL STEPIN
        LD    A,1001B
        LD    (FD_SEL),A
        JR    TRKLOP
;
FHTDUN: CALL  RESTOR
        PUSH HL
        EX   DE,HL
        POP  HL
        PUSH HL
        INC  DE
        XOR  A
        LD    (HL),A
        LD    BC,512*12-1
        LDIR
        POP  HL
        PUSH HL
        LD    BC,512
        ADD  HL,BC
        LD    A,0FDh
        LD    (HL),A
        INC  HL
        LD    A,0FFh
        LD    (HL),A
        INC  HL
        LD    (HL),A
;
        LD    BC,512*2-2
        ADD  HL,BC
        LD    A,0FDh
        LD    (HL),A
        INC  HL
        LD    A,0FFh
        LD    (HL),A
        INC  HL
        LD    (HL),A

```

```

;
; LD      A,0           ;0-BASED DRIVE #
; LD      B,12          ;# SECTOR
; LD      C,0FDH        ;MEDIA
; LD      DE,0          ;START OF LOGICAL SECTOR #
; POP     HL            ;X'FER ADDR
; SCF                     ;WRITE OPERATION
; CALL   WRBOOT
; POP     IX
; RET

;
;DIAYL: PUSH   HL
;        PUSH   AF
;        LD     HL,0
;DIAYL1: LD     DEC     HL
;        LD     A,H
;        OR     L
;        JR     NZ,DIAYL1
;        POP   AF
;        POP   HL
;        RET
;

STEPIN: LD     A,STEPIN_CND
        PUSH  AF
        CALL  BSYCHK
        POP   AF
        LD   (FD_CND),A
STEPIN1: LD   DLAYS
        LD   A,(FD_REQ)
        ADD  A,A
        JR   NC,STEPIN1
        CALL BSYCHK
;

DLAYS:  EX   (SP),HL
        EX   (SP),HL
        EX   (SP),HL
        EX   (SP),HL
        RET

;
SECNUM EQU 9
SECLN  EQU 512
SECLID EQU 2
;
GAP1LN EQU 50
GAP2LN EQU 22
GAP3LN EQU 32
GAP5LN EQU 80
SYNCLN EQU 12
;
GAPCOD EQU 04EH
IM      EQU 0F6H
IAM     EQU 0FCH
IDAH    EQU 0F5H
IDH     EQU 0FEH
DAH     EQU 0F5H
DH      EQU 0FBH
CRCGEN EQU 0F7H
;
SKWTBL: DB 1,6,2,7,3,8,4,9,5
;
SPRDBL: LD   C,1
        LD   D,H
        LD   E,L
        LD   A,GAPCOD
        LD   B,GAP1LN
        CALL FLDPAD
SPR1PD: XOR  A
        LD   B,SYNCLN
        CALL FLDPAD
        LD   A,IDAH
        LD   B,3
        CALL FLDPAD
        LD   A,IDH
        LD   (DE),A
        INC  DE
        LD   A,(IX+TRACK)
        LD   (DE),A
        INC  DE
        LD   A,(IX+HEAD)
        LD   (DE),A
        INC  DE
        PUSH HL
        LD   HL,SKWTBL-1
        LD   B,C
        ADD  HL,BC
        LD   A,(HL)
        POP  HL
        LD   (DE),A

```

```

INC      DE
LD       A,SECLID
LD       (DE),A
INC      DE
LD       A,CRCGEN
LD       (DE),A
INC      DE
LD       A,GAPCOD
LD       B,GAP2LN
CALL     FLOPAD
XOR      A
LD       B,SYNCLN
CALL     FLOPAD
LD       A,DAH
LD       B,3
CALL     FLOPAD
LD       A,DM
LD       (DE),A
INC      DE
LD       A,OE5H
LD       B,0
CALL     FLOPAD
CALL     FLOPAD
LD       A,CRCGEN
LD       (DE),A
INC      DE
LD       A,GAPCOD
LD       B,GAP3LN
CALL     FLOPAD
INC      C
LD       A,C
CP       SECNUM+1
JR       NZ,SPRLPD
LD       A,GAPCOD
LD       B,0
CALL     FLOPAD
CALL     FLOPAD
RET

;
FLOPAD: LD       (DE),A
INC      DE
DEC      B
JR       NZ,FLOPAD
RET

;
TRKWRT: PUSH     HL
LD       DE,TWRD0H
PUSH     DE
LD       BC,FD_REQ
LD       DE,FD_DAT
CALL     BSYCHK
LD       A,(IX+TRACK)
LD       (FD_DAT),A
CALL     DLAYS
CALL     BSYCHK
LD       A,SEEK_CMD
LD       (FD_CMD),A
;TR1: LD       A,(FD_STT)
AND     00010010B
BIT     4,A
JR       NZ,SEEKER
BIT     1,A
JR       NZ,TR1
;TR2: LD       A,(FD_STT)
AND     INDEX
JR       Z,TR2
;TR3: LD       A,(FD_STT)
AND     INDEX
JR       NZ,TR3
LD       A,WRITRK_CMD
LD       (FD_CMD),A
TWRLOP: LD       A,(BC)
ADD     A,A
RET     C
JP      M,TWRLOP
LD       A,(HL)
LD       (DE),A
INC      HL
JR       TWRLOP

;SEEKER: POP     DE
;        POP     HL
;        POP     HL
;        POP     IX
;        LD      A,6
;        SCF
;        RET
;
TWRD0H: POP     HL
LD       A,(FD_STT)
LD       E,A
AND     11100100B
RET     Z
POP     BC
TWRD10: SCF
POP     IX
JP      P,TWRD11
LD       A,2
RET
TWRD11: BIT     6,A
JR      Z,TWRD12
LD       A,0
RET
TWRD12: BIT     5,E
LD       A,10
RET     NZ
BIT     4,E
LD       A,3
RET     NZ
BIT     3,E
LD       A,4
RET     NZ
LD       A,16
RET

;
VERIFY: PUSH     HL
XOR     A
NXTSEC: PUSH     AF
CALL    BSYCHK
POP     AF
INC     A
LD      (FD_SEC),A
PUSH    AF
CALL    DLAYS
CALL    BSYCHK
LD      DE,SRDONE
PUSH    DE
LD      A,READ_CMD
LD      (FD_CMD),A
LD      BC,FD_REQ
LD      DE,FD_DAT
SRDLOP: LD      A,(BC)
ADD     A,A
RET     C
JP      M,SRDLOP
LD      A,(DE)
LD      (HL),A
INC     HL
JR      SRDLOP
;
SRDONE: LD      A,(FD_STT)
LD      E,A
AND     11111100B
JR      Z,GNXSEC
POP     AF
POP     HL
POP     HL
POP     IX
SCF
JP      P,SRDN1
LD      A,2
RET
SRDN1: BIT     5,E
LD      A,10
RET     NZ
BIT     4,E
LD      A,3
RET     NZ
BIT     3,E
LD      A,4
RET     NZ
LD      A,16
RET

;
GNXSEC: POP     AF

```

```

        CP      9
        JR      C,HXTSEC
        POP     HL
        PUSH   HL
        LD      BC,1200H
GNXSC1: LD      A,0E5H
        CP      (HL)
        JR      NZ,ERROR
        INC     DE
        DEC     BC
        LD      A,B
        OR      C
        JR      NZ,GNXSC1
        POP     HL
        RET

;
ERROR:  SCF
        POP     HL
        RET

;
WRBOOT: AND     A                ;CLEAR CARRY
        CALL   BTWRH
        PUSH   AF
        LD     A,FORCE_CMD
        LD     (FD_CMD),A
        CALL   DLAYS
        LD     A,(FD_DAT)
        LD     A,(FD_STT)
        POP   AF
        RET

;
BTWRH:  LD     A,1001B
        LD     (FD_SEL),A
;
        CALL   DLAYL
        XOR    A
        LD     (IX+SECTOR),A
        LD     B,9
BTWRLP: PUSH   BC
        INC    (IX+SECTOR)
        CALL   BSYCHK
        LD     A,(IX+SECTOR)
        LD     (FD_SEC),A
        CALL   BSYCHK
        LD     A,WRITE_CMD
        LD     DE,WRDUNE
        PUSH   DE
        LD     (FD_CMD),A
        LD     BC,FD_REQ
        LD     DE,FD_DAT
WRLOP:  LD     A,(BC)
        ADD    A,A
        RET    C
        JP     M,WRLOP
        LD     A,(HL)
        LD     (DE),A
        INC    HL
        JP     WRLOP

;
WRDUNE: LD     A,(FD_STT)
        AND    1111100B
        POP    BC
        JP     NZ,TWRDNO
        LD     A,(IX+SECTOR)
        CP     9
        JR     Z,CHG_HEAD
        DEC    B
        JR     NZ,BTWRLP
        SCF
        CCF
        RET

;
CHG_HEAD:
        LD     A,1101B
        LD     (FD_SEL),A
        LD     B,3
        XOR    A
        LD     (IX+SECTOR),A
        JR     BTWRLP

BOOT_SECTOR:
BASENT EQU 04020H
FUNC    EQU 0F370H
KHUF    EQU 0F41FH

        .PHASE 0C000H

        DB     0EBH,0FEH        ;JUMP SHORT S
        DB     090H             ;NOP

```



```

; DISK PARAMETER BLOCK

DB      'ASC 2.2'
DW      0          ;BYTES PER SECTOR
DB      0          ;SECTORS PER CLUSTER
DW      0          ;NO. OF RESERVED SECTORS
DB      0          ;NO. OF FATS
DW      0          ;NO. OF DIRECTORY ENTRIES
DW      0          ;TOTAL NO. OF SECTORS IN MEDIA
DB      0          ;MEDIA DESCRIPTOR
DW      0          ;NO. OF SECTORS PER FAT
DW      0          ;SECTORS PER TRACK
DW      0          ;NO. OF HEADS
DW      0          ;NO. OF HIDDEN SECTORS

```

```

BOOT:
RET     HC
LD      ($DOSON*1),DE
LD      (NOTFIRST),A
LD      (HL),LOW BOOTERRVECT
INC     HL
LD      (HL),HIGH BOOTERRVECT

```

```

TRYAGAIN:
LD      SP,KBUF+100H
LD      DE,COMFCB
LD      C,00FH
CALL    FUNC
INC     A
JP      Z,NOFILE
LD      DE,00100H
LD      C,01AH
CALL    FUNC
LD      HL,00001H
LD      (COMFCB+14),HL
LD      HL,03F00H
LD      DE,COMFCB
LD      C,027H
CALL    FUNC
JP      00100H

```

```

; ** NO EXECUTION PATH TO HERE **

```

```

BOOTERRVECT:

```

```

DW      BOOTERR
BOOTERR:
$DOSON:
CALL    00000H
LD      A,C
AND     0FEH
CP      002H
JP      NZ,BOOTERR1

```

```

NOFILE:
LD      A,(NOTFIRST)
AND     A
JP      Z,BASENT

```

```

BOOTERR1:
LD      DE,BOOTERR_HSG
LD      C,009H
CALL    FUNC
LD      C,007H
CALL    FUNC
JR      TRYAGAIN

```

```

BOOTERR_HSG:
DB      'Boot error',00DH,00AH
DB      'Press any key for retry',00DH,00AH,'S'

```

```

COMFCB:
DB      0,'MSXDOS SYS'
DB      0,0,0,0,0,0,0,0,0,0
DB      0,0,0,0,0,0,0,0,0,0
DB      0,0,0,0,0

```

```

NOTFIRST:
DS      1

```

```

.DEPHASE

```

```

RD_RELOCATE:
PUSH    HL
PUSH    DE
PUSH    BC
LD      HL,READDATA      ;transfer raw data for read
LD      DE,($SECDUF)
LD      BC,PR_LEN
LDIR
LD      HL,RD_RELOC_DATA
JR      REL1

```

```

;
WR_RELOCATE:
    PUSH    HL
    PUSH    DE
    PUSH    BC
    LD      HL,WRITEDATA    ;transfer raw data for write
    LD      DE,(SSECBUF)
    LD      BC,PW_LEN
    LDIR
RELI:
    LD      HL,WR_RELOC_DATA
    LD      E,(HL)          ;get from table
    INC    HL
    LD      D,(HL)
    INC    HL
    LD      A,E              ;end of table?
    OR     D
    JR     Z,RELEND        ;yes, all done
    PUSH   HL                ;save table pointer
    LD     HL,(SSECBUF)
    ADD    HL,DE
    INC    HL                ;point to operand
    LD     C,(HL)            ;get operand in [BC]
    INC    HL
    LD     D,(HL)
    EX    DE,HL              ;save operand address in [DE]
    LD     HL,(SSECBUF)
    ADD    HL,BC              ;calculate real operand
    EX    DE,HL
    LD     (HL),D            ;modify operand
    DEC    HL
    LD     (HL),E
    POP    HL
    JR     RELI
;
RELEND: POP    BC
        POP    DE
        POP    HL
        RET
;
WR_RELOC_DATA:
    DEFW   WR01
    DEFW   WR02
    DEFW   WR03
    DEFW   WR04
    DEFW   WR05
    DEFW   WR06
    DEFW   WR07
    DEFW   WR08
    DEFW   WR09
    DEFW   WR10
    DEFW   WR11
    DEFW   WR12
    DEFW   WR13
    DEFW   WR14
    DEFW   WR15
    DEFW   0
;
RD_RELOC_DATA:
    DEFW   RR01
    DEFW   RR02
    DEFW   RR03
    DEFW   RR04
    DEFW   RR05
    DEFW   RR06
    DEFW   RR07
    DEFW   RR08
    DEFW   RR09
    DEFW   RR10
    DEFW   RR11
    DEFW   RR12
    DEFW   RR13
    DEFW   RR14
    DEFW   RR15
    DEFW   RR16
    DEFW   RR17
    DEFW   0
;
JP_SSECBUF:
    PUSH   HL
    LD     HL,(SSECBUF)
    EX    (SP),HL
    RET

```

```

/
PRINTV MACRO VALUE
IF1
.PRINTX * Length = VALUE *
ENDIF
ENDM

/
; Read routine which is transferred to SSECBUF
/
READDATA:
.PHASE 0
PUSH HL
PUSH DE
PUSH BC
CALL GETSLOT ;get where I am
RR01: LD (PR_DUMMY+1),A ;to return to original slot
LD H,80H ;set page 2 to FDC
CALL ENASLT
LD A,(RAMAD1) ;set page 1 to RAM
LD H,40H
CALL ENASLT
POP BC
POP DE
POP HL
PR_RDSECT:
DEC HL
LD A,H
ADD A,2
INC HL
RR02: JP M,PR_END ;transfer is above 8000H
LD E,RETRIES ;set retry count
PR_RDTRY:
RR03: CALL PR_BSYCHK ;make sure FDC is ready for command
LD A,READ_CMD ;sector read command
BIT 6,D ;single sided disk?
JR Z,PR_READ2 ;yes
OR 00000010B ;enable side compare flag
BIT 2,D ;which side are we dealing with?
JR Z,PR_READ2 ;side 0
OR 00001000B ;compare for side 1
PR_READ2:
PUSH HL ;save target address
PUSH DE ;save FD_SEL content and retry count
PUSH BC ;save sector count and target track
LD BC,FD_REQ+4000H ;load address of IRQ and DRQ
RR04: LD DE,PR_RDDONE ;set where to jump when done
PUSH DE
CALL DISINT
DI
LD A,(FD_CMD+4000H),A ;issue read command
;
; Check first response from FDC. We do this because our hardware
; cannot sense if media is inserted or not
;
LD DE,0 ;set counter to 65536
PR_WAIT:
LD A,(BC)
ADD A,A
RR05: JP P,PR_DISKOK ;got response, media is inserted
RET C ;read confirmed
DEC E
RR06: JP NZ,PR_WAIT
DEC D
RR07: JP NZ,PR_WAIT
POP BC ;discard jump address
POP BC ;get remaining sector count
POP DE ;discard retry count
POP HL ;discard target address
JR PR_READ_NOTRDY ;drive not ready, no retries
;
PR_DISKOK:
LD DE,FD_DAT+4000H ;load address of FDC data register
RR08: JP PR_RDLOP1 ;data ready, get it
;
PR_RDLOOP:
LD A,(BC) ;read data request port
ADD A,A
RET C ;read confirmed
RR09: JP M,PR_RDLOOP ;no data ready
PR_RDLOP1:
LD A,(DE) ;read FDC's data register
LD (HL),A ;write to transfer address
INC HL ;bump address pointer
RR10: JP PR_RDLOOP

```

```

;
PR_RDDONE:
    POP    DC           ;restore sector count and target track
    POP    DE           ;restore current FD_SEL and retry count
    POP    HL           ;restore target address
    EI
    CALL   ENAINT
    LD     A,(FD_STT+4000H)
    AND   10011100B    ;not ready          or
                                ;record not found or
                                ;CRC error        or
                                ;lost data       ?
    JR     NZ,PR_RDERR ;yes
    DEC   -D            ;done all sectors?
    JR     Z,PR_END    ;yes
RR11:    CALL  PR_UPDATE ;update parameters
    JR     PR_RDSECT   ;continue
;
PR_RDERR:
RR12:    JP     H,PR_READ_NOTRDY ;not ready, do not retry (this
                                ;is not likely to happen)
;
;       here when retry might be necessary
;
RR13:    PUSH  AF           ;save error code
    CALL  PR_RESEEK       ;restore and re-seek head if necessary
    POP   AF             ;restore error code
    DEC   E              ;should retry?
    JR     NZ,PR_RDRTRY  ;yes
;
;       no more retries, report errors to the caller with carry flag
;       set and (B) containing # of sectors remaining
;
    LD     E,A           ;move error status to (E)
    BIT   4,E           ;record not found?
    LD     A,B
    JR     NZ,PR_ERR    ;yes
    BIT   3,E           ;CRC error?
    LD     A,1
    JR     NZ,PR_ERR    ;yes
    LD     A,12         ;treat as 'other error'
    JR     PR_ERR
;
;       here when 'not-ready' error occurred during read
;
PR_READ_NOTRDY:
    LD     A,2           ;ID for 'drive not ready'
PR_ERR:  SCF
PR_END:  PUSH  HL
    PUSH  DE
    PUSH  BC
    PUSH  AF
    LD     A,(RAMAD2)   ;change page 2 to RAM
    LD     H,80H
    CALL  ENASLT
PR_DUMMY:
    LD     A,0           ;where I was
    LD     H,40H
    CALL  ENASLT
    POP   AF
    POP   BC
    POP   DE
    POP   HL
    RET
;
PR_UPDATE:
RR14:    CALL  PR_DSYCHK  ;make sure FDC is ready
    INC   H              ;advance memory address by 512 bytes
    INC   H
    LD     A,(FD_SEC+4000H) ;load FDC's sector register
    INC   A
    LD     (FD_SEC+4000H),A ;assume within range
    BIT   7,D           ;8 or 9 sectors?
    JR     NZ,PR_UPDAS  ;8 sectors
    CP   9+1           ;need to wrap to next side or track?
    RET   -C            ;no

```

```

PR_UPDRS:
    CP      0+1          ;need to wrap to next side or track?
    RET     C            ;no
    LD      A,1          ;start from sector 1
    LD      (FD_SEC+4000H),A
    BIT     6,D          ;single sided drive?
    JR      Z,PR_NXTTRK ;yes, always go to next track
    BIT     2,D          ;which side are we in now?
    JR      NZ,PR_NXTTRK ;side 1, have to go to next track
    SET     2,D          ;side 0, have to go to the other side
    LD      A,D
    LD      (FD_SEL+4000H),A ;select side 1
    RET

;
PR_NXTTRK:
    RES     2,D          ;go to the other side
    LD      A,D
    LD      (FD_SEL+4000H),A ;select side 0
    INC     C            ;advance target sector
RR15:    CALL PR_BSYCHK  ;make sure FDC is ready
    LD      A,STEPIN_CMD ;issue step-in command
    LD      (FD_CMD+4000H),A
    EX     (SP),HL      ;kill time
    EX     (SP),HL
    JR     PR_BSYCHK    ;wait for termination

;
;      Re-calibrate then re-seek
;
PR_RESEEK:
    BIT     0,E          ;re-seek only on even retry counts
    RET     NZ
RR16:    CALL PR_RESTORE ;restore head of current drive

;
;      Seek track specified by [C]
;
    LD      A,C          ;get target track
    LD      (FD_DAT+4000H),A ;set destination
    EX     (SP),HL      ;kill time
    EX     (SP),HL
    LD      A,SEEK_CMD  ;issue seek command

PR_CMDWAT:
    LD      (FD_CMD+4000H),A
    EX     (SP),HL      ;kill time
    EX     (SP),HL

;
;      Wait until FDC is ready for new command
;
PR_BSYCHK:
    LD      A,(FD_STT+4000H) ;get FDC status
    RRA
    JR      C,PR_BSYCHK  ;FDC still busy, wait
    RET

;
PR_RESTORE:
RR17:    CALL PR_BSYCHK
    LD      A,RESTORE_CMD ;issue restore command
    JR     PR_CMDWAT

PR_LEN EQU $
        .RADIX 16
        PRINTV 'PR_LEN
        .RADIX 10
        .DEPHASE

;
;      Write routine which is transferred to SSECBUF
;
WRITEDATA:
        .PHASE 0
        PUSH HL
        PUSH DE
        PUSH BC
WR01:    CALL GETSLOT    ;get where I am
    LD      (PW_DUMMY+1),A ;to return to original slot
    LD      H,30H        ;set page 2 to FDC
    CALL ENASLT
    LD      A,(RAMAD1)  ;set page 1 to RAM

```

```

        LD      H,40H
        CALL   ENASLT
        POP    BC
        POP    DE
        POP    HL
PW_WRSCT:
        DEC    HL
        LD     A,H
        ADD   A,2
        INC   HL
WR02:   JP     M,PW_END      ;transfer is above 8000H
        LD     E,RETRIES    ;set retry count
PW_WRRTRY:
WR03:   CALL   PW_BSYCHK     ;make sure FDC is ready for command
        LD     A,WRITE_CMD  ;sector write command
        BIT   6,D           ;single sided disk?
        JR    Z,PW_WRITE2   ;yes
        OR    0000010B      ;enable side compare flag
        BIT   2,D           ;which side are we dealing with?
        JR    Z,PW_WRITE2   ;side 0
        OR    0000100B      ;compare for side 1
PW_WRITE2:
        PUSH  HL            ;save target address
        PUSH  DE            ;save FD_SEL content and retry count
        PUSH  BC            ;save sector count and target track
WR04:   LD     DE,PW_WRDONE  ;set where to jump when done
        PUSH  DE
        CALL  DISINT
        DI
        LD    (FD_CMD+4000H),A ;issue write command
        LD    BC,FD_REQ+4000H
        LD    DE,FD_DAT+4000H
PW_WRLOOP:
        LD    A,(BC)        ;read data request port
        ADD  A,A
        RET  C              ;write confirmed
WR05:   JP    M,PW_WRLOOP   ;no data requested
        LD    A,(HL)        ;get from target address
        LD    (DE),A        ;write to FDC's data register
        INC  HL             ;bump address pointer
WR06:   JP    PW_WRLOOP
;
PW_WRDONE:
        POP   BC            ;restore sector count and target track
        POP   DE            ;restore current FD_SEL and retry count
        POP   HL            ;restore target address
        EI
        CALL  ENAINT
        LD    A,(FD_STT+4000H)
        AND  1111100B      ;not ready          or
                          ;write protect         or
                          ;write fault          or
                          ;record not found     or
                          ;CRC error           or
                          ;lost data           ?
        JR    NZ,PW_WRERR   ;yes
        DEC  B              ;done all sectors?
        JR    Z,PW_END      ;yes
WR07:   CALL  PW_UPDATE     ;update parameters
        JR    PW_WRSCT     ;continue
;
PW_WRERR:
WR08:   JP    M,PW_WRITE_NOTRDY ;not ready, do not retry (this
                          ;is not likely to happen)
        BIT  6,A            ;write protect?
        JR    NZ,PW_WRTPRT  ;yes, do not retry
;
;       here when retry might be necessary
;
WR09:   PUSH  AF            ;save error code
        CALL  PW_RESEK     ;restore and re-seek head if necessary
        POP  AF            ;restore error code
        DEC  E              ;should retry?
        JR    NZ,PW_WRRTRY  ;yes
;
;       no more retries, report errors to the caller

```

```

;
LD      E,A          ;move error status to (E)
BIT     5,E          ;write fault?
LD      A,10
JR      NZ,PW_ERR    ;yes
BIT     4,E          ;record not found?
LD      A,3
JR      NZ,PW_ERR    ;yes
BIT     3,E          ;CRC error?
LD      A,4
JR      NZ,PW_ERR    ;yes
LD      A,12        ;treat as 'other error'
JR      PW_ERR
;
; here when 'write protected' error occurred. But we are not
; yet sure this is really 'write protected' error because when
; no disk is inserted, the FDC reports the same result.
;
PW_WRTPRT:
LD      A,FORCE_CMD ;terminate write command
LD      (FD_CMD+4000H),A
EX      (SP),HL
EX      (SP),HL
LD      A,READ_CMD  ;issue read command
LD      (FD_CMD+4000H),A
EX      (SP),HL    ;kill time
EX      (SP),HL
LD      HL,FD_REQ+4000H
LD      DE,0
PW_WRPRT1:
LD      A,(HL)      ;get requests
ADD     A,A
JR      C,PW_DSKRDY ;read confirmed, disk is ready
WR10:   JP      P,PW_DSKRDY ;data ready, disk is ready
DEC     DE          ;need to wait more?
LD      A,E
OR      D
WR11:   JP      NZ,PW_WRPRT1 ;yes
;
; here when 'not-ready' error occurred
;
PW_WRITE_NOTRDY:
LD      A,2          ;ID for 'drive not ready'
JR      PW_ERR
;
; FDC responded to read command, disk is surely inserted
;
PW_DSKRDY:
XOR     A           ;ID for 'disk write protected'
PW_ERR: SCF
PW_END: PUSH HL
        PUSH DE
        PUSH BC
        PUSH AF
LD      A,(RAMAD2) ;change page 2 to RAM
LD      H,00H
CALL   ENASLT
PW_DUMMY:
LD      A,0         ;where I was
LD      H,40H
CALL   ENASLT
POP     AF
POP     BC
POP     DE
POP     HL
RET
;
PW_UPDATE:
WR12:   CALL    PW_BSYCHK ;make sure FDC is ready
        INC     H          ;advance memory address by 512 bytes
        INC     H
LD      A,(FD_SEC+4000H) ;load FDC's sector register
INC     A
LD      (FD_SEC+4000H),A ;assume within range
BIT     7,D         ;3 or 9 sectors?
JR      NZ,PW_UPDSS ;3 sectors
CP      9+1        ;need to wrap to next side or track?
RET     C          ;no

```

```

PW_UPD8S:
    CP      3+1          ;need to wrap to next side or track?
    RET     C            ;no
    LD      A,1         ;start from sector 1
    LD      (FD_SEC+4000H),A
    BIT     5,D         ;single sided drive?
    JR      Z,PW_NXTTRK ;yes, always go to next track
    BIT     2,D         ;which side are we in now?
    JR      NZ,PW_NXTTRK ;side 1, have to go to next track
    SET     2,D         ;side 0, have to go to the other side
    LD      A,D
    LD      (FD_SEL+4000H),A ;select side 1
    RET

;
PW_NXTTRK:
    RES     2,D         ;go to the other side
    LD      A,D
    LD      (FD_SEL+4000H),A ;select side 0
    INC     C           ;advance target sector
WR13:    CALL  PW_BSYCHK  ;make sure FDC is ready
    LD      A,STEPIN_CMD ;issue step-in command
    LD      (FD_CMD+4000H),A
    EX     (SP),HL     ;kill time
    EX     (SP),HL
    JR      PW_BSYCHK  ;wait for termination

;
;       Re-calibrate then re-seek
;
PW_RESEEK:
    BIT     0,E         ;re-seek only on even retry counts
    RET     NZ
WR14:    CALL  PW_RESTOR ;restore head of current drive
;
;       Seek track specified by [C]
;
    LD      A,C         ;get target track
    LD      (FD_DAT+4000H),A ;set destination
    EX     (SP),HL     ;kill time
    EX     (SP),HL
    LD      A,SEEK_CMD  ;issue seek command
PW_CMDWAT:
    LD      (FD_CMD+4000H),A
    EX     (SP),HL     ;kill time
    EX     (SP),HL
;
;       Wait until FDC is ready for new command
;
PW_BSYCHK:
    LD      A,(FD_STT+4000H) ;get FDC status
    RRA
    JR      C,PW_BSYCHK  ;FDC still busy, wait
    RET

;
PW_RESTOR:
WR15:    CALL  PW_BSYCHK
    LD      A,RESTORE_CMD ;issue restore command
    JR      PW_CMDWAT
PW_LEN   EQU   $
        .RADIX 16
        PRINTV APW_LEN
        .RADIX 10
        .DEPHASE

END

```



APPENDIX D

COMPONENT SPECIFICATION

TMS9128/TMS9129 VDP TERMINAL ASSIGNMENTS

SIGNATURE	TERMINAL	I/O	DESCRIPTION
XTAL1	40	I	10.7 MHz crystal connections
XTAL2	39	O	
R-Y	38	O	R-Y color difference output.
CPUCLK	37	O	CPUCLK equals XTAL/3 (Color Burst Frequency)
Y	36	O	Y (Black/White luminance and composite sync) output.
B-Y	35	O	B-Y color difference output.
RESET/ SYNC	34	I	RESET --This pin is a trilevel input pin. When it is below 0.8 volts, RESET initializes the VDP. When it is above 9 volts, RESET is the synchronizing input for external VDP video.
VCC	33	I	+5 volt supply

RAS	1	40	XTAL1
CAS	2	39	XTAL2
AD7	3	38	R-Y
AD6	4	37	CPUCLK
AD5	5	36	Y
AD4	6	35	B-Y
AD3	7	34	RESET/SYNC
AD2	8	33	VCC
AD1	9	32	RD0
AD0	10	31	RD1
R/W	11	30	RD2
VSS	12	29	RD3
MODE	13	28	RD4
CSW	14	27	RD5
CSR	15	26	RD6
INT	16	25	RD7
CD7	17	24	CD0
CD6	18	23	CD1
CD5	19	22	CD2
CD4	20	21	CD3

TMS9118 VDP TERMINAL ASSIGNMENTS (Concluded)

SIGNATURE	TERMINAL	I/O	DESCRIPTION
RD0 MSB	32	I/O	VRAM read/write data bus. RD0 is the most significant bit.
RD1	31	I/O	
RD2	30	I/O	
RD3	29	I/O	
RD4	28	I/O	
RD5	27	I/O	
RD6	26	I/O	
RD7 LSB	25	I/O	CPU data bus. CD0 is the most significant bit.
CD0 MSB	24	I/O	
CD1	23	I/O	
CD2	22	I/O	
CD3	21	I/O	
CD4	20	I/O	
CD5	19	I/O	
CD6	18	I/O	
CD7 LSB	17	I/O	
INT	16	O	CPU interrupt output.
CSR	15	I	CPU-VDP read strobe
CSW	14	I	CPU-VDP write strobe
MODE	13	I	CPU interface mode select; usually a processor address line
VSS	12	I	Ground Reference
R/W	11	O	VRAM write strobe
AD0 MSB	10	O	VRAM address bus. AD0 is the most significant bit.
AD1	9	O	
AD2	8	O	
AD3	7	O	
AD4	6	O	
AD5	5	O	
AD6	4	O	
AD7 LSB	3	O	
CAS	2	O	VRAM column address strobe
RAS	1	O	VRAM row address strobe

\* The least-significant address bit, AD7, is wired to A0 of the dynamic RAMs. Likewise, AD6 is wired to A1 of the RAMs. Care must be exercised in assuring proper orientation of the TMS9118 address outputs to the dynamic RAM address inputs (See Figures 4 and 5).

TMS9118 VDP TERMINAL ASSIGNMENTS

SIGNATURE	TERMINAL	I/O	DESCRIPTION																																																												
XTAL1	40	I	10.7 MHz crystal connections																																																												
XTAL2	39	O																																																													
CPUCLK	38	O	CPUCLK equals XTAL-3 (color burst frequency) <table border="1" style="float: right; margin-left: 20px;"> <tr><td>RAS</td><td>1</td><td>40</td></tr> <tr><td>CAS</td><td>2</td><td>39</td></tr> <tr><td>AD7</td><td>3</td><td>38</td></tr> <tr><td>AD6</td><td>4</td><td>37</td></tr> <tr><td>AD5</td><td>5</td><td>36</td></tr> <tr><td>AD4</td><td>6</td><td>35</td></tr> <tr><td>AD3</td><td>7</td><td>34</td></tr> <tr><td>AD2</td><td>8</td><td>33</td></tr> <tr><td>AD1</td><td>9</td><td>32</td></tr> <tr><td>AD0</td><td>10</td><td>31</td></tr> <tr><td>R/W</td><td>11</td><td>30</td></tr> <tr><td>VSS</td><td>12</td><td>29</td></tr> <tr><td>MODE</td><td>13</td><td>28</td></tr> <tr><td>CSW</td><td>14</td><td>27</td></tr> <tr><td>CSR</td><td>15</td><td>26</td></tr> <tr><td>INT</td><td>16</td><td>25</td></tr> <tr><td>CD7</td><td>17</td><td>24</td></tr> <tr><td>CD6</td><td>18</td><td>23</td></tr> <tr><td>CD5</td><td>19</td><td>22</td></tr> <tr><td>CD4</td><td>20</td><td>21</td></tr> </table>	RAS	1	40	CAS	2	39	AD7	3	38	AD6	4	37	AD5	5	36	AD4	6	35	AD3	7	34	AD2	8	33	AD1	9	32	AD0	10	31	R/W	11	30	VSS	12	29	MODE	13	28	CSW	14	27	CSR	15	26	INT	16	25	CD7	17	24	CD6	18	23	CD5	19	22	CD4	20	21
RAS	1	40																																																													
CAS	2	39																																																													
AD7	3	38																																																													
AD6	4	37																																																													
AD5	5	36																																																													
AD4	6	35																																																													
AD3	7	34																																																													
AD2	8	33																																																													
AD1	9	32																																																													
AD0	10	31																																																													
R/W	11	30																																																													
VSS	12	29																																																													
MODE	13	28																																																													
CSW	14	27																																																													
CSR	15	26																																																													
INT	16	25																																																													
CD7	17	24																																																													
CD6	18	23																																																													
CD5	19	22																																																													
CD4	20	21																																																													
N.C.	37		Reserved. Do not use.																																																												
COMVID	36	O	Composite video output <table border="1" style="float: right; margin-left: 20px;"> <tr><td>XTAL1</td><td>40</td></tr> <tr><td>XTAL2</td><td>39</td></tr> <tr><td>CPUCLK</td><td>38</td></tr> <tr><td>N.C.</td><td>37</td></tr> <tr><td>COMVID</td><td>36</td></tr> <tr><td>EXT VDP</td><td>35</td></tr> <tr><td>RESET/SYNC</td><td>34</td></tr> <tr><td>VCC</td><td>33</td></tr> <tr><td>RD0</td><td>32</td></tr> <tr><td>RD1</td><td>31</td></tr> <tr><td>RD2</td><td>30</td></tr> <tr><td>RD3</td><td>29</td></tr> <tr><td>RD4</td><td>28</td></tr> <tr><td>RD5</td><td>27</td></tr> <tr><td>RD6</td><td>26</td></tr> <tr><td>RD7</td><td>25</td></tr> <tr><td>CD0</td><td>24</td></tr> <tr><td>CD1</td><td>23</td></tr> <tr><td>CD2</td><td>22</td></tr> <tr><td>CD3</td><td>21</td></tr> </table>	XTAL1	40	XTAL2	39	CPUCLK	38	N.C.	37	COMVID	36	EXT VDP	35	RESET/SYNC	34	VCC	33	RD0	32	RD1	31	RD2	30	RD3	29	RD4	28	RD5	27	RD6	26	RD7	25	CD0	24	CD1	23	CD2	22	CD3	21																				
XTAL1	40																																																														
XTAL2	39																																																														
CPUCLK	38																																																														
N.C.	37																																																														
COMVID	36																																																														
EXT VDP	35																																																														
RESET/SYNC	34																																																														
VCC	33																																																														
RD0	32																																																														
RD1	31																																																														
RD2	30																																																														
RD3	29																																																														
RD4	28																																																														
RD5	27																																																														
RD6	26																																																														
RD7	25																																																														
CD0	24																																																														
CD1	23																																																														
CD2	22																																																														
CD3	21																																																														
EXT VDP	35	I	Parallel operation (more than 4 sprites per line)																																																												
RESET/SYNC	34	I	RESET --This pin is a trilevel input pin. When it is below 0.8 volts, RESET initializes the VDP. When it is above 9 volts, RESET is the synchronizing input for external VDP video.																																																												
VCC	33	I	+5 volt supply																																																												

TMS9128/TMS9129 VDP TERMINAL ASSIGNMENTS (Concluded)

SIGNATURE	TERMINAL	I/O	DESCRIPTION
RD0 MSB	32	I/O	VRAM read/write data bus. RD0 is the most significant bit.
RD1	31	I/O	
RD2	30	I/O	
RD3	29	I/O	
RD4	28	I/O	
RD5	27	I/O	
RD6	26	I/O	
RD7	25	I/O	CPU data bus. CD0 is the most significant bit.
CD0 MSB	24	I/O	
CD1	23	I/O	
CD2	22	I/O	
CD3	21	I/O	
CD4	20	I/O	
CD5	19	I/O	
CD6	18	I/O	
CD7 LSB	17	I/O	
INT	16	O	CPU interrupt output.
CSR	15	I	CPU-VDP read strobe
CSW	14	I	CPU-VDP write strobe
MODE	13	I	CPU interface mode select; usually a processor address line
VSS	12	I	Ground Reference
R/W	11	O	VRAM write strobe
AD0 MSB	10	O	VRAM address (multiplexed high and low order VRAM address). AD0 is the most significant bit.*
AD1	9	O	
AD2	8	O	
AD3	7	O	
AD4	6	O	
AD5	5	O	
AD6	4	O	
AD7 LSB	3	O	
CAS	2	O	VRAM column address strobe
RAS	1	O	VRAM row address strobe

\* The least-significant address bit, AD7, is wired to A0 of the dynamic RAMs. Likewise, AD6 is wired to A1 of the RAMs. Care must be exercised in assuring proper orientation of the TMS9118 address outputs to the dynamic RAM address inputs (See Figures 4 and 5).

TMS9118/9128/9129 PRELIMINARY ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS OVER OPERATING FREE-AIR TEMPERATURE RANGE  
(unless otherwise noted)\*

Supply voltage, VCC	-0.3 to 20 V
All input voltages	-0.3 to 20 V
Output voltage	-2 to 7 V
Continuous power dissipation	1.3 W
Operating free-air temperature range	0°C to 70°C
Storage temperature range	-55°C to +150°C

\*Stresses beyond 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 beyond those indicated in the Recommended Operating Conditions section of this specification is not implied. Exposure to absolute maximum rated conditions for extended periods may affect device reliability.

RECOMMENDED OPERATING CONDITIONS\*

PARAMETER		MIN	NOM	MAX	UNIT
Supply voltage, VCC		4.75		5.25	V
Supply voltage, VSS			0		V
Input voltage, VI, RESET/SYNC pin	SYNC active	10		12	V
	RESET active			0.6	V
	SYNC and RESET inactive	3		6	V
High-level input, VIH	XTAL1, XTAL2	2.75			V
	All other inputs	2.2			V
Low-level input voltage, VIL				0.8	V
Operating free-air temperature, TA		0		70	°C

\*All voltage values are with respect to VSS.

ELECTRICAL CHARACTERISTICS OVER FULL RANGES OF RECOMMENDED OPERATING CONDITIONS (Unless Otherwise Noted)

TMS9118/9128/9129

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
VOH	High-level output voltage	RAS, CAS, R/W All other outputs IOH = 400 uA	2.7	3.4		V
	Low-level output voltage	CPU data DRAM interface IOL = 1.2 mA		0.3	0.6	V
VOL	Off-state output current high-level voltage applied, CD0-CD7 outputs	VO = 5.25 V		1	100	uA
	Off-state output current low-level voltage applied, CD0-CD7 outputs	VO = 0.4 V		1	-100	uA
IIH	High-level input current	VI = 5.25 V, all other pins at 0 V			10	uA
IIl	Low-level input current	VI = 0 V, All other pins at 0 V			-10	uA

TMS9118 Only (Figure-12)

Vwhite	Video voltage level of white, Y, R-Y, B-Y outputs		2.5	3	3.6	V
Vblack	Video voltage level of black (blank), Y/R, Y, B-Y outputs	R = 470 Ohms L	1.6	2.3	2.5	V
Vsync	Video voltage level of sync, Y output	C = 50 pF L	1.2	1.8	2	V
R L	Video output load resistor		470			ohms

ELECTRICAL CHARACTERISTICS OVER FULL RANGES OF  
RECOMMENDED OPERATING CONDITIONS (unless otherwise noted)  
(Continued)

TMS9118 Only (Figure-12)

V	Video voltage difference, Vwhite, Vblack, Y, R-Y, B-Y outputs	RL = 470 Ohms CL = 50 pF	.8	1	1.5	V
V	black sync	RL = 470 Ohms CL = 50 pF	.300	.400	.700	V
Vpos	Color burst video voltage level with respect to V no color	R-Y output (TMS9129 only)		.25		V
Vneg	Color burst video voltage level with respect to V no color	B-Y output (TMS9128/9129)		-.25		V

TMS9118/9128/9129 (Figure 13)

ICC	Average supply current from VCC	TA = 25°C	180	250	mA
Ci	Input capacitance	CD0-CD7 unmeasured f = 11 MHz, pins at 0 V		20	pF
		All other inputs		10	
Co	Output capacitance	unmeasured f = 11 MHz, pins at 0 V		20	pF

All typical values are at VCC = 5.25 V, TA = 25°C.

TIMING REQUIREMENTS OVER FULL RANGES OF RECOMMENDED OPERATING  
CONDITIONS FOR TMS9118/9128/9129

CPU-VDP Interface (Figure 14 and 15)

PARAMETER	MIN	NOM	MAX	UNIT
tsu(ARL) Address setup time before CSR low		0		ns
tsu(AWL) Address setup time before CSW low		30		ns
th(WLA) Address hold time after CSW low		30		ns
tsu(DWH) Data setup time before CSW high		100		ns
th(WHD) Data hold time after CSW high		30		ns
tw(WL) Pulse width, CSW low		200		ns

VDP-VRAM Interface

PARAMETER	MIN	NOM	MAX	UNIT
tsu(DCH) Input data setup time before CAS high	60			ns
th(CBD) Input data hold time after CAS high	0			ns

Register And Memory Access Cycle Times

PARAMETER	MIN	NOM	MAX	UNIT
tcy (REG) Cycle time between registers read or write	2		2	us
tcy (MEM) Cycle time between memory read or write				
MODES				
BLANKED SCREEN	2		2	us
RETRACE (4.3 msec after Interrupt)	2		2	us
TEXT				
MULTICOLOR \ ACTIVE	2		3.1	us
GRAPHICS I / AREA	2		3.5	us
GRAPHICS II /	2		8.0	us
NOTE: Maximum cycle times must be respected at all times for proper operation				

EXTERNAL CLOCK SOURCE (TMS9118/9128/9129) (Figure 16)

PARAMETER		MIN	TYP	MAX	UNIT
f <sub>ext</sub>	External source frequency	10.738098	10.738635	10.739172	MHz
t <sub>r</sub> /t <sub>f</sub>	External source rise/fall time		10	15	ns
t <sub>WH</sub>	External source high level pulse width	42	47	52	ns
t <sub>WL</sub>	External source low level pulse width	42	47	52	ns
t <sub>PD</sub>	External source phase delay from XTAL1 falling edge to XTAL2 falling edge	42	47	52	ns

SWITCHING CHARACTERISTICS OVER FULL RANGE OF RECOMMENDED OPERATING CONDITIONS (TMS9118/9128/9129)

CPU-VDP Interface

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
t <sub>A</sub> (CSR)	Data access time from CSR low			100	150	ns
t <sub>PVX</sub>	Data disable time after CSR high			65	100	ns
t <sub>PVX,A</sub>	Data invalid time from address changes	C = 300 pF L		0		ns
f <sub>CPUCLK</sub>	CPU clock output clock frequency (XTAL:3)		3.4	3.58	3.76	MHz

SWITCHING CHARACTERISTICS OVER FULL RANGE OF RECOMMENDED OPERATING CONDITIONS (Continued)

TMS9118 COMPOSITE VIDEO output (Figures 17 and 18)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
t <sub>f1</sub>	Fall time, V <sub>black</sub> to V <sub>sync</sub>			10		ns
t <sub>w</sub> (HS)	Pulse width, horizontal sync			4.84		µs
t <sub>r1</sub>	Rise time, V <sub>sync</sub> to V <sub>black</sub>			20		ns
t <sub>HS-CD</sub>	Delay time, Sync to Color Burst			372		ns
t <sub>w</sub> (CB)	Width, color burst			2.61		µs
t <sub>CB-LB</sub>	Delay time, Color Burst to left border			1.49		µs
t <sub>r2</sub>	Rise time, V <sub>black</sub> to V <sub>white</sub>			60		ns
t <sub>w</sub> (LB)	Left border video width	R = 470 Ω L		2.42		µs
t <sub>f2</sub>	Fall time, V <sub>white</sub> to V <sub>black</sub>	C = 50 pF L		110		ns
t <sub>w</sub> (AD)	Width of Active Display Area			47.68		µs
t <sub>w</sub> (RB)	Right border video width			2.79		µs
t <sub>RB-HS</sub>	Delay time, right border to horizontal sync			1.49		µs
t <sub>VFB</sub>	Vertical front blanking			191.1		µs
t <sub>VS</sub>	Vertical sync			191.1		µs
t <sub>VBB</sub>	Vertical back blanking			828		µs
t <sub>ABA</sub>	Active plus border area time			18.8		ms

SWITCHING CHARACTERISTICS OVER FULL RANGE OF  
RECOMMENDED OPERATING CONDITIONS (Continued)

TMS9128/9129 Y, R-Y, B-Y outputs (Figures 19 through 22)

PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
tf3 Fall time, Vblack to Vsync			100		n
tw(HS1) Pulse width, Horizontal SYNC			4.84		u
tr3 Rise time, Vsync to Vblack			150		n
tw(BP) Width, back porch	R = 470 $\Omega$ L		4.47		u
tw(LB1) Width, left border			2.8		u
tw(P) Pulse width, pixel	C = 50pF L		186.24		n
tw(HORZ) Width, horizontal line			63.695		u
tw(AD1) Width, active display area			47.67		u
tr4 Rise time, Vblack to Vwhite			75		n
tr4 Fall time, Vwhite to Vblack			50		n
tw(RB1) Width, right border			2.42		u
tw(FP) Width, front porch			1.49		u
tr5 Rise time, V no color to V pos color burst	R = 470 $\Omega$ L C = 150 pF L		150		n
tw(CB1) Pulse width, pos color burst			2.6		u
tf5 Fall time, V pos color burst to V no color			100		n

SWITCHING CHARACTERISTICS OVER FULL RANGE OF  
RECOMMENDED OPERATING CONDITIONS (Concluded)

TMS9128/9129 Y, R-Y, B-Y outputs (Figures 19 through 22)

PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
tw(CB-LB1) Delay time, pos color burst to left border			1.49		u
tf6 Fall time, V no color to V neg color burst			100		n
tr6 Rise time, V neg color burst to V no color	R = 470 $\Omega$ L		150		n
tw(VS1) Pulse width, vertical sync	C = 50pF L		465		n
tVFB1 Vertical front blanking			191.09		u
tVS1 Vertical sync			191.09		u
tVBB1 Vertical back blanking			828.04		u
tABA1 Active area plus border area total			18.70		m
VERTICAL TIME			19.91		m

Note: Fall times depend on external pull-down resistor.

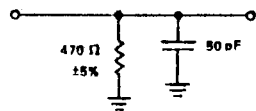
TIMING REQUIREMENTS OVER RECOMMENDED SUPPLY VOLTAGE RANGE AND  
OPERATING FREE-AIR TEMPERATURE RANGE FOR TMS9118/9128/9129

(Figures 14 and 15)

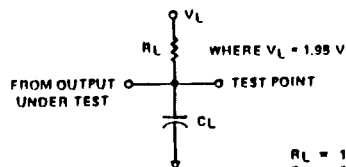
PARAMETER	ALT. SYMBOL	MIN	TYP	MAX	UNIT
t <sub>c</sub> (rd) Read cycle time*	t <sub>RC</sub>	370	372		ns
t <sub>c</sub> (W) Write cycle time	t <sub>WC</sub>	370	372		ns
t <sub>w</sub> (CH) Pulse width, $\overline{\text{CAS}}$ high (precharged time)**	t <sub>CP</sub>	85	125		ns
t <sub>w</sub> (CL) Pulse width, $\overline{\text{CAS}}$ low T	t <sub>CAS</sub>	195	235	270	ns
t <sub>w</sub> (RH) Pulse width $\overline{\text{RAS}}$ high (precharged time)	t <sub>RP</sub>	120	135		ns
t <sub>w</sub> (RL) Pulse width, $\overline{\text{RAS}}$ low*	t <sub>RS</sub>	200	230	270	ns
t <sub>w</sub> (W) Write pulse width *	t <sub>WP</sub>	330	370		ns
t <sub>t</sub> Transition times (rise and fall) for $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$	t <sub>T</sub>	3	15	50	ns
t <sub>su</sub> (CA) Column address setup time	t <sub>ASC</sub>	0	25		ns
t <sub>su</sub> (RA) Row address setup time	t <sub>ASR</sub>	10	40		ns
t <sub>su</sub> (D) Data setup time with respect to $\overline{\text{CAS}}$ high	t <sub>DS</sub>	0	25		ns
t <sub>su</sub> (rd) Read command setup time	t <sub>RCS</sub>	0	35		ns
t <sub>su</sub> (WCH) Write command setup time before $\overline{\text{CAS}}$ high	t <sub>CWL</sub>	240	280		ns
t <sub>su</sub> (WRH) Write command setup time before $\overline{\text{RAS}}$ high	t <sub>RWL</sub>	145	185		ns
t <sub>h</sub> (CLCA) Column address hold time after $\overline{\text{CAS}}$ low	t <sub>CAH</sub>	190	230		ns
t <sub>h</sub> (RA) Row address hold time	t <sub>RAH</sub>	25	45		ns
t <sub>h</sub> (RLCA) Column address hold time after $\overline{\text{RAS}}$ low	t <sub>AR</sub>	275	315		ns
t <sub>h</sub> (CLD) Data hold time after $\overline{\text{CAS}}$ low	t <sub>DH</sub>	100	140		ns
t <sub>h</sub> (RLD) Data hold time after $\overline{\text{RAS}}$ low	t <sub>DHR</sub>	185	230		ns
t <sub>h</sub> (WLD) Data hold time after $\overline{\text{W}}$ low	t <sub>DH</sub>	145	185		ns
t <sub>h</sub> (RHrd) Read command hold time after $\overline{\text{RAS}}$ high	t <sub>RRH</sub>	135	175		ns
t <sub>h</sub> (CHrd) Read command hold time after $\overline{\text{CAS}}$ high	t <sub>RCH</sub>	40	80		ns
t <sub>h</sub> (CLW) Write command holdtime after $\overline{\text{CAS}}$ low	t <sub>WCH</sub>	285	325		ns
t <sub>h</sub> (RLW) Write command hold time after $\overline{\text{RAS}}$ low	t <sub>WCR</sub>	370	410		ns
t <sub>RLCH</sub> Delay time. $\overline{\text{RAS}}$ low to $\overline{\text{RAS}}$ high	t <sub>CSH</sub>	280	320		ns
t <sub>CHRL</sub> Delay time. $\overline{\text{CAS}}$ high to $\overline{\text{RAS}}$ low	t <sub>CRP</sub>	0	40		ns
t <sub>CLRH</sub> Delay time. $\overline{\text{CAS}}$ low to $\overline{\text{RAS}}$ high	t <sub>RSH</sub>	120	140		ns
t <sub>RLCL</sub> Delay time. $\overline{\text{RAS}}$ low to $\overline{\text{CAS}}$ low	t <sub>RCD</sub>	40	80	120	ns
t <sub>WLCL</sub> Delay time. $\overline{\text{W}}$ low to $\overline{\text{CAS}}$ low (early write cycle)	t <sub>WCS</sub>	10	40		ns
t <sub>rf</sub> refresh time interval	t <sub>REF</sub>	4	4		ms

\* All cycle times assume t = 5 ns

\*\*Page mode only



TMS9118/9128/9129 LOAD CIRCUIT FOR COMVID AND  
TMS9128/9129 R-Y, Y, B-Y SWITCHING CHARACTERISTICS



TMS9118/9128/9129 LOAD CIRCUIT FOR  
ALL OUTPUTS EXCEPT COMVID, R-Y, Y, B-Y

$R_L = 1.1 \text{ k}\Omega$   
 $C_L = 300 \text{ pF}$ 
} FOR  
 CO BUS  
  
 $R_L = 1.8 \text{ k}\Omega$   
 $C_L = 50 \text{ pF}$ 
} FOR  
 DRAM INTERFACE



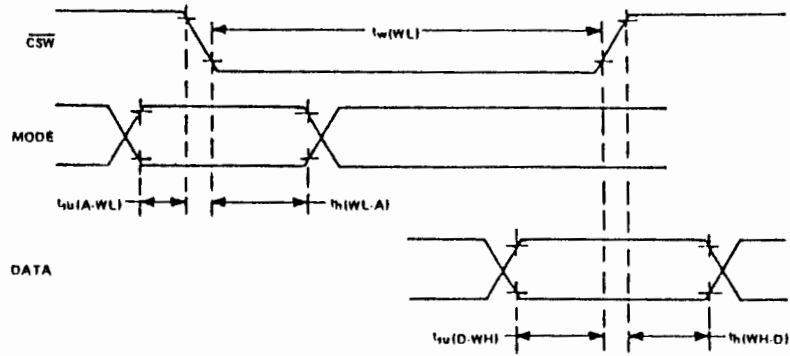
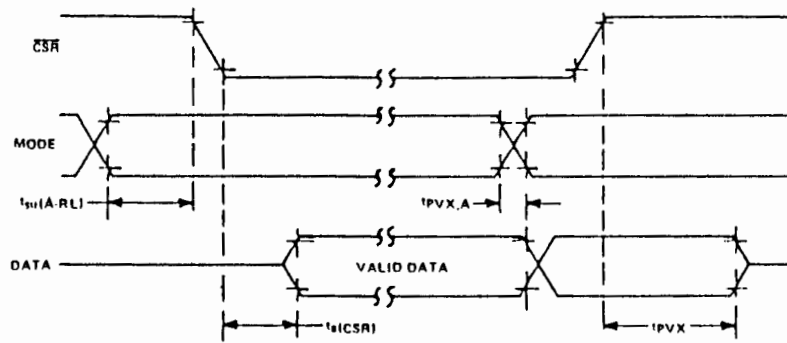


FIGURE 14 - CPU-VDP WRITE CYCLE FOR TMS9118/9128/9129

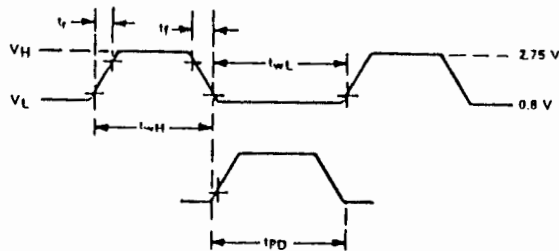
READ CYCLE



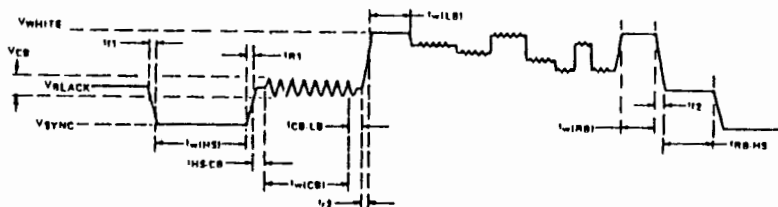
NOTE: All measurements are made at 10% and 90% points.

CPU-VDP READ CYCLE FOR TMS9118/9128/9129

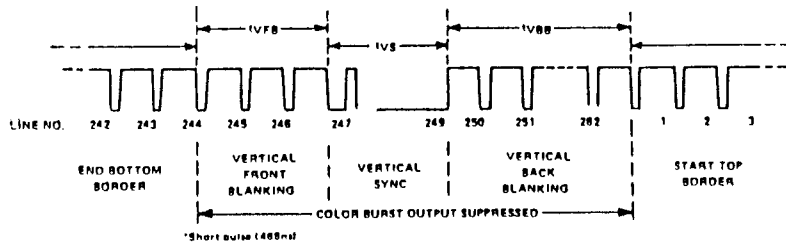
NOTE: For register and memory read/write cycle times refer to timing requirements given in preliminary electrical specifications.



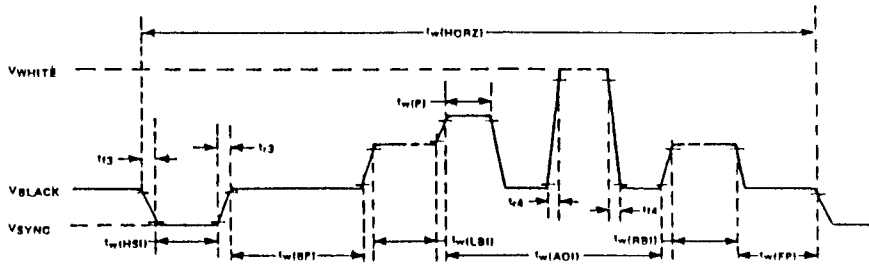
EXTERNAL CLOCK TIMING WAVEFORM



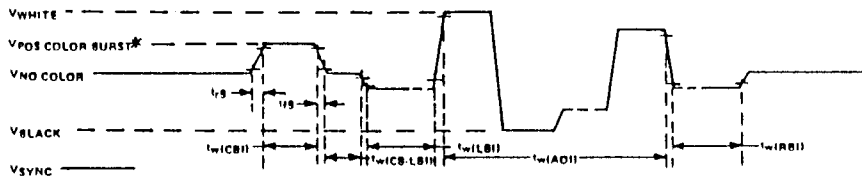
TMS9118 COMVID HORIZONTAL TIMING



TMS9118 VERTICAL TIMING

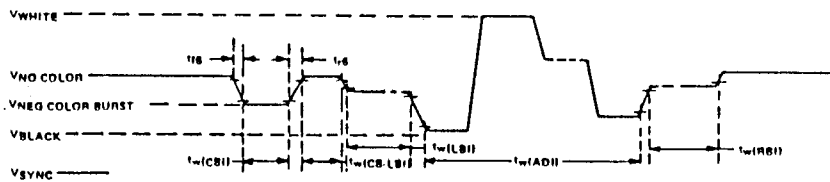


TMS9128/9129 Y HORIZONTAL TIMING

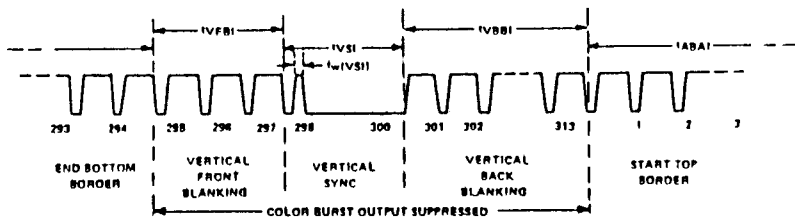


\* Absent for the TMS9128

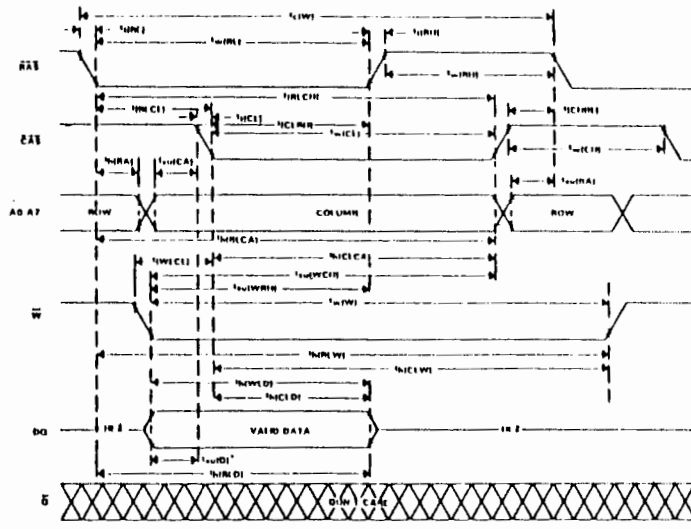
TMS9128/9129 R-Y HORIZONTAL TIMING



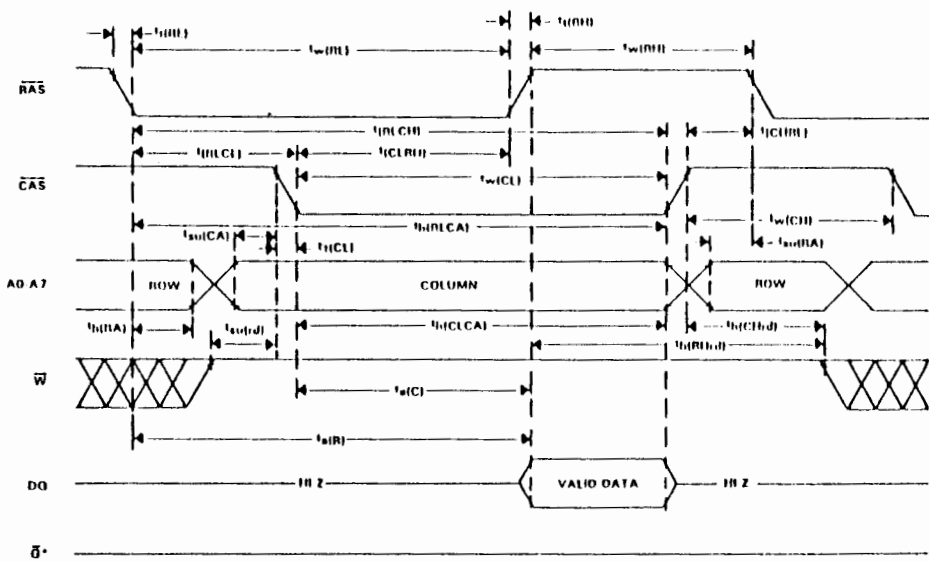
TMS9128/9129 B-Y HORIZONTAL TIMING



TMS9129 VERTICAL TIMING



VRAM EARLY WRITE CYCLE



\*  $\bar{G}$  Will be held at  $V_{SS}$  (GND) for all operations.

VRAM READ CYCLE

# ELECTRICAL SPECIFICATIONS

(AY-3-8910)

## Maximum Ratings

Storage Temperature ..... -55°C to +150°C  
 Operating Temperature ..... 0°C to +40°C  
 V<sub>cc</sub> and all other input and output  
 voltages with respect to V<sub>ss</sub> ..... -0.3V to +8.0V

Exceeding these ratings could cause permanent damage to these devices.  
 Functional operation at these conditions is not implied—operating conditions  
 are specified below.

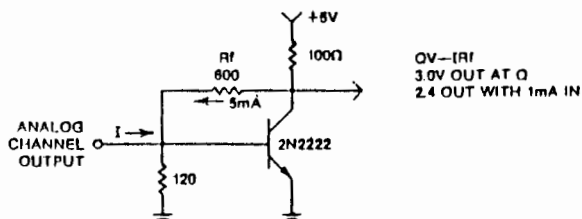
## Standard Conditions

V<sub>cc</sub> = +5V ±5%  
 V<sub>ss</sub> = GND  
 Operating temperature: 0°C to +40°C

## DC Characteristics

Characteristic	Sym	Min	Typ*	Max	Units	Conditions
<b>Input Logic Levels</b>						
Logic 0	V <sub>IL</sub>	0	—	0.6	V	
Logic 1	V <sub>IH</sub>	2.4	—	V <sub>cc</sub>	V	
<b>Input Leakage</b>						
Clock	—	—	—	20	μA	
BC1, BC2, BDIR	—	—	—	20	μA	
<b>Inputs with Pullups</b>						
A8, Reset	I <sub>IL</sub>	-10	-30	-100	μA	
<b>Inputs with Pulldowns</b>						
A9	I <sub>IH</sub>	5	10	40	μA	
<b>Input/Output with Pullup</b>						
A7-A0, B7-B0	I <sub>IL</sub>	-15	-30	-200	μA	
<b>Data/Address with Pullup</b>						
DA7-DA0	I <sub>IL</sub>	—	-150	-500	μA	
<b>Input/Output V<sub>OH</sub></b>						
A7-A0, B7-B0	V <sub>OL</sub>	2.4	—	V <sub>cc</sub>	V	40μA
<b>Data/Address V<sub>OH</sub></b>						
DA7-DA0	V <sub>OL</sub>	2.4	—	V <sub>cc</sub>	V	100μA
<b>Input/Output V<sub>OL</sub></b>						
A7-A0, B7-B0	V <sub>OL</sub>	0	—	0.5	V	1.6mA
<b>Data/Address V<sub>OL</sub></b>						
DA7-DA0	V <sub>OL</sub>	0	—	0.5	V	1.6mA
<b>Power Supply</b>						
V <sub>cc</sub>	I <sub>cc</sub>	—	45	75	mA	

\*Typical values are at +25°C and nominal voltages.



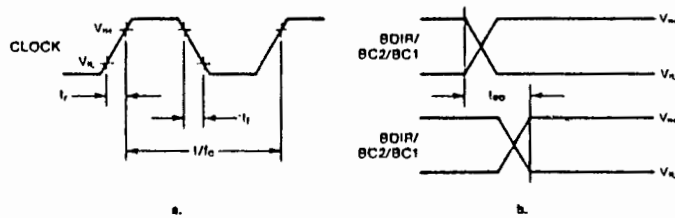
NOTE: CURRENT IN REDUCES R<sub>f</sub> CURRENT FLOW AND LOWERS OUTPUT VOLTAGE.

CURRENT TO VOLTAGE CONVERTER

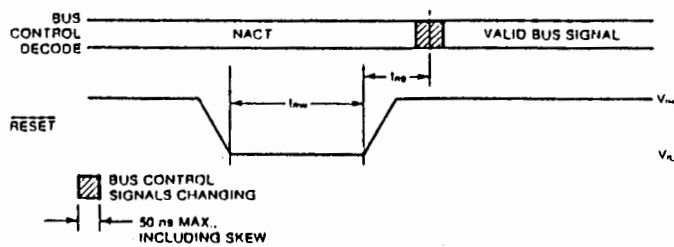
# AC Characteristics

Characteristic	Sym	Min.	Typ.*	Max.	Units	Conditions
<b>Clock Input</b>						
Frequency	$f_c$	1.0	—	2.0	MHz	} Fig. 35
Rise time	$t_r$	—	—	50	ns	
Fall time	$t_f$	—	—	50	ns	
Duty Cycle	—	25	50	75	%	
Bus Signals (BDIR, BC2, BC1)						
Associative Delay Time	$t_{ao}$	—	—	50	ns	} Fig. 36
<b>Reset</b>						
Reset Pulse Width	$t_{rw}$	500	—	—	ns	} Fig. 37
Reset to Bus Control Delay Time	$t_{re}$	100	—	—	ns	
<b>A9, A8, DA7-DA0 (Address Mode)</b>						} Fig. 37
Address Setup Time	$t_{as}$	400	—	—	ns	
Address Hold Time	$t_{ah}$	100	—	—	ns	} Fig. 38
<b>DA7-DA0 (Write Mode)</b>						
Write Data Pulse Width	$t_{dw}$	500	—	10,000	ns	
Write Data Setup Time	$t_{os}$	50	—	—	ns	} Fig. 38
Write Data Hold Time	$t_{oh}$	100	—	—	ns	
<b>DA7-DA0 (Read Mode)</b>						} Fig. 39
Read Data Access Time	$t_{da}$	—	250	500	ns	
<b>DA7-DA0 (Inactive Mode)</b>						
Tristate Delay Time	$t_{rs}$	—	100	200	ns	

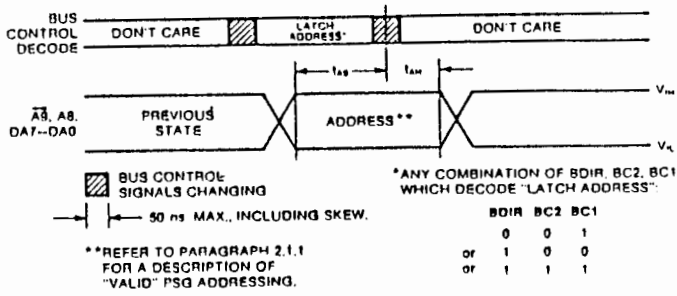
\* Typical values are at 25°C and nominal voltages.



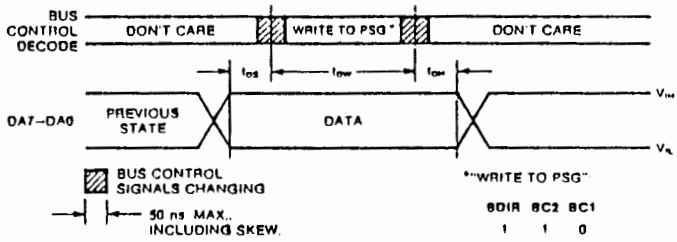
CLOCK AND BUS SIGNAL TIMING



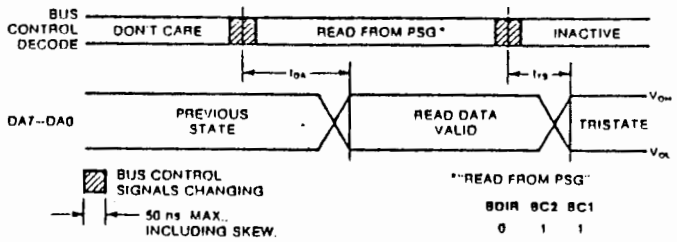
RESET TIMING



### LATCH ADDRESS TIMING

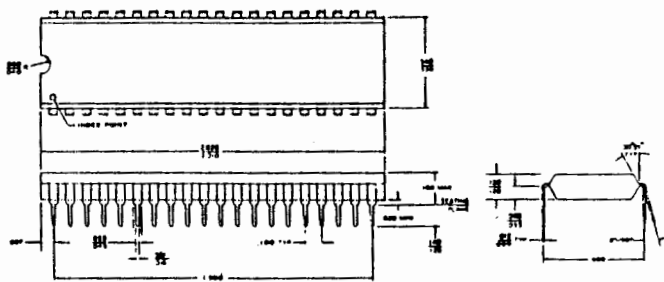


### WRITE DATA TIMING

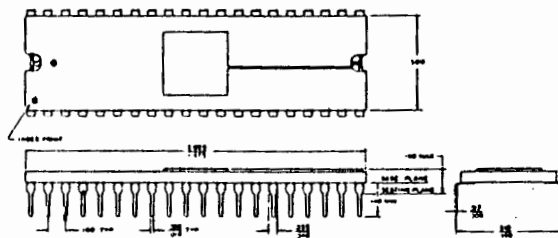


### READ DATA TIMING

## Package Outlines

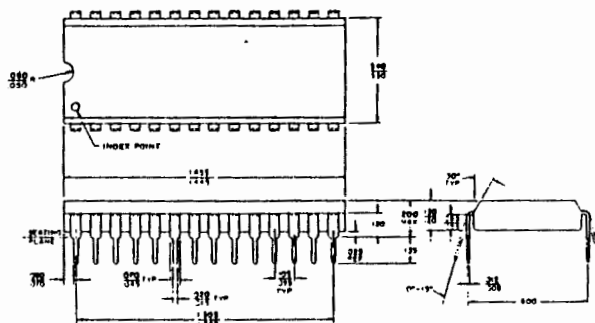


PLASTIC

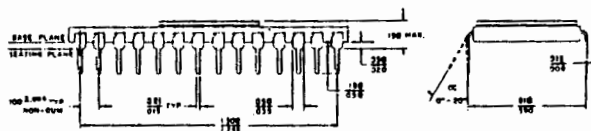
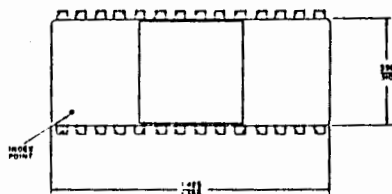


CERAMIC

40 LEAD DUAL IN LINE PACKAGES (for AY-3-8910)



PLASTIC



CERAMIC

28 LEAD DUAL IN LINE PACKAGES (for AY-3-8912)

# HM4864-2, HM4864-3 HM4864P-2, HM4864P-3

## 65536-word X 1-bit Dynamic Random Access Memory

The HM4864 is a 65,536-words by 1-bit, MOS random access memory circuit fabricated with HITACHI's double-poly N-channel silicon gate process for high performance and high functional density. The HM4864 uses a single transistor dynamic storage cell and dynamic control circuitry to achieve high speed and low power dissipation.

Multiplexed address inputs permit the HM4864 to be packaged in a standard 16 pin DIP on 0.3 inch centers.

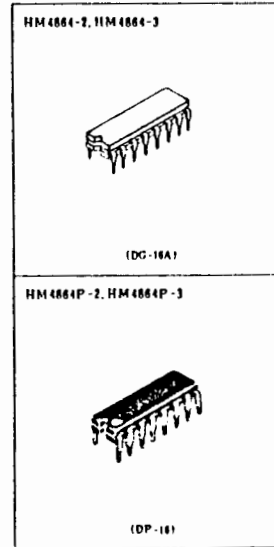
This package size provides high system bit densities and is compatible with widely available automated testing and insertion equipment. System oriented features include single power supply of +5V with  $\pm 10\%$  tolerance, direct interfacing capability with high performance logic families such as Schottky TTL, maximum input noise immunity to minimize "false triggering" of the inputs, on-chip address and data registers which eliminate the need for interface registers, and two chip select methods to allow the user to determine the appropriate speed/power characteristics of this memory system. The HM4864 also incorporates several flexible timing/operating modes.

In addition to the usual read, write, and read-modify-write cycles, the HM4864 is capable of delayed write cycles, page-mode operation and  $\overline{RAS}$ -only refresh.

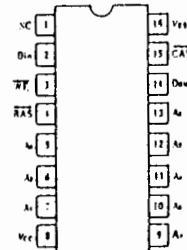
Proper control of the clock inputs ( $\overline{RAS}$ ,  $\overline{CAS}$ , and  $\overline{WE}$ ) allows common I/O capability, two dimensional chip selection, and extended page boundaries (when operating in page mode).

### FEATURES

- Recognized industry standard 16-pin configuration
- 180ns access time, 270ns cycle time (HM4864-2, HM4864P-2)
- 200ns access time, 335ns cycle time (HM4864-3, HM4864P-3)
- Single power supply of +5V  $\pm 10\%$  with a built-in  $V_{DD}$  generator
- Low Power; 330 mW active, 20 mW standby (max)
- The inputs TTL compatible, low capacitance, and protected against static charge
- Output data controlled by  $\overline{CAS}$  and unlatched at end of cycle to allow two dimensional chip selection and extended page boundary
- Common I/O capability using "early write" operation
- Read-Modify-Write,  $\overline{RAS}$ -only refresh, and Page-mode capability
- 128 refresh cycle



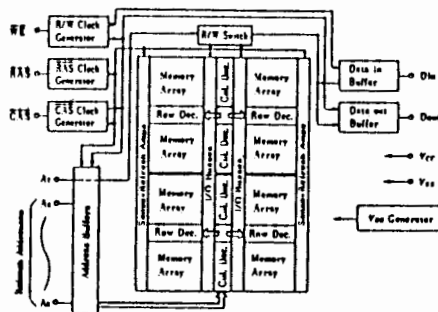
### PIN ARRANGEMENT



(Top View)

A <sub>0</sub> -A <sub>7</sub>	Address Inputs
$\overline{CAS}$	Column Address Strobe
Di	Data In
Dout	Data Out
$\overline{RAS}$	Row Address Strobe
$\overline{WE}$	Read/Write Input
V <sub>CC</sub>	Power (+5V)
V <sub>SS</sub>	Ground
A <sub>8</sub> -A <sub>9</sub>	Refresh Address Input

### FUNCTIONAL BLOCK DIAGRAM



### ABSOLUTE MAXIMUM RATINGS

- Voltage on any pin relative to  $V_{SS}$  ..... -1.0 to +7V
- Operating Temperature,  $T_a$  (Ambient) ..... 0 to +70°C
- Storage Temperature (Ambient) ..... -65 to +150°C (Crdip)  
-55 to +125°C (Plastic)
- Short-circuit Output Current ..... 50 mA
- Power Dissipation ..... 1 W



■ RECOMMENDED DC OPERATING CONDITIONS ( $T_a=0$  to  $+70^\circ\text{C}$ )

Parameter	Symbol	min	typ	max	Unit	Notes
Supply Voltage	$V_{CC}$	4.5	5.0	5.5	V	1
	$V_{SS}$	0	0	0	V	
Input High Voltage	$V_{IH}$	2.4	—	6.5	V	1
Input Low Voltage	$V_{IL}$	-1.0	—	0.8	V	1

■ DC ELECTRICAL CHARACTERISTICS ( $T_a=0$  to  $+70^\circ\text{C}$ ,  $V_{CC}=5\text{V} \pm 10\%$ ,  $V_{SS}=0\text{V}$ )

Parameter	Symbol	min	max	Unit	Notes
<b>OPERATING CURRENT</b>					
Average Power Supply Operating Current (RAS, CAS Cycling; $f_{CL} = \text{min.}$ )	$I_{CC1}$	—	80	mA	2, 4
<b>STANDBY CURRENT</b>					
Power Supply Standby Current (RAS = $V_{IH}$ , DOUT = High Impedance)	$I_{CC2}$	—	3.5	mA	2
<b>REFRESH CURRENT</b>					
Average Power Supply Current, Refresh Mode (RAS Cycling, CAS = $V_{IL}$ ; $f_{CL} = \text{min.}$ )	$I_{CC3}$	—	15	mA	2, 4
<b>PAGE MODE CURRENT</b>					
Average Power Supply Current, Page-mode Operation (RAS = $V_{IL}$ , CAS Cycling; $f_{CL} = \text{min.}$ )	$I_{CC4}$	—	15	mA	2, 4
<b>INPUT LEAKAGE</b>					
Input Leakage Current, any Input ( $V_i = 0$ to $+8.5\text{V}$ , all other pins not under test = $0\text{V}$ )	$I_{L1}$	-10	10	$\mu\text{A}$	
<b>OUTPUT LEAKAGE</b>					
Output Leakage Current (DOUT is disabled, $V_{OH} = 0$ to $+8.5\text{V}$ )	$I_{L2}$	-10	10	$\mu\text{A}$	3
<b>OUTPUT LEVELS</b>					
Output High (Logic 1) Voltage ( $I_{OH} = -5\text{mA}$ )	$V_{OH}$	2.4	$V_{CC}$	V	
Output Low (Logic 0) Voltage ( $I_{OL} = 4.2\text{mA}$ )	$V_{OL}$	0	0.4	V	

NOTES

- All voltages referenced to  $V_{SS}$ .
- $I_{CC}$  depends on output loading condition when the device is selected.  $I_{CC \text{ max.}}$  is specified at the output open condition.
- $I_{L2}$  consists of leakage current only.
- Current depends on cycle rate: maximum current is measured at the fastest cycle rate.

■ AC ELECTRICAL CHARACTERISTICS

Parameter	Symbol	typ	max	Unit	Notes
Input Capacitance (A <sub>1</sub> -A <sub>16</sub> , D <sub>in</sub> )	$C_{in1}$	—	7	pF	1
Input Capacitance (RAS, CAS, WE)	$C_{in2}$	—	10	pF	1
Output Capacitance (DOUT)	$C_{out}$	—	7	pF	1, 2

NOTES

- Capacitance measured with Boonton Meter or effective capacitance measuring method.
- CAS =  $V_{IH}$  to disable DOUT.

■ ELECTRICAL CHARACTERISTICS AND RECOMMENDED AC OPERATING CONDITIONS (1, 2)  
( $T_a=0$  to  $+70^\circ\text{C}$ ,  $V_{CC}=5\text{V} \pm 10\%$ ,  $V_{SS}=0\text{V}$ )

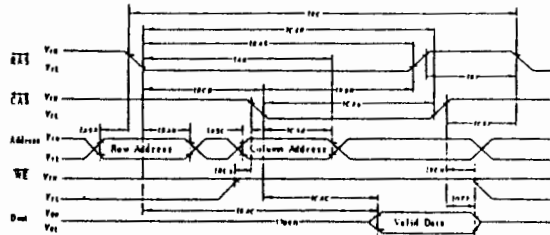
Parameter	Symbol	HM4864-2/P-2		HM4864-3/P-3		Unit	Notes
		min	max	min	max		
Random Read or Write Cycle Time	$t_{RC}$	270	—	335	—	ns	
Read/Write Cycle Time	$t_{mRC}$	270	—	335	—	ns	
Page Mode Cycle Time	$t_{PC}$	170	—	225	—	ns	
Access Time from RAS	$t_{AAC}$	—	150	—	200	ns	4, 8
Access Time from CAS	$t_{CAC}$	—	100	—	135	ns	5, 8
Output Buffer Turn-off Delay	$t_{OBT}$	0	40	0	50	ns	7
Transition Time (Rise and Fall)	$t_T$	3	35	3	50	ns	3
RAS Precharge Time	$t_{AP}$	100	—	120	—	ns	
RAS Pulse Width	$t_{ASW}$	150	10000	200	10000	ns	
RAS Hold Time	$t_{ASH}$	100	—	135	—	ns	
CAS Pulse Width	$t_{CSW}$	100	—	135	—	ns	
CAS Hold Time	$t_{CSH}$	150	—	200	—	ns	
RAS to CAS Delay Time	$t_{ACD}$	20	50	25	65	ns	8
CAS to RAS Precharge Time	$t_{ACP}$	-20	—	-20	—	ns	
Row Address Set-up Time	$t_{ASU}$	0	—	0	—	ns	
Row Address Hold Time	$t_{ASH}$	20	—	25	—	ns	
Column Address Set-up Time	$t_{CSU}$	-10	—	-10	—	ns	
Column Address Hold Time	$t_{CSH}$	45	—	55	—	ns	
Column Address Hold Time referenced to RAS	$t_{CAH}$	95	—	120	—	ns	
Read Command Set-up Time	$t_{RCU}$	0	—	0	—	ns	
Read Command Hold Time	$t_{RCH}$	0	—	0	—	ns	
Write Command Hold Time	$t_{WCH}$	45	—	55	—	ns	
Write Command Hold Time referenced to RAS	$t_{WCH}$	95	—	120	—	ns	
Write Command Pulse Width	$t_{WCP}$	45	—	55	—	ns	
Write Command to RAS Lead Time	$t_{WCL}$	45	—	55	—	ns	
Write Command to CAS Lead Time	$t_{WCL}$	45	—	55	—	ns	
Data-in Set-up Time	$t_{DSU}$	0	—	0	—	ns	9
Data-in Hold Time	$t_{DSH}$	45	—	55	—	ns	9
Data-in Hold Time referenced to RAS	$t_{DSH}$	95	—	120	—	ns	
CAS Precharge Time (for Page-mode Cycle Only)	$t_{CPC}$	80	—	80	—	ns	
Refresh Period	$t_{RFR}$	—	2	—	2	ms	
Write Command Set-up Time	$t_{WCU}$	-20	—	-20	—	ns	10
CAS to WE Delay	$t_{CWD}$	60	—	80	—	ns	10
RAS to WE Delay	$t_{RWD}$	110	—	145	—	ns	10
RAS Precharge to CAS Hold Time	$t_{RPH}$	0	—	0	—	ns	

NOTES

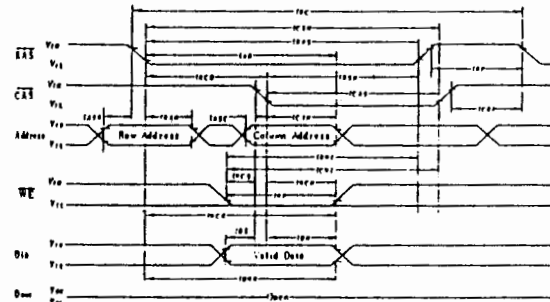
1. AC measurements assume  $t_T = 5ns$ .
2. 8 cycles are required after power-on or prolonged periods (greater than 2ms) of RAS inactivity before proper device operation is achieved. Any 8 cycles which perform refresh are adequate for this purpose.
3.  $V_{IH}$  (min) and  $V_{IL}$  (max) are reference levels for measuring timing of input signals. Also, transition times are measured between  $V_{IH}$  and  $V_{IL}$ .
4. Assumes that  $t_{RCD} \leq t_{RCD}(max)$ . If  $t_{RCD}$  is greater than the maximum recommended value shown in this table  $t_{RAC}$  exceeds the value shown.
5. Assumes that  $t_{RCD} \geq t_{RCD}(max)$ .
6. Measured with a load circuit equivalent to 2TTL loads and 100 pF.
7.  $t_{OFF}$  (max) defines the time at which the output achieves the open circuit condition and is not referenced to output voltage levels.
8. Operation with the  $t_{RCD}(max)$  limit insures that  $t_{RAC}(max)$  can be met.  $t_{RCD}(max)$  is specified as a reference point only; if  $t_{RCD}$  is greater than the specified  $t_{RCD}(max)$  limit, then access time is controlled exclusively by  $t_{CAC}$ .
9. These parameters are reference to  $\overline{CAS}$  leading edge in early write cycles and to  $\overline{WE}$  leading edge in delayed write or read-modify-write cycles.
10.  $t_{WCS}$ ,  $t_{CWD}$  and  $t_{RWD}$  are not restrictive operating parameters. They are included in the data sheet as electrical characteristics only: If  $t_{WCS} \geq t_{WCS}(min)$ , the cycle is an early write cycle and the data out pin will remain open circuit (high impedance) throughout the entire cycle; If  $t_{CWD} \geq t_{CWD}(min)$  and  $t_{RWD} \geq t_{RWD}(min)$  the cycle is a read/write and the data output will contain data read from the selected cell; If neither of the above sets of conditions is satisfied the condition of the data out (at access time) is indeterminate.

■ TIMING WAVEFORMS

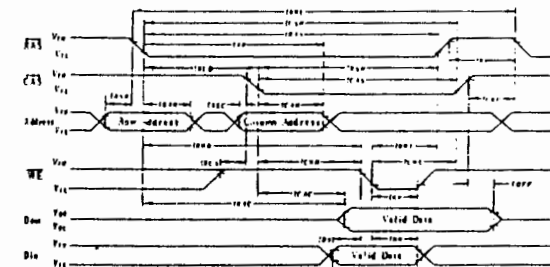
● READ CYCLE



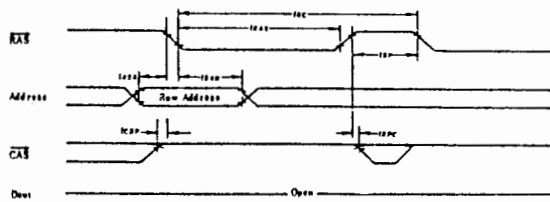
● WRITE CYCLE



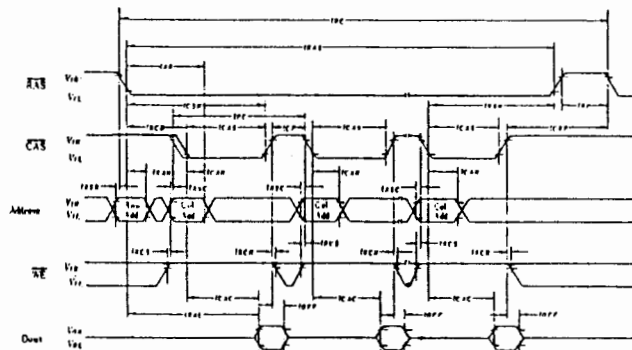
● READ-WRITE/READ-MODIFY-WRITE CYCLE



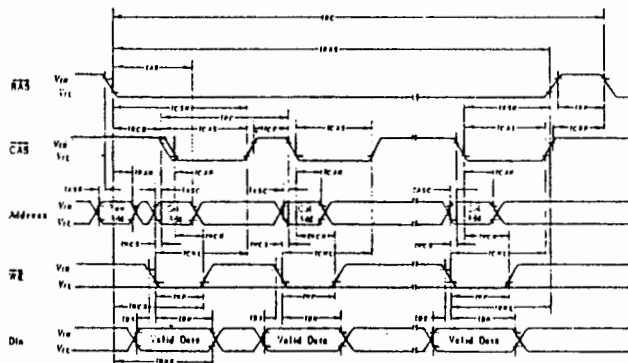
● "RAS-ONLY" REFRESH CYCLE



● PAGE MODE READ CYCLE

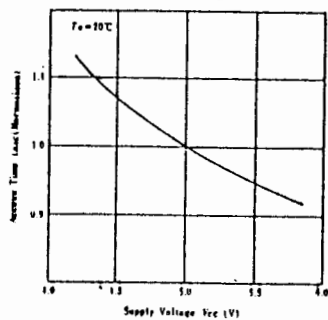


● PAGE MODE WRITE CYCLE

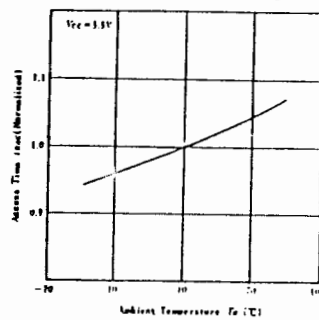


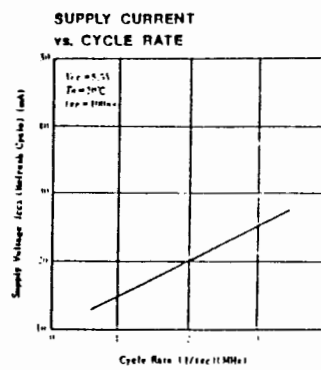
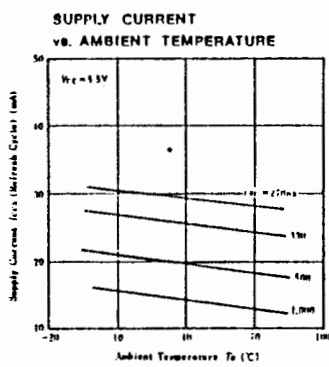
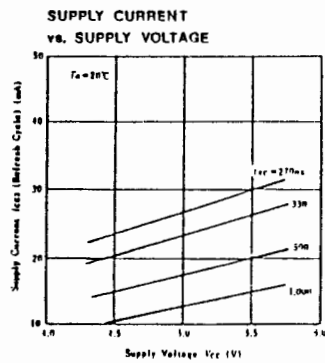
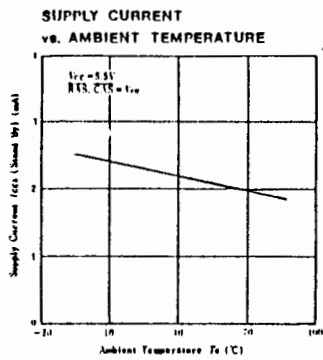
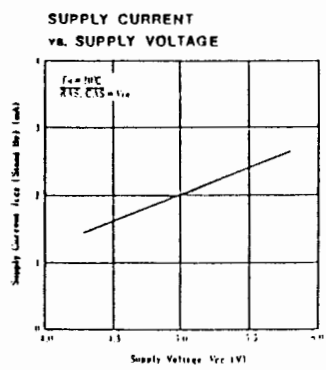
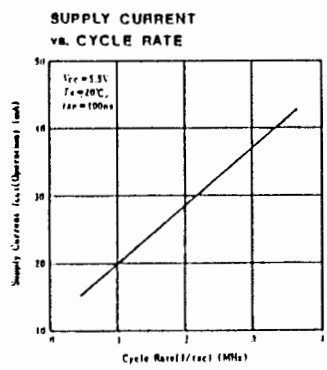
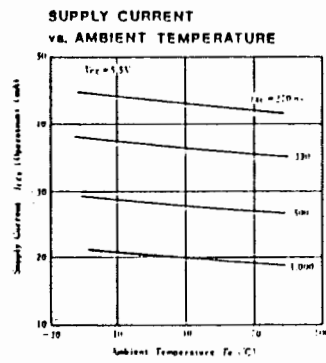
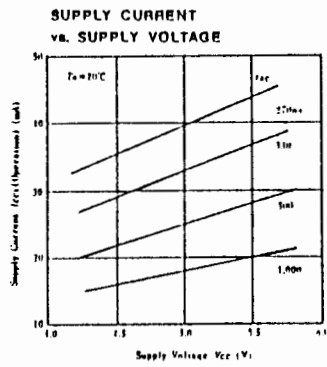
■ TYPICAL CHARACTERISTICS

ACCESS TIME  
vs. SUPPLY VOLTAGE

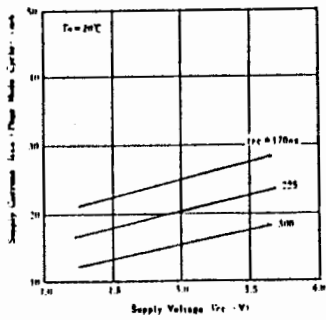


ACCESS TIME  
vs. AMBIENT TEMPERATURE

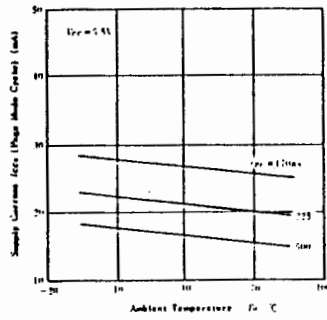




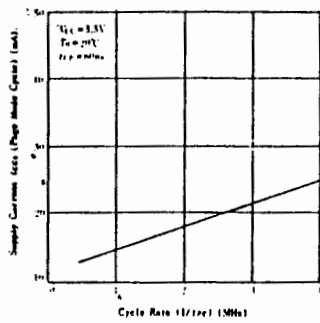
**SUPPLY CURRENT  
vs. SUPPLY VOLTAGE**



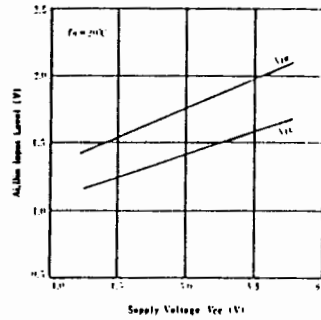
**SUPPLY CURRENT  
vs. AMBIENT TEMPERATURE**



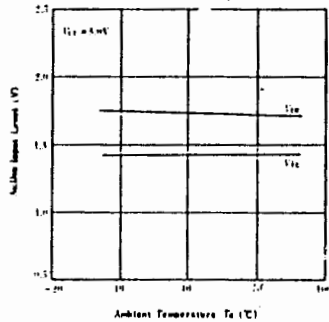
**SUPPLY CURRENT  
vs. CYCLE RATE**



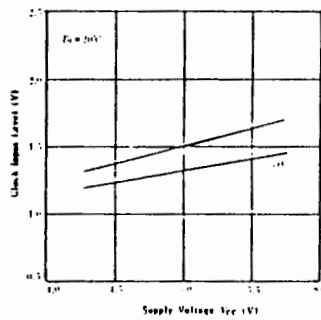
**INPUT LEVEL  
vs. SUPPLY VOLTAGE**



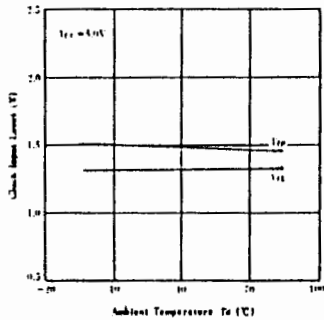
**INPUT LEVEL  
vs. AMBIENT TEMPERATURE**

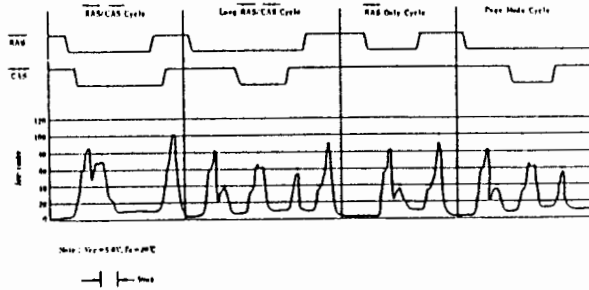


**CLOCK INPUT LEVEL  
vs. SUPPLY VOLTAGE**



**CLOCK INPUT LEVEL  
vs. AMBIENT TEMPERATURE**





## APPLICATION INFORMATION

### POWER ON

An initial pause of 500  $\mu$ s is required after power-up and a minimum of eight (8) initialization cycle, (any combination of cycles containing a RAS clock such as RAS-only refresh) must follow an initial pause.

The  $V_{CC}$  current ( $I_{CC}$ ) requirement of the HM4864 during power on is, however, dependent upon the input levels ( $\overline{RAS}$ ,  $\overline{CAS}$ ) and the rise time of  $V_{CC}$ , as shown in Fig. 1.

### READ CYCLE

A read cycle begins with addresses stable and a negative going transition of  $\overline{RAS}$ . The time delay between the stable address and the start of  $\overline{RAS}$ -on is controlled by parameter  $t_{ASR}$ . Following the time when  $\overline{RAS}$  reaches its low level, the row address must be held stable long enough to be captured. This controlling parameter is  $t_{RAH}$ . Following this interval, the address can be changed from row address to column address. When the column address is stable,  $\overline{CAS}$  can be turned on. The leading edge of  $\overline{CAS}$  is controlled by parameter  $t_{ACD}$ . The basic limit on the  $\overline{CAS}$  leading edge is that  $\overline{CAS}$  can not start until the column address is stable, and this is controlled by parameter  $t_{ASC}$ . The column address must be held stable long enough to be captured. The controlling parameter is  $t_{CAH}$ . Note that  $t_{ACD}$  (max) is not an operating limit of the HM4864 though its specification is listed on the data sheets. If  $\overline{CAS}$  becomes on later than  $t_{ACD}$  (max), the access time from  $\overline{RAS}$  will be increased by the time which  $t_{ACD}$  exceeds  $t_{ACD}$  (max).

Following the time when  $\overline{CAS}$  reaches its low level, the data-out pin remains in a high impedance state until a valid data appears. This parameter is  $t_{CAC}$ -access time from  $\overline{CAS}$ . The access time from  $\overline{RAS}$ - $t_{RAC}$ -is the time from  $\overline{RAS}$ -on to valid Dout.

The minimum value of  $t_{RAC}$  is derived as the sum of  $t_{ACD}$  (max) and  $t_{CAC}$ .

The selected output data is held valid internally until  $\overline{CAS}$  becomes high, and then Dout pin becomes high impedance. This parameter is  $t_{OFF}$ .

### WRITE CYCLE

A write cycle is performed by bringing  $\overline{WE}$  low before or during  $\overline{CAS}$ -on.

Two different write cycles can be defined as:

Write cycle-Write data are available at the beginning of the  $\overline{CAS}$ -on so that the write operation starts at the beginning. In this mode, Dout and  $\overline{WE}$  signal times are not in any critical path for determining cycle time.

Following the time when  $\overline{WE}$  reaches its low level,  $\overline{WE}$  must be held stable long enough to be captured. This  $\overline{WE}$ -on pulse duration is called  $t_{WPP}$ . The time required to capture write data in a latch is called  $t_{DH}$ . This cycle is called an "early write".

Read Write cycle-This cycle starts as a read cycle, but as soon as the device specification is met, a write cycle is initiated.

$\overline{WE}$  and Din are delayed until after Dout. This cycle is called a "delayed write". A "Read-modify-write" cycle is a variation of this operation. In this mode, Din and  $\overline{WE}$  become critical path signals for determining cycle time.

### REFRESH

Refresh of the HM4864 is accomplished by performing a memory cycle at each of the 128 row addresses within each two millisecond time interval. A0 to A6 are refresh address pin compatible with standard 16K RAM (HM4716A, HM4816A). During refresh, either  $V_{IL}$  or  $V_{IH}$  is permitted for A7. Any cycle in which  $\overline{RAS}$  signal occurs refreshes the entire selected row.  $\overline{RAS}$ -only refresh results in substantial reduction in operating power. This reduction in power is reflected in the  $I_{CC2}$  specification.

### PAGE MODE

Page mode operation allows faster successive memory operations at multiple column locations of the same row address with increased speed.

This is done by strobing the row address into the chip and maintaining  $\overline{RAS}$  at a logic low throughout all successive  $\overline{CAS}$  memory cycles in which the row address is latched. As the time normally required for strobing a new row address is eliminated, access and cycle times can be decreased and the operating power is reduced. These are specifications.

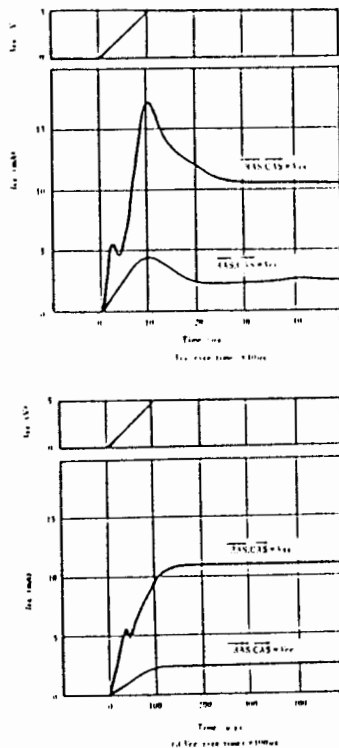


Fig. 1  $I_{CC}$  vs.  $V_{CC}$  during power up.

• CLOCK-OFF TIMING

$\overline{RAS}$  and  $\overline{CAS}$  must stay on for Dout stabilized to valid data. In the case of  $\overline{CAS}$ , this is controlled by parameter  $t_{CAS}$  (min).

In the case of  $\overline{RAS}$ , this is controlled by parameter  $t_{CAS}$  (min). Following the end of  $\overline{RAS}$ ,  $\overline{CAS}$  must stay off long enough to precharge internal circuits. The only parameter of concern is  $t_{RP}$ . Normally  $\overline{CAS}$  is not required to be off for minimum time of  $t_{CRP}$ . However, in a page mode memory operation, there is a  $t_{CP}$  (min) specification to control the  $\overline{CAS}$ -off time.

• DATA OUTPUT

Dout is three-state TTL compatible with a fan-out of two standard TTL loads.

When  $\overline{CAS}$  is high, Dout is in a high impedance state. When  $\overline{CAS}$  is low, valid data appears after  $t_{CAC}$  at a read cycle, and Dout is not valid as an early-write cycle.

**TMS4416, SMJ4416**  
**16,384-WORD BY 4-BIT DYNAMIC RAM**

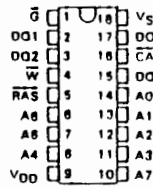
- 16,384 X 4 Organization
- Single +5-V Supply (10% Tolerance)
- Performance Ranges:

	ACCESS TIME	ACCESS ROW ADDRESS (MAX)	READ OR WRITE CYCLE (MIN)	READ-MODIFY-WRITE CYCLE (MIN)
'4416-12	120 ns	70 ns	230 ns	320 ns
'4416-16	150 ns	80 ns	260 ns	330 ns
'4416-20	200 ns	120 ns	330 ns	440 ns

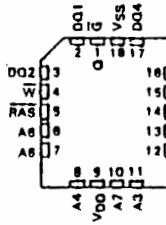
- Available Temperature Ranges\*:
  - S . . . -55°C to 100°C
  - E . . . -40°C to 85°C
  - L . . . 0°C to 70°C

- Long Refresh Period . . . 4 milliseconds
- Low Refresh Overhead Time . . . As Low As 1.7% of Total Refresh Period
- All Inputs, Outputs, Clocks Fully TTL Compatible
- 3-State Unlatched Outputs
- Early Write or  $\bar{Q}$  to Control Output Buffer Impedance
- Page-Mode Operation for Faster Access
- Low Power Dissipation
  - Operating . . . 200 mW (TYP)
  - Standby . . . 17.5 mW (TYP)
- New SMOS (Scaled-MOS) N-Channel Technology

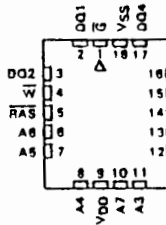
TMS4416 . . . NL PACKAGE  
 SMJ4416 . . . JD PACKAGE  
 (TOP VIEW)



TMS4416 . . . FPL PACKAGE  
 (TOP VIEW)



SMJ4416 . . . FG PACKAGE  
 (TOP VIEW)



PIN NOMENCLATURE	
A0-A7	Address Inputs
CAS	Column Address Strobe
DQ1-DQ4	Data In/Data Out
$\bar{Q}$	Output Enable
RAS	Row Address Strobe
VDD	+5-V Supply
VSS	Ground
W	Write Enable

**description**

The '4416 is a high-speed, 65,536-bit, dynamic, random-access memory, organized as 16,384 words of 4 bits each. It employs state-of-the-art SMOS (scaled MOS) N-channel double-level polysilicon gate technology for very high performance combined with low cost and improved reliability.

The '4416 features RAS access times to 120 ns maximum. Power dissipation is 200 mW typical operating, 17.5 mW typical standby.

New SMOS technology permits operation from a single +5-V supply, reducing system power supply and decoupling requirements, and easing board layout. IDD peaks have been reduced to 80 mA typical, and a -1-V input voltage undershoot can be tolerated, minimizing system noise considerations. Input clamp diodes are used to ease system design.

Refresh period is extended to 4 milliseconds, and during this period each of the 256 rows must be strobed with RAS in order to retain data. CAS can remain high during the refresh sequence to conserve power.

All inputs and outputs, including clocks, are compatible with Series 54/74 TTL. All address lines and data-in are latched on chip to simplify system design. Data-out is unlatched to allow greater system flexibility.

\*H temperature range (-55°C to 125°C) to be available in future.

The TMS4416 is offered in 16-pin plastic dual-in line and 18-pin plastic chip carrier packages. It is guaranteed for operation from 0°C to 70°C. The SMJ4416 is offered in 18-pin ceramic side-brace dual-in-line and 18-pin ceramic chip carrier packages. It is available in -55°C to 100°C and -40°C to 85°C temperature ranges. Dual-in-line packages are designed for insertion in mounting-hole rows on 300-mil (7.62 mm) centers.

**operation**

**address (A0 through A7)**

Fourteen address bits are required to decode 1 of 16,384 storage locations. Eight row-address bits are set up on pins A0 through A7 and latched onto the chip by the row-address strobe (RAS). Then the six column-address bits are set up on pins A1 through A6 and latched onto the chip by the column-address strobe (CAS). All addresses must be stable on or before the falling edges of RAS and CAS. RAS is similar to a chip enable in that it activates the sense amplifiers as well as the row decoder. CAS is used as a chip select activating the column decoder and the input and output buffers.

**write enable ( $\bar{W}$ )**

The read or write mode is selected through the write enable (W) input. A logic high on the W input selects the read mode and a logic low selects the write mode. The write enable terminal can be driven from standard TTL circuits without a pull-up resistor. The data input is disabled when the read mode is selected. When W goes low prior to CAS, data-out will remain in the high-impedance state following a write cycle with Q-bar grounded.

**data-in (DQ1 through DQ4)**

Data is written during a write or read-modify write cycle. Depending on the mode of operation, the falling edge of CAS or W strobes data into the on-chip data latches. These latches can be driven from standard TTL circuits without a pull-up resistor. In an early-write cycle, W is brought low prior to CAS and the data is strobed in by CAS with setup and hold times referenced to this signal. In a delayed write or read-modify-write cycle, CAS will already be low, thus the data will be strobed in by W with setup and hold times referenced to this signal. In delayed or read-modify-write, Q-bar must be high to bring the output buffers to high impedance prior to impressing data on the I/O lines.



**data-out (OO1 through OO4)**

The three-state output buffer provides direct TTL compatibility (no pull-up resistor required) with a fan-out of two Series 54/74 TTL loads. Data-out is the same polarity as data-in. The output is in the high-impedance (floating) state until  $\overline{CAS}$  is brought low. In a read cycle the output goes active after the access time interval  $t_{a(C)}$  that begins with the negative transition of  $\overline{CAS}$  as long as  $t_{a(R)}$  and  $t_{a(E)}$  are satisfied. The output becomes valid after the access time has elapsed and remains valid while  $\overline{CAS}$  and  $\overline{G}$  are low.  $\overline{CAS}$  or  $\overline{G}$  going high returns it to a high impedance state. In an early-write cycle, the output is always in the high impedance state. In a delayed-write or read-modify-write cycle, the output must be put in the high impedance state prior to applying data to the DO input. This is accomplished by bringing  $\overline{G}$  high prior to applying data, thus satisfying  $t_{GHQ}$ .

**output enable ( $\overline{G}$ )**

The  $\overline{G}$  controls the impedance of the output buffers. When  $\overline{G}$  is high, the buffers will remain in the high impedance state. Bringing  $\overline{G}$  low during a normal cycle will activate the output buffers putting them in the low impedance state. It is necessary for both  $\overline{RAS}$  and  $\overline{CAS}$  to be brought low for the output buffers to go into the low impedance state. Once in the low impedance state, they will remain in the low impedance state until  $\overline{G}$  or  $\overline{CAS}$  is brought high.

**refresh**

A refresh operation must be performed at least every four milliseconds to retain data. Since the output buffer is in the high-impedance state unless  $\overline{CAS}$  is applied, the  $\overline{RAS}$ -only refresh sequence avoids any output during refresh. Strobing each of the 256 row addresses (A0 through A7) with  $\overline{RAS}$  causes all bits in each row to be refreshed.  $\overline{CAS}$  can remain high (inactive) for this refresh sequence to conserve power.

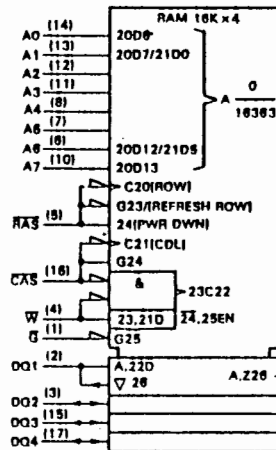
**page mode**

Page mode operation allows effectively faster memory access by keeping the same row address and strobing successive column addresses onto the chip. Thus, the time required to setup and strobe sequential row addresses for the same page is eliminated. To extend beyond the 64 column locations on a single RAM, the row address and  $\overline{RAS}$  are applied to multiple 16K x 4 RAMs.  $\overline{CAS}$  is then decoded to select the proper RAM.

**power-up**

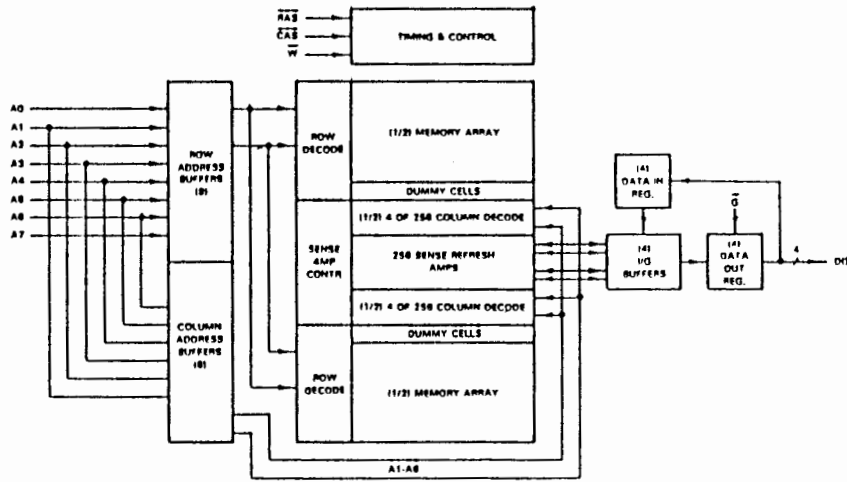
After power-up, the power supply must remain at its steady-state value for 1 ms. In addition, the  $\overline{RAS}$  input must remain high for 100  $\mu$ s immediately prior to initialization. Initialization consists of performing eight  $\overline{RAS}$  cycles before proper device operation is achieved.

**logic symbol†**



† This symbol is in accordance with IEEE Std 91/ANSI Y32.14 and recent decisions by IEEE and IEC.

functional block diagram



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)<sup>†</sup>

Voltage on any pin except V <sub>DD</sub> and data out (see Note 1)	-1.5 V to 10 V
Voltage on V <sub>DD</sub> supply and data out with respect to V <sub>SS</sub>	-1 V to 8 V
Short circuit output current	50 mA
Power dissipation	1 W
Operating free-air temperature range: TMS <sup>†</sup>	0°C to 70°C
Operating case temperature range: SMJ <sup>†</sup> - S version	-55°C to 100°C
- E version	-40°C to 85°C
Storage temperature range	-65°C to 150°C

<sup>†</sup> Stresses beyond 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 beyond those indicated in the "Recommended Operating Conditions" section of this specification is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: All voltage values in this data sheet are with respect to V<sub>SS</sub>.

recommended operating conditions

PARAMETER	TMS4418			UNIT
	MIN	NOM	MAX	
Supply voltage, V <sub>DD</sub>	4.5	5	5.5	V
Supply voltage, V <sub>SS</sub>	0			V
High-level input voltage, V <sub>IH</sub>	V <sub>DD</sub> = 4.5 V		2.4	4.8
	V <sub>DD</sub> = 5.5 V		2.4	5.8
Low-level input voltage, V <sub>IL</sub> (see Note 2)	V <sub>IK</sub>		0.8	V
Operating free-air temperature, T <sub>A</sub>	0	70		°C

NOTE 2: The algebraic convention, where the more negative (less positive) limit is designated as minimum, is used in this data sheet for logic voltage levels only.

electrical characteristics over full ranges of recommended operating conditions (unless otherwise noted)

PARAMETER	TEST CONDITIONS	TMS4418-12		UNIT
		MIN	TYP <sup>†</sup>	
V <sub>IK</sub>	Input clamp voltage I <sub>I</sub> = -16 mA, see Figure 1	-1.2		V
V <sub>OH</sub>	High-level output voltage I <sub>OH</sub> = -2 mA	2.4		V
V <sub>OL</sub>	Low-level output voltage I <sub>OL</sub> = 4.2 mA	0.4		V
I <sub>I</sub>	Input current (leakage) V <sub>I</sub> = 0 V to 5.8 V, V <sub>DD</sub> = 5 V, All other pins = 0 V	±10		µA
I <sub>O</sub>	Output current (leakage) V <sub>O</sub> = 0.4 V to 5.5 V, V <sub>DD</sub> = 5 V, C <sub>AS</sub> high	±10		µA
I <sub>DD1</sub>	Average operating current during read or write cycle At t <sub>c</sub> = minimum cycle	54		mA
I <sub>DD2</sub> <sup>†</sup>	Standby current After 1 memory cycle, R <sub>AS</sub> and C <sub>AS</sub> high	3.5	5	mA
I <sub>DD3</sub>	Average refresh current t <sub>c</sub> = minimum cycle, R <sub>AS</sub> cycling, C <sub>AS</sub> high	46		mA
I <sub>DD4</sub>	Average page-mode current t <sub>c</sub> (P) = minimum cycle, R <sub>AS</sub> low, C <sub>AS</sub> cycling	46		mA

<sup>†</sup> All typical values are at T<sub>A</sub> = 25°C and nominal supply voltages.

V<sub>IK</sub> is -0.6 V on all inputs.

switching characteristics over recommended supply voltage range and operating free-air temperature range

PARAMETER	TEST CONDITIONS	ALT. SYMBOL	TMS4416-12		UNIT
			MIN	MAX	
$t_{a1}(C)$ Access time from $\overline{CAS}$	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{CAC}$		70	ns
$t_{a1}(R)$ Access time from $\overline{RAS}$	$t_{RLCL} = \text{MAX}$ , $C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{RAC}$		120	ns
$t_{a1}(Q)$ Access time after $\overline{Q}$ low	$C_L = 100$ pF, Load = 2 Series 74 TTL gates			30	ns
$t_{d1a}(CH)$ Output disable time after $\overline{CAS}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{OFF}$	0	30	ns
$t_{d1a}(Q)$ Output disable time after $\overline{Q}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates		0	30	ns

PARAMETER	TEST CONDITIONS	ALT. SYMBOL	TMS4416-15		TMS4416-20		UNIT
			MIN	MAX	MIN	MAX	
$t_{a1}(C)$ Access time from $\overline{CAS}$	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{CAC}$		80		120	ns
$t_{a1}(R)$ Access time from $\overline{RAS}$	$t_{RLCL} = \text{MAX}$ , $C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{RAC}$		150		200	ns
$t_{a1}(Q)$ Access time after $\overline{Q}$ low	$C_L = 100$ pF, Load = 2 Series 74 TTL gates			40		50	ns
$t_{d1a}(CH)$ Output disable time after $\overline{CAS}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{OFF}$	0	30	0	40	ns
$t_{d1a}(Q)$ Output disable time after $\overline{Q}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates		0	30	0	40	ns

electrical characteristics over full ranges of recommended operating conditions (unless otherwise noted)

PARAMETER	TEST CONDITIONS	TMS4416-15			TMS4416-20			UNIT
		MIN	TYP <sup>1</sup>	MAX	MIN	TYP <sup>1</sup>	MAX	
$V_{IK}$ Input clamp voltage	$I_I = -15$ mA, see Figure 1			-1.2			-1.2	V
$V_{OH}$ High-level output voltage	$I_{QH} = -2$ mA	2.4			2.4			V
$V_{OL}$ Low-level output voltage	$I_{QL} = 4.2$ mA			0.4			0.4	V
$I_I$ Input current (leakage)	$V_I = 0$ V to 5.8 V, $V_{DD} = 5$ V, All other pins = 0 V			$\pm 10$			$\pm 10$	$\mu$ A
$I_O$ Output current (leakage)	$V_O = 0.4$ V to 5.8 V, $V_{DD} = 5$ V, $\overline{CAS}$ high			$\pm 10$			$\pm 10$	$\mu$ A
$I_{DD1}$ Average operating current during read or write cycle	At $t_0 = \text{minimum cycle}$	40	48		35	42		mA
$I_{DD2}$ Standby current	After 1 memory cycle, $\overline{RAS}$ and $\overline{CAS}$ high	3.5	5		3.5	5		mA
$I_{DD3}$ Average refresh current	$t_C = \text{minimum cycle}$ , $\overline{RAS}$ cycling, $\overline{CAS}$ high	25	40		21	34		mA
$I_{DD4}$ Average page-mode current	$t_C(P) = \text{minimum cycle}$ , $\overline{RAS}$ low, $\overline{CAS}$ cycling	25	40		21	34		mA

<sup>1</sup>All typical values are at  $T_A = 25^\circ\text{C}$  and nominal supply voltages.

<sup>2</sup> $V_{IL} \geq -0.8$  V on all inputs.

capacitance over recommended supply voltage range and operating free-air temperature range,  $f = 1$  MHz

PARAMETER	TMS4416		UNIT
	TYP <sup>1</sup>	MAX	
$C_{i(A)}$ Input capacitance, address inputs	5	7	pF
$C_{i(RC)}$ Input capacitance, strobe inputs	8	10	pF
$C_{i(W)}$ Input capacitance, write enable input	8	10	pF
$C_{i(O)}$ Input/output capacitance, data ports	8	10	pF

<sup>1</sup>All typical values are at  $T_A = 25^\circ\text{C}$  and nominal supply voltages.

Timing requirements over recommended supply voltage range and operating free-air temperature range

PARAMETER	ALT. SYMBOL	TMS4416-12		UNIT
		MIN	MAX	
t <sub>CP</sub> Page mode cycle time	IPC	120		ns
t <sub>CRd</sub> Read cycle time*	IRC	230		ns
t <sub>CRW</sub> Write cycle time	IWC	230		ns
t <sub>CRdW</sub> Read-write/read-modify-write cycle time	IRWC	320		ns
t <sub>w(CH)</sub> Pulse width, CAS high (precharge time)**	ICP	40		ns
t <sub>w(CL)</sub> Pulse width, CAS low†	ICAS	70	10,000	ns
t <sub>w(RH)</sub> Pulse width RAS high (precharge time)	IRP	80		ns
t <sub>w(RL)</sub> Pulse width, RAS low†	IRAS	120	10,000	ns
t <sub>w(W)</sub> Write pulse width	IWP	30		ns
t <sub>t</sub> Transition times (rise and fall) for RAS and CAS	IT	3	50	ns
t <sub>su(CA)</sub> Column address setup time	IASC	0		ns
t <sub>su(RA)</sub> Row address setup time	IASR	0		ns
t <sub>su(D)</sub> Data setup time	IDS	0		ns
t <sub>su(rd)</sub> Read command setup time	IRCS	0		ns
t <sub>su(WCH)</sub> Write command setup time before CAS high	ICWL	50		ns
t <sub>su(WRH)</sub> Write command setup time before RAS high	IRWL	50		ns
t <sub>h(CLCA)</sub> Column address hold time after CAS low	ICAH	35		ns
t <sub>h(RA)</sub> Row address hold time	IRAH	15		ns
t <sub>h(RLCA)</sub> Column address hold time after RAS low	IAR	85		ns
t <sub>h(CLD)</sub> Data hold time after CAS low	IDH	40		ns
t <sub>h(RLD)</sub> Data hold time after RAS low	IDHR	100		ns
t <sub>h(WLD)</sub> Data hold time after W low	IDH	30		ns
t <sub>h(RHrd)</sub> Read command hold time after RAS high	IRRH	10		ns
t <sub>h(CHrd)</sub> Read command hold time after CAS high	IRCH	0		ns
t <sub>h(CLW)</sub> Write command hold time after CAS low	IWCH	40		ns
t <sub>h(RLW)</sub> Write command hold time after RAS low	IWCR	100		ns
t <sub>RLCH</sub> Delay time, RAS low to CAS high	ICSH	150		ns
t <sub>CHRL</sub> Delay time, CAS high to RAS low	ICRP	0		ns
t <sub>CLRH</sub> Delay time, CAS low to RAS high	IRSH	80		ns
t <sub>CLWL</sub> Delay time, CAS low to W low (read, modify-write-cycle only)***	ICWD	120		ns
t <sub>RLCL</sub> Delay time, RAS low to CAS low (maximum value specified only to guarantee access time)	IRCD	20	50	ns
t <sub>RLWL</sub> Delay time, RAS low to W low (read, modify-write-cycle only)***	IRWD	170		ns
t <sub>WLCL</sub> Delay time, W low to CAS low (early write cycle)	IWCS	-5		ns
t <sub>GHQ</sub> Delay time, G high before data applied at DQ		30		ns
t <sub>f</sub> Refresh time interval	IREF		4	ms

\* Note: All cycle times assume t<sub>1</sub> = 5 ns.

\*\* Page mode only.

\*\*\* Necessary to insure  $\bar{Q}$  has disabled the output buffers prior to applying data to the device.

† In a read-modify-write cycle, t<sub>CLWL</sub> and t<sub>su(WCH)</sub> must be observed. Depending on the user's transition times, this may require additional CAS low time t<sub>w(CL)</sub>.

‡ In a read-modify-write cycle, t<sub>RLWL</sub> and t<sub>su(WRH)</sub> must be observed. Depending on the user's transition times, this may require additional RAS low time t<sub>w(RL)</sub>.

Timing requirements over recommended supply voltage range and operating free-air temperature range

PARAMETER	ALT. SYMBOL	TMS4416-15		TMS4416-20		UNIT
		MIN	MAX	MIN	MAX	
t <sub>CP</sub> Page mode cycle time	IPC	140		210		ns
t <sub>CRd</sub> Read cycle time*	IRC	280		330		ns
t <sub>CRW</sub> Write cycle time	IWC	280		330		ns
t <sub>CRdW</sub> Read-write/read-modify-write cycle time	IRWC	380		440		ns
t <sub>w(CH)</sub> Pulse width, CAS high (precharge time)**	ICP	50		80		ns
t <sub>w(CL)</sub> Pulse width, CAS low†	ICAS	80	10,000	120	10,000	ns
t <sub>w(RH)</sub> Pulse width RAS high (precharge time)	IRP	100		120		ns
t <sub>w(RL)</sub> Pulse width, RAS low†	IRAS	150	10,000	200	10,000	ns
t <sub>w(W)</sub> Write pulse width	IWP	40		50		ns
t <sub>t</sub> Transition times (rise and fall) for RAS and CAS	IT	3	50	3	50	ns
t <sub>su(CA)</sub> Column address setup time	IASC	0		0		ns
t <sub>su(RA)</sub> Row address setup time	IASR	0		0		ns
t <sub>su(D)</sub> Data setup time	IDS	0		0		ns
t <sub>su(rd)</sub> Read command setup time	IRCS	0		0		ns
t <sub>su(WCH)</sub> Write command setup time before CAS high	ICWL	80		80		ns
t <sub>su(WRH)</sub> Write command setup time before RAS high	IRWL	80		80		ns
t <sub>h(CLCA)</sub> Column address hold time after CAS low	ICAH	40		50		ns
t <sub>h(RA)</sub> Row address hold time	IRAH	20		25		ns
t <sub>h(RLCA)</sub> Column address hold time after RAS low	IAR	110		130		ns
t <sub>h(CLD)</sub> Data hold time after CAS low	IDH	80		80		ns
t <sub>h(RLD)</sub> Data hold time after RAS low	IDHR	130		160		ns
t <sub>h(WLD)</sub> Data hold time after W low	IDH	40		50		ns
t <sub>h(RHrd)</sub> Read command hold time after RAS high	IRRH	10		10		ns
t <sub>h(CHrd)</sub> Read command hold time after CAS high	IRCH	0		0		ns
t <sub>h(CLW)</sub> Write command hold time after CAS low	IWCH	60		80		ns
t <sub>h(RLW)</sub> Write command hold time after RAS low	IWCR	130		160		ns
t <sub>RLCH</sub> Delay time, RAS low to CAS high	ICSH	150		200		ns
t <sub>CHRL</sub> Delay time, CAS high to RAS low	ICRP	0		0		ns
t <sub>CLRH</sub> Delay time, CAS low to RAS high	IRSH	80		120		ns
t <sub>CLWL</sub> Delay time, CAS low to W low (read, modify-write-cycle only)***	ICWD	120		150		ns
t <sub>RLCL</sub> Delay time, RAS low to CAS low (maximum value specified only to guarantee access time)	IRCD	20	70	25	80	ns
t <sub>RLWL</sub> Delay time, RAS low to W low (read, modify-write-cycle only)***	IRWD	190		230		ns
t <sub>WLCL</sub> Delay time, W low to CAS low (early write cycle)	IWCS	-5		-5		ns
t <sub>GHQ</sub> Delay time, G high before data applied at DQ		30		40		ns
t <sub>f</sub> Refresh time interval	IREF		4		4	ms

\* Note: All cycle times assume t<sub>1</sub> = 5 ns.

\*\* Page mode only.

\*\*\* Necessary to insure  $\bar{Q}$  has disabled the output buffers prior to applying data to the device.

† In a read-modify-write cycle, t<sub>CLWL</sub> and t<sub>su(WCH)</sub> must be observed. Depending on the user's transition times, this may require additional CAS low time t<sub>w(CL)</sub>.

‡ In a read-modify-write cycle, t<sub>RLWL</sub> and t<sub>su(WRH)</sub> must be observed. Depending on the user's transition times, this may require additional RAS low time t<sub>w(RL)</sub>.

recommended operating conditions

PARAMETER	SMJ4418						UNIT
	S VERSION			E VERSION			
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, $V_{DD}$	4.5	5	5.5	4.5	5	5.5	V
Supply voltage, $V_{SS}$	0						V
High-level input voltage, $V_{IH}$	$V_{DD} = 4.5$ V		2.4	4.8		4.8	V
	$V_{DD} = 5.5$ V		2.4	5.8		5.8	
Low-level input voltage, $V_{IL}$ (see Note 2)			$V_{IK}$	0.8		$V_{IK}$	V
Operating case temperature, $T_C$	-55		100		-40		85 °C

NOTE 2: The algebraic convention, where the more negative (less positive) limit is designated as minimum, is used in this data sheet for logic voltage levels only.

electrical characteristics over full range of recommended operating conditions (unless otherwise noted)

PARAMETER	TEST CONDITIONS	SMJ4418-12			UNIT
		MIN	TYP <sup>1</sup>	MAX	
$V_{IK}$ Input clamp voltage	$I_I = -15$ mA, see Figure 1	-1.2			V
$V_{OH}$ High-level output voltage	$I_{OH} = -2$ mA	2.4			V
$V_{OL}$ Low-level output voltage	$I_{OL} = 4.2$ mA	0.4			V
$I_I$ Input current (leakage)	$V_I = 0$ V to 5.5 V, $V_{DD} = 5$ V, All other pins = 0 V	$\pm 10$			$\mu$ A
$I_O$ Output current (leakage)	$V_O = 0.4$ V to 5.5 V, $V_{DD} = 5$ V, $\overline{CAS}$ high	$\pm 10$			$\mu$ A
$I_{DD1}$ Average operating current during read or write cycle	At $t_C =$ minimum cycle	54			mA
$I_{DD2}$ <sup>2</sup> Standby current	After 1 memory cycle, $\overline{RAS}$ and $\overline{CAS}$ high	3.5 5			mA
$I_{DD3}$ Average refresh current	$t_C =$ minimum cycle, $\overline{RAS}$ cycling, $\overline{CAS}$ high	46			mA
$I_{DD4}$ Average page-mode current	$t_{C(P)} =$ minimum cycle, $\overline{RAS}$ low, $\overline{CAS}$ cycling	46			mA

<sup>1</sup>All typical values are at  $T_C = 25^\circ\text{C}$  and nominal supply voltages.

<sup>2</sup> $V_{IK} = -0.8$  V on all inputs.

electrical characteristics over full range of recommended operating conditions (unless otherwise noted)

PARAMETER	TEST CONDITIONS	SMJ4418-18			SMJ4418-20			UNIT
		MIN	TYP <sup>1</sup>	MAX	MIN	TYP <sup>1</sup>	MAX	
$V_{IK}$ Input clamp voltage	$I_I = -15$ mA, see Figure 1	-1.2			-1.2			V
$V_{OH}$ High-level output voltage	$I_{OH} = -2$ mA	2.4			2.4			V
$V_{OL}$ Low-level output voltage	$I_{OL} = 4.2$ mA	0.4			0.4			V
$I_I$ Input current (leakage)	$V_I = 0$ V to 5.5 V, $V_{DD} = 5$ V, All other pins = 0 V	$\pm 10$			$\pm 10$			$\mu$ A
$I_O$ Output current (leakage)	$V_O = 0.4$ V to 5.5 V, $V_{DD} = 5$ V, $\overline{CAS}$ high	$\pm 10$			$\pm 10$			$\mu$ A
$I_{DD1}$ Average operating current during read or write cycle	At $t_C =$ minimum cycle	40 48		38 42		mA		
$I_{DD2}$ <sup>2</sup> Standby current	After 1 memory cycle, $\overline{RAS}$ and $\overline{CAS}$ high	3.8 5		3.5 5		mA		
$I_{DD3}$ Average refresh current	$t_C =$ minimum cycle, $\overline{RAS}$ cycling, $\overline{CAS}$ high	28 40		21 34		mA		
$I_{DD4}$ Average page-mode current	$t_{C(P)} =$ minimum cycle, $\overline{RAS}$ low, $\overline{CAS}$ cycling	25 40		21 34		mA		

<sup>1</sup>All typical values are at  $T_C = 25^\circ\text{C}$  and nominal supply voltages.

<sup>2</sup> $V_{IK} = -0.8$  V on all inputs.

capacitance over recommended supply voltage range and operating case temperature range,  $f = 1$  MHz

PARAMETER	SMJ4418		UNIT
	TYP <sup>1</sup>	MAX	
$C_{I(A)}$ Input capacitance, address inputs	5	7	pF
$C_{I(RC)}$ Input capacitance, strobe inputs	8	10	pF
$C_{I(W)}$ Input capacitance, write enable input	8	10	pF
$C_{I(O)}$ Input/output capacitance, data ports	8	10	pF

<sup>1</sup>All typical values are at  $T_C = 25^\circ\text{C}$  and nominal supply voltages.

switching characteristics over recommended supply voltage range and operating case temperature range

PARAMETER	TEST CONDITIONS	ALT. SYMBOL	SMJ4416-12		UNIT
			MIN	MAX	
$t_{a(C)}$ Access time from $\overline{CAS}$	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{CAC}$	70		ns
$t_{a(R)}$ Access time from $\overline{RAS}$	$t_{RLCL} = MAX$ , $C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{RAC}$	120		ns
$t_{a(G)}$ Access time after $\overline{G}$ low	$C_L = 100$ pF, Load = 2 Series 74 TTL gates		30		ns
$t_{dlat(CH)}$ Output disable time after $\overline{CAS}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{OFF}$	0	30	ns
$t_{dlat(G)}$ Output disable time after $\overline{G}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates		0	30	ns

PARAMETER	TEST CONDITIONS	ALT. SYMBOL	SMJ4416-15		SMJ4416-20		UNIT
			MIN	MAX	MIN	MAX	
$t_{a(C)}$ Access time from $\overline{CAS}$	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{CAC}$	80		120		ns
$t_{a(R)}$ Access time from $\overline{RAS}$	$t_{RLCL} = MAX$ , $C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{RAC}$	150		200		ns
$t_{a(G)}$ Access time after $\overline{G}$ low	$C_L = 100$ pF, Load = 2 Series 74 TTL gates		40		50		ns
$t_{dlat(CH)}$ Output disable time after $\overline{CAS}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates	$t_{OFF}$	0	30	0	40	ns
$t_{dlat(G)}$ Output disable time after $\overline{G}$ high	$C_L = 100$ pF, Load = 2 Series 74 TTL gates		0	30	0	40	ns

timing requirements over recommended supply voltage range and operating case temperature range

PARAMETER	ALT. SYMBOL	SMJ4416-12		UNIT
		MIN	MAX	
$t_{c(P)}$ Page mode cycle time	$t_{PC}$	120		ns
$t_{c(RD)}$ Read cycle time*	$t_{RC}$	230		ns
$t_{c(WR)}$ Write cycle time	$t_{WC}$	230		ns
$t_{c(RDWR)}$ Read-write/read-modify-write cycle time	$t_{RWC}$	320		ns
$t_{w(CH)}$ Pulse width, $\overline{CAS}$ high (precharge time)**	$t_{CP}$	40		ns
$t_{w(CL)}$ Pulse width, $\overline{CAS}$ low†	$t_{CAS}$	70	10,000	ns
$t_{w(RH)}$ Pulse width, $\overline{RAS}$ high (precharge time)	$t_{RP}$	80		ns
$t_{w(RL)}$ Pulse width, $\overline{RAS}$ low†	$t_{RAS}$	120	10,000	ns
$t_{w(W)}$ Write pulse width	$t_{WP}$	30		ns
$t_t$ Transition times (rise and fall) for RAS and CAS	$t_T$	3	50	ns
$t_{su(CA)}$ Column address setup time	$t_{ASC}$	0		ns
$t_{su(RA)}$ Row address setup time	$t_{ASR}$	0		ns
$t_{su(D)}$ Data setup time	$t_{DS}$	0		ns
$t_{su(Rd)}$ Read command setup time	$t_{RCS}$	0		ns
$t_{su(WC)}$ Write command setup time before $\overline{CAS}$ high	$t_{CWL}$	50		ns
$t_{su(WRH)}$ Write command setup time before $\overline{RAS}$ high	$t_{RWL}$	50		ns
$t_{h(CLCA)}$ Column address hold time after $\overline{CAS}$ low	$t_{CAH}$	35		ns
$t_{h(RA)}$ Row address hold time	$t_{RAH}$	15		ns
$t_{h(RLCA)}$ Column address hold time after $\overline{RAS}$ low	$t_{RAH}$	85		ns
$t_{h(CL)}$ Data hold time after $\overline{CAS}$ low	$t_{DH}$	40		ns
$t_{h(RL)}$ Data hold time after $\overline{RAS}$ low	$t_{DR}$	100		ns
$t_{h(WL)}$ Data hold time after $\overline{W}$ low	$t_{DH}$	30		ns
$t_{h(RHrd)}$ Read command hold time after $\overline{RAS}$ high	$t_{RRH}$	10		ns
$t_{h(CHrd)}$ Read command hold time after $\overline{CAS}$ high	$t_{RCH}$	0		ns
$t_{h(CLW)}$ Write command hold time after $\overline{CAS}$ low	$t_{WCH}$	40		ns
$t_{h(RLW)}$ Write command hold time after $\overline{RAS}$ low	$t_{WCR}$	100		ns
$t_{RLCH}$ Delay time, $\overline{RAS}$ high to $\overline{CAS}$ high	$t_{CSH}$	150		ns
$t_{CHRL}$ Delay time, $\overline{CAS}$ high to $\overline{RAS}$ low	$t_{CRP}$	0		ns
$t_{CLRH}$ Delay time, $\overline{CAS}$ low to $\overline{RAS}$ high	$t_{RSH}$	80		ns
$t_{CLWL}$ Delay time, $\overline{CAS}$ low to $\overline{W}$ low (read, modify-write-cycle only)***	$t_{CWD}$	120		ns
$t_{RLCL}$ Delay time, $\overline{RAS}$ low to $\overline{CAS}$ low (maximum value specified only to guarantee access time)	$t_{RCD}$	20	50	ns
$t_{RLWL}$ Delay time, $\overline{RAS}$ low to $\overline{W}$ low (read, modify-write-cycle only)***	$t_{RWO}$	170		ns
$t_{WLCL}$ Delay time, $\overline{W}$ low to $\overline{CAS}$ low (early write cycle)	$t_{WCS}$	-5		ns
$t_{GHQ}$ Delay time, $\overline{G}$ high before data applied at DQ		30		ns
$t_{rf}$ Refresh time interval	$t_{REF}$		4	ms

\* Note: All cycle times assume  $t_1 = 5$  ns.

\*\* Page mode only.

\*\*\* Necessary to insure  $\overline{G}$  has disabled the output buffers prior to applying data to the device.

† In a read-modify-write cycle,  $t_{w(CL)}$  and  $t_{su(WC)}$  must be observed. Depending on the user's transition times, this may require additional  $\overline{CAS}$  low time  $t_{w(CL)}$ .

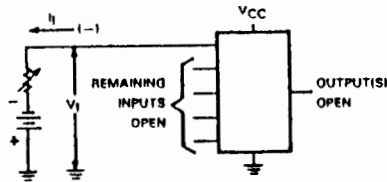
‡ In a read-modify-write cycle,  $t_{w(RL)}$  and  $t_{su(WRH)}$  must be observed. Depending on the user's transition times, this may require additional  $\overline{RAS}$  low time  $t_{w(RL)}$ .

timing requirements over recommended supply voltage range and operating case temperature range

PARAMETER	ALT. SYMBOL	SMJ4418-15		SMJ4418-20		UNIT	
		MIN	MAX	MIN	MAX		
t <sub>CP</sub>	Page mode cycle time	t <sub>PC</sub>	140	210		ns	
t <sub>CRd</sub>	Read cycle time	t <sub>RC</sub>	260	330		ns	
t <sub>CRW</sub>	Write cycle time	t <sub>WC</sub>	260	330		ns	
t <sub>CRdW</sub>	Read-write/read-modify-write cycle time	t <sub>RWC</sub>	360	440		ns	
t <sub>w(CH)</sub>	Pulse width, CAS high (precharge time) **	t <sub>CP</sub>	50	80		ns	
t <sub>w(CL)</sub>	Pulse width, CAS low †	t <sub>CAS</sub>	80	10,000	120	10,000	ns
t <sub>w(RH)</sub>	Pulse width RAS high (precharge time)	t <sub>RP</sub>	100	120		ns	
t <sub>w(RL)</sub>	Pulse width, RAS low ‡	t <sub>RAS</sub>	150	10,000	200	10,000	ns
t <sub>w(W)</sub>	Write pulse width	t <sub>WP</sub>	40	50		ns	
t <sub>t</sub>	Transition times (rise and fall) for RAS and CAS	t <sub>T</sub>	3	50	3	50	ns
t <sub>su(CA)</sub>	Column address setup time	t <sub>ASC</sub>	0	0		ns	
t <sub>su(RA)</sub>	Row address setup time	t <sub>ASR</sub>	0	0		ns	
t <sub>su(D)</sub>	Data setup time	t <sub>DS</sub>	0	0		ns	
t <sub>su(rd)</sub>	Read command setup time	t <sub>RCS</sub>	0	0		ns	
t <sub>su(WCH)</sub>	Write command setup time before CAS high	t <sub>CWL</sub>	60	80		ns	
t <sub>su(WRH)</sub>	Write command setup time before RAS high	t <sub>RWL</sub>	80	80		ns	
t <sub>h(CLA)</sub>	Column address hold time after CAS low	t <sub>CAH</sub>	40	50		ns	
t <sub>h(RA)</sub>	Row address hold time	t <sub>RAH</sub>	20	25		ns	
t <sub>h(RLCA)</sub>	Column address hold time after RAS low	t <sub>AR</sub>	110	130		ns	
t <sub>h(CLD)</sub>	Data hold time after CAS low	t <sub>DH</sub>	80	80		ns	
t <sub>h(RLD)</sub>	Data hold time after RAS low	t <sub>DHR</sub>	130	160		ns	
t <sub>h(WLD)</sub>	Data hold time after W low	t <sub>DH</sub>	40	50		ns	
t <sub>h(RHrd)</sub>	Read command hold time after RAS high	t <sub>RRH</sub>	10	10		ns	
t <sub>h(CHrd)</sub>	Read command hold time after CAS high	t <sub>RCH</sub>	0	0		ns	
t <sub>h(CLW)</sub>	Write command hold time after CAS low	t <sub>WCH</sub>	60	80		ns	
t <sub>h(RLW)</sub>	Write command hold time after RAS low	t <sub>WCR</sub>	130	160		ns	
t <sub>RLCH</sub>	Delay time, RAS low to CAS high	t <sub>CSH</sub>	150	200		ns	
t <sub>CHRL</sub>	Delay time, CAS high to RAS low	t <sub>CRP</sub>	0	0		ns	
t <sub>CLRH</sub>	Delay time, CAS low to RAS high	t <sub>RSR</sub>	80	120		ns	
t <sub>CLWL</sub>	Delay time, CAS low to W low (read, modify-write-cycle only) ***	t <sub>CWD</sub>	120	150		ns	
t <sub>RLCL</sub>	Delay time, RAS low to CAS low (maximum value specified only to guarantee access time)	t <sub>RCD</sub>	20	70	25	80	ns
t <sub>RLWL</sub>	Delay time, RAS low to W low (read, modify-write-cycle only) ***	t <sub>RWD</sub>	190	230		ns	
t <sub>WLCL</sub>	Delay time, W low to CAS low (early write cycle)	t <sub>WCS</sub>	-5	-5		ns	
t <sub>GHD</sub>	Delay time, G high before data applied at DQ		30	40		ns	
t <sub>REF</sub>	Refresh time interval	t <sub>REF</sub>		4		4	ms

- \* Note: All cycle times assume t<sub>r</sub> = 5 ns.
- \*\* Page mode only.
- \*\*\* Necessary to insure  $\bar{C}$  has disabled the output buffers prior to applying data to the device.
- † In a read-modify-write cycle, t<sub>CLWL</sub> and t<sub>su(WCH)</sub> must be observed. Depending on the user's transition times, this may require additional CAS low time t<sub>w(CL)</sub>.
- ‡ In a read-modify-write cycle, t<sub>RLWL</sub> and t<sub>su(WRH)</sub> must be observed. Depending on the user's transition times, this may require additional RAS low time t<sub>w(RL)</sub>.

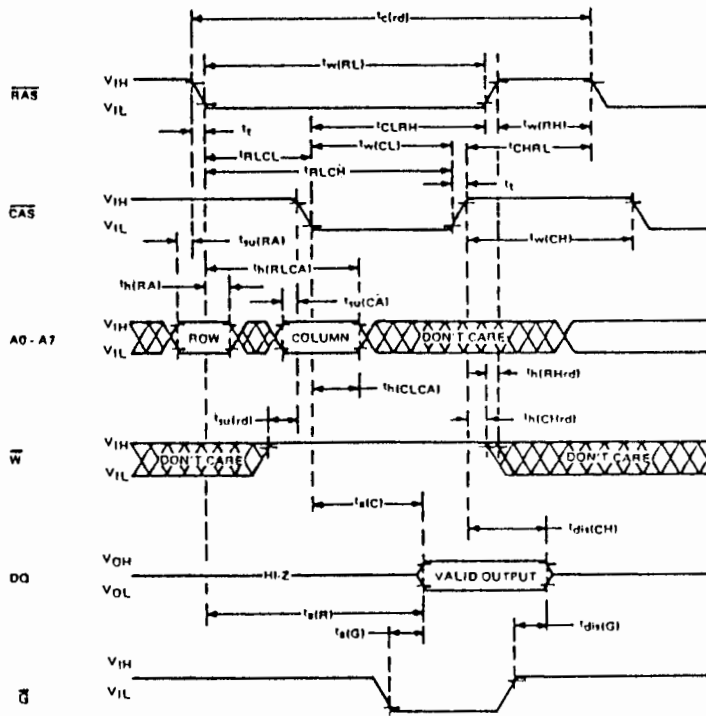
PARAMETER MEASUREMENT INFORMATION



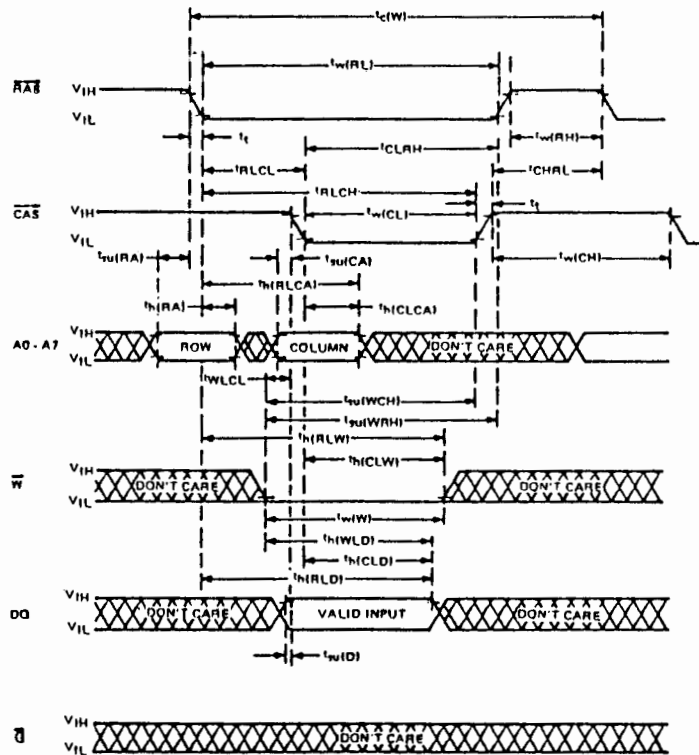
NOTE: Each input is tested separately.

FIGURE 1 - INPUT CLAMP VOLTAGE TEST CIRCUIT

read cycle timing

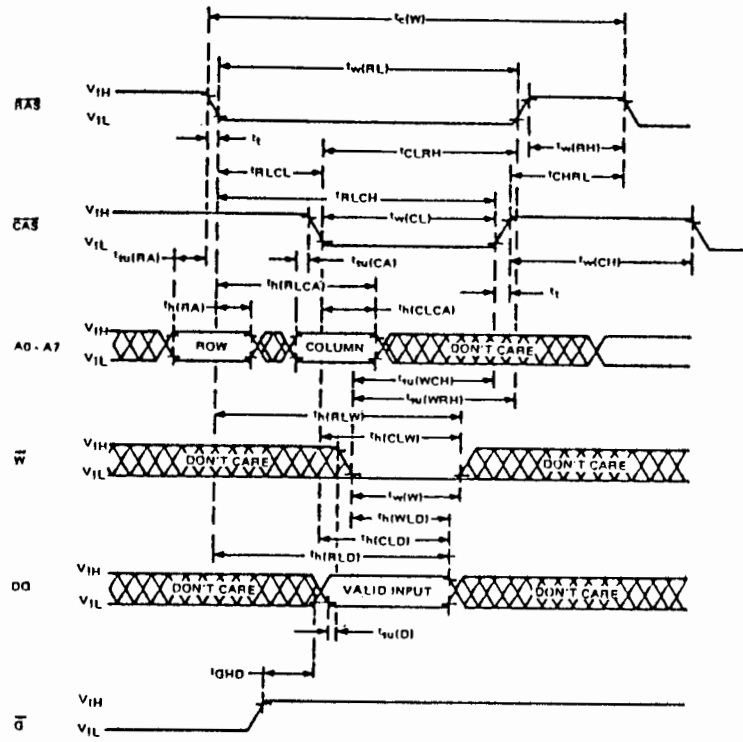


early write cycle timing

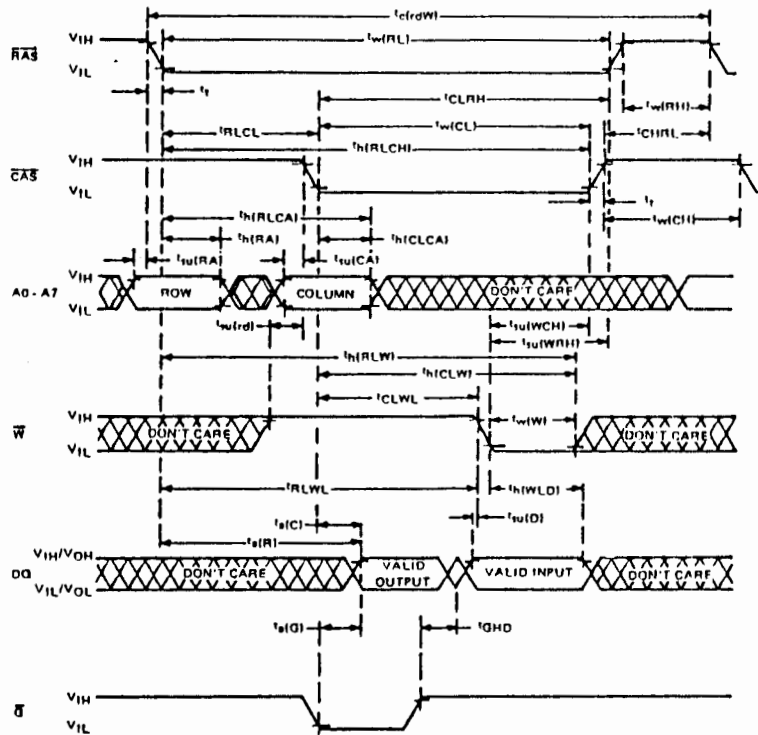




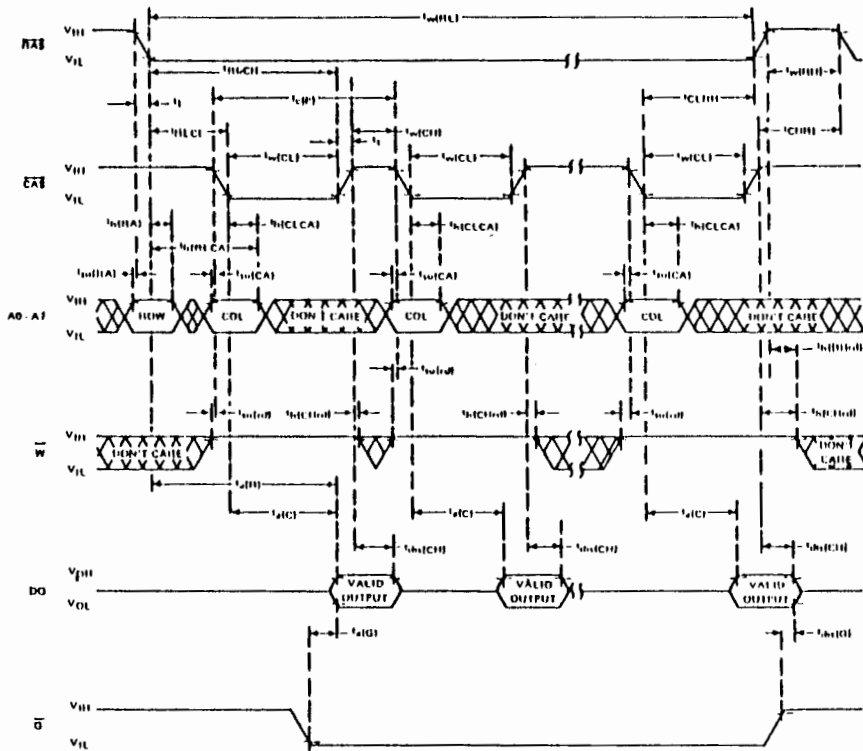
write cycle timing



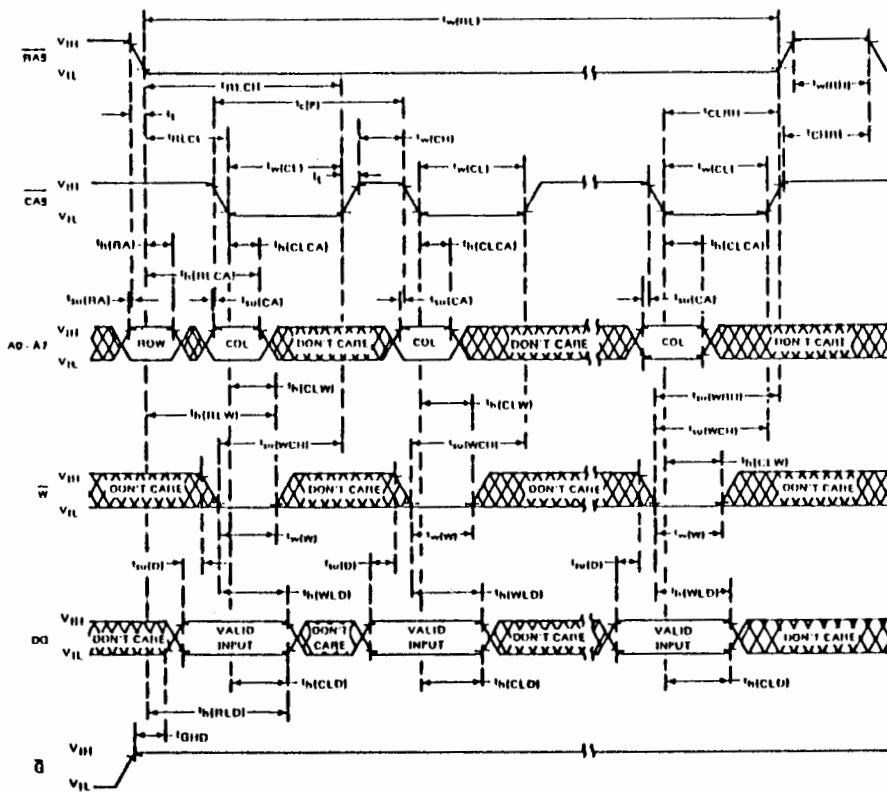
read-write/read-modify-write cycle timing



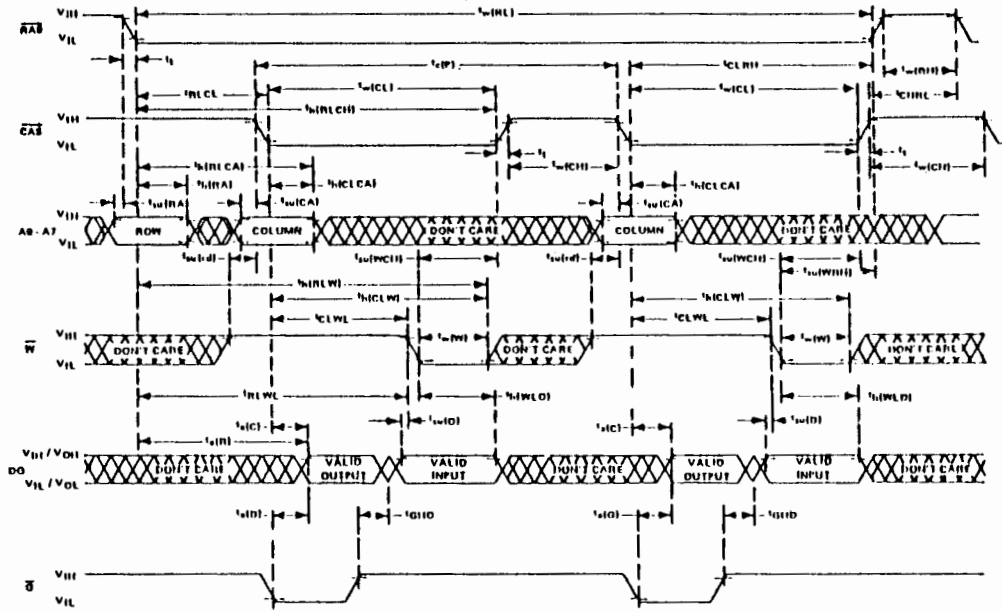
page-mode read cycle timing



page-mode write cycle timing

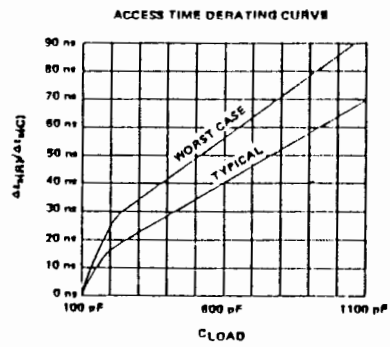
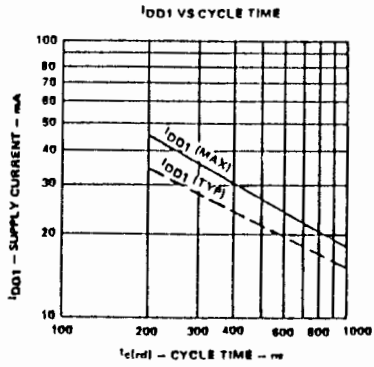
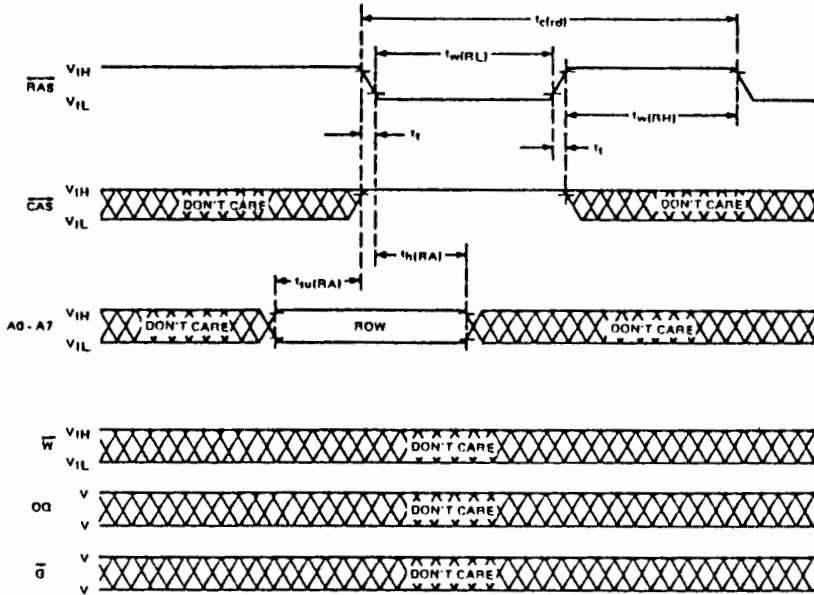


page mode read-modify-write timing



NOTE: A write cycle of read-modify-write cycle can be intermixed with read cycles as long as the write and read-modify-write timing specifications are violated.

**RAS-only refresh timing**



APPENDIX E

SOFTWARE TESTING PROGRAMMES

```

*****
SV-728 PERSONNEL COMPUTER TESTING PROGRAM
*****

TESTING 1: USER RAM TEST
-----
1. WRITE TEST PATTERN SET TO RAM
2. LEAKAGE PROCESS
3. READ AND CHECK THE RAM CONTENTS

TESTING 2: VRAM TEST
-----
1. VDP INITIALIZATION
2. WRITE TEST PATTERN SET INTO 16K VRAM
3. READ TEST PATTERN FROM VRAM FOR CHECKING

TESTING 3: PSG TEST
-----
1. WRITE DATA TO ALL PSG REGISTER
2. READ AND CHECK ALL DATA READ FROM PSG
3. TEST THREE SOUND CHANNELS

TESTING 4: SOUND TEST
-----

TESTING 5: VDP INTERRUPT TEST
-----

TESTING 6: KEYBOARD TEST
-----
CONTACT -> TONE GIVEN
NOT CONTACT -> NO TONE

TESTING 7: JOYSTICKS TEST
-----

TESTING 8: PRINTER TEST
-----

TESTING 8: CASSETTE TEST
-----

LINKING:

LBO TEST728/W
#P:0000/D:0000
#MAIN,ROM,RVMSUB,SUB,KEYBRD
#JOYSTICK,PRINT,GRAPHIC,DWASET/E

VER. 1 : DEC. 23, 1983
VER. 2 : 1st, July, 1984, for MSX sv7.8
VER. 3 : AUG. 15, 1984 FOR MSX SV728
*****

.ZDD
TITLE SVTEST.MMC

ENTRY POST1,POST2,RVMSUB,VDFREQ,VFMT,VFCT,VFGT
ENTRY PSG_TEST,FUN,OPTION,CAS_TEST,ROM,ROM_TT

EXT PRINTG,PRINT,DISPLAY,RMSET,WMSET,AMPA,AMPB,AMPC
EXT OFFD1A,OFFD1B,OFFD1C,TONEA,TONEB,TONEC,PSG1,PSG2
EXT PSGR1,PSGR2,CVRLK,SVHLK,WHHLK,RVMBYT,WMBYT,WDFRG
EXT VDFSET,JOYRDD,FIRE,DELAY,BXASH,SOUND,RVDFRG,SOUND
EXT NOISE,KBJOYS
EXT VRAM_TEST,SRAM_TEST,KEY_TEST,JOY_TEST,PRINT_TEST
EXT ASCII,DHSG,DELAY1

```

```

----- INTERRUPT VECTOR -----
0038 INTVECT EQU 0038H ; INTERRUPT VECTOR

----- VDP INIT. PARAMETERS -----
0002 VR0 EQU 021 ; 0,0,0,0,0,0,MS,EV
0000 VR1 EQU 000H ; 4/16K,16/16K,1E,M1,M2,0,SIZE,MFG
000E VR2 EQU 0E3H ; 0,0,0,0,0,MMVE TABLE BASE ADDRESS
00FF VR3 EQU 0FFH ; COLOR TABLE BASE ADDRESS
0003 VR4 EQU 031H ; 0,0,0,0,0,PATTERN GEN. BASE ADDR.
0076 VR5 EQU 761H ; 0,SPRITE ATTRIBUTE TABLE B. ADDR.
0003 VR6 EQU 031H ; 0,0,0,0,0,SPRITE PAT GEN B. ADDR.
0000 VR7 EQU 00H ; TEXT COL. 1, TEXT COL. 0/BACKDROP

----- VRAM MAP -----
0000 VFGT EQU (VR4 AND 04H)+800H
1800 VSGT EQU VR6+000H
2000 VFCT EQU (VR3 AND 04H)+40H
3800 VFMT EQU VR2+400H
3B00 VSAT EQU VR5+80H

----- VDP INPUT/OUTPUT PORT ASSIGNMENT -----
0098 MVDATA EQU 98H ; DATA PORT, WRITE TO VDP
0098 RVDATA EQU 98H ; DATA PORT, READ FROM VDP
009F SVDFRB EQU 9FH ; VDP REGISTER

009F SVADDR EQU 9FH ; ADDRESS LATCH
009F RVSTAT EQU 9FH ; READ VDP STATUS

1800 SSTRFR EQU 1800H ; SPRITE GENERATOR TABLE POINTER
3B00 SATPTR EQU 3B00H ; SPRITE ATTRIBUTE TABLE POINTER

----- PSG PARAMETER -----
00A0 PSELAT EQU 0A0H ; PORT#, PSG LATCH ADDRESS
00A2 RPSB EQU 0A2H ; READ FROM PSG DATA PORT
00A1 WPSB EQU 0A1H ; WRITE TO PSG PORT
00CE PSEPTA EQU 0E1H ; PSG REGISTER 14
00CF PSEPTB EQU 0FH ; PSG REGISTER 15

----- FPI I/O PORT ASSIGNMENT -----
00AA WFPIC EQU 0Aah ; WRITE INTO FPI PORT C
00A9 RFP1B EQU 0A9h ; READ FROM FPI PORT C
00AB RFP1A EQU 0Abh ; READ FROM FPI PORT A
00A8 WFP1A EQU 0A8h ;
00B2 FPICMD EQU 0B2H ; FPI COMMAND WORD
00AB FPICMP EQU 0Abh ; FPI COMMAND PORT
00AA FPICR EQU 0Aah ;

0000' 41 db 41h ;for msx game cart. indicator
0001' 42 db 42h ;
0002' 0011' dw START ; start addr. vector
0004' 00 db 00h ;

0005' 00 00 00 00 db 0,0,0,0,0,0,0,0,0,0,0
0009' 00 00 00 00
000D' 00 00 00 00

0011' F3 START: DI ; INSTRUCTION DI AND LOAD STACK ->
0012' C3 0015' JP START ; JUMP TO MAIN PROGRAM

ORG 0030H ; AREA FOR VDP INT

JP VDFINT ; JUMP TO VDP INTERRUPT ROUTINE
ORG 0100H ; MAIN PROGRAM START HERE

----- HARDWARE INITIALIZATION -----
0015' C3 0018' START1: JP START2
0018' ED 56 START2: IM 1 ; SET INTERRUPT MODE 1

001A' 21 05E6' LD HL,VDFREB
001D' CD 0000* CALL VDFSET

```

```

    ROM & VRAM TEST OK, PROVIDE OPTION SELECT
    -----
0020'          OPTION#
    |
    | << CLEAR VRAM AND SET GENERATOR TABLE >>
    |
0020' 01 4000      LD BC,4000H      | CLEAR 16K VRAM
0023' 11 0000      LD DE,VPBT
0026' CD 0000#     CALL CVMBLK
    |
0029' 21 0000#     LD HL,ASC11      | SET ASCII PATTERNS
002C' 11 0100      LD DE,VPBT+20H#B
002F' 01 0300      LD BC,300H
0032' CD 0000#     CALL WMBLK
    |
0035' 21 0000#     LD HL,ASC11      | SET ASCII PATTERNS
0038' 11 0900      LD DE,VPBT+204B+20H#B
003B' 01 0300      LD BC,300H
003E' CD 0000#     CALL WMBLK
    |
0041' 21 0000#     LD HL,ASC11      | SET ASCII PATTERNS
0044' 11 1100      LD DE,VPBT+204B+2+20H#B
0047' 01 0300      LD BC,300H
004A' CD 0000#     CALL WMBLK
    |
004D' 3E F1        LD A,0FH          | TEXT COLOR
004F' 11 2000      LD DE,VPCT
0052' 01 1800      LD BC,256+B*3
0055' CD 0000#     CALL SMBLK
    |
    | << DISPLAY MESSAGE >>
    |
0058' 21 01CA'     LD HL,TEXT1          | 1) VIDEO RAM TEST
005B' 11 3B47      LD DE,VPNT+2*32+7
005E' DD 21 0065'  LD IX,#+7
0062' C3 0000#     JP DHEB
    |
0065' 21 01DE'     LD HL,TEXT2          | 2) SYSTEM RAM TEST
0068' 11 3B87      LD DE,VPNT+4*32+7
006B' DD 21 0072'  LD IX,#+7
006F' C3 0000#     JP DHEB
    |
0072' 21 01F1'     LD HL,K00K
0075' 11 3BC7      LD DE,VPNT+6*32+7
0078' DD 21 007E'  LD IX,#+7
007C' C3 0000#     JP DHEB
    |
007F' 21 0205'     LD HL,TEXT3          | 3) PSG AND SOUND TEST
0082' 11 3707      LD DE,VPNT+B*32+7
0085' DD 21 008C'  LD IX,#+7
0089' C3 0000#     JP DHEB
    |
008C' 21 021C'     LD HL,TEXT4          | 4) VDP INTERRUPT TEST
008F' 11 3747      LD DE,VPNT+0FH*32+7
0092' DD 21 0099'  LD IX,#+7
0096' C3 0000#     JP DHEB
    |
0099' 21 0233'     LD HL,TEXT5          | 5) KEYBOARD TEST
009C' 11 3787      LD DE,VPNT+0CH*32+7
009F' DD 21 00A6'  LD IX,#+7
00A3' C3 0000#     JP DHEB
    |
00A6' 21 0245'     LD HL,TEXT6          | 6) JOYSTICKS TEST
00A9' 11 37C7      LD DE,VPNT+0EH*32+7
00AC' DD 21 00B3'  LD IX,#+7
00B0' C3 0000#     JP DHEB
    |
00B3' 21 0257'     LD HL,TEXT7          | 7) CASSETTE TEST
00B6' 11 3707      LD DE,VPNT+10H*32+7
00B9' DD 21 00C0'  LD IX,#+7
00BD' C3 0000#     JP DHEB
    |
00C0' 21 0268'     LD HL,TEXT8          | 8) PRINTER TEST
00C3' 11 3747      LD DE,VPNT+12H*32+7
00C6' DD 21 00CD'  LD IX,#+7
00CA' C3 0000#     JP DHEB
    |
00CD' 21 027A'     LD HL,TEXT9          | 9) ALL TEST
00D0' 11 3787      LD DE,VPNT+14H*32+7
00D3' DD 21 00DA'  LD IX,#+7
00D7' C3 0000#     JP DHEB
    |
00DA' 21 0287'     LD HL,TEXT10         | FUNCTION SELECT
00DD' 11 37C3      LD DE,VPNT+16H*32+3
00E0' DD 21 00E7'  LD IX,#+7
00E4' C3 0000#     JP DHEB
    |
    | << WAIT FOR REPLY >>
    |
    | KEY_IN#
    | IN A,(RFF10R) | UNAFFECT OTHER BITS
    | AND 0FH
    | LD C,A
    | LD E,0FH
    |
    | KEY_IN#1
    | LD A,C
    | OUT (WP10),A | SELECT LINE 0
    | IN A,(RFF10B)
    | CP 0FH        | KEY PRESS ?
    | JR NZ,FUN_SEL1
    | LD A,C
    | INC A
    | OUT (WP10),A | SELECT LINE 1
    | IN A,(RFF10B)
    | CP 0FH1      | KEY 8 PRESS ?
    | JR Z,FUN_SEL
    | CP 0FH1      | KEY 9 PRESS
    | JR NZ,KEY_IN1
    |
    | << GET FUNCTION SELECT NO. IN HEX >>
    |
    | FUN_SEL#
    | LD E,07H
    |
    | FUN_SEL#1
    | INC E          | SCAN LINE
    | RRA
    | JR C,FUN_SEL1
    | LD A,E
    | LD (FUN),A
    | AND A,A
    | LD E,A
    | LD D,00H
    | LD IX,FUN_TBL | GET JUMP TABLE HEAD
    | AND IX,DE
    | LD L,(IX)
    | LD H,(IX+1)
    | JP (HL)
    |
    | ROM#
    | CALL EE_E
    | LD HL,VRAM
    | LD DE,VPNT+6*32+7
    | LD IX,#+7
    | JP DHEB
    | IN A,(RFF10A)
    | LD (STORE),A
    | AND 11001111B
    | OR 00100000B
    | OUT (WP10A),A
    | JP 0000H
    |
    | ROM_FT#
    | LD DE,SUM2
    | PUSH IX
    | POP HL
    | OR A
    | SBC HL,DE
    | JP NZ,FNT_ERR
    | LD DE,SUM1
    | PUSH IX
    | POP HL
    | OR A

```

0151' ED 52  
0153' C2 0179'  
0156' 21 0189'  
0159' 11 3907  
015C' DD 21 0163'  
0160' C3 0000\*  
0163' CD 0000\*  
0166' CD 0000\*  
0169' 3A 0007"  
016C' D3 00  
016E' 3A 000E\*  
0171' FE 09  
0173' C2 0020'  
  
0176' C3 0000\*  
  
0179'   
0179' 21 01A5'  
017C' 11 3907  
017F' DD 21 01B6'  
0183' C3 0000\*  
0186' C3 0163'  
  
0189' 52 4F 4D 20  
01E0' 5A 45 53 54  
0191' 20 4F 4B 00  
0195' 53 59 53 54  
0199' 45 4D 20 52  
019D' 4F 4D 20 54  
01A1' 45 53 54 00  
01A5' 52 4F 4D 20  
01A9' 5A 45 53 54  
01AD' 20 46 41 49  
01B1' 4C 00  
  
003B  
04B0  
  
0183'  
0183' C3 0120'  
  
01B6'  
01B6' 0120'  
01B9' 0000\*  
01BA' 0000\*  
01BC' 02A2'  
01BE' 03F8'  
01C0' 0000\*  
01C2' 0000\*  
01CA' 0000\*  
01CB' 053E'  
01CB' 01B3'  
  
01CA' 30 29 20 20  
01CE' 53 59 53 54  
01D2' 45 4D 20 52  
01D6' 4F 4D 20 54  
01DA' 45 53 54 00  
01DE' 31 29 20 20  
01E2' 56 49 44 45  
01E6' 4F 20 52 41  
01EA' 4D 20 54 45  
01EE' 53 54 00  
01F1' 32 29 20 20  
01F5' 53 59 53 54  
01F9' 45 4D 20 52  
01FD' 41 4D 20 54  
0201' 45 53 54 00  
0205' 33 29 20 20  
0209' 50 53 47 20  
020D' 41 4E 44 20  
0211' 53 4F 53 4E  
0215' 44 20 54 45

SBC HL,DE  
JP NZ,FMT\_ERR  
LD HL,FFFF  
LD DE,VFNT+0\*32+7  
LD IX,4+7  
JP DNEG\*  
XXXX:  
CALL DELAY1  
CALL DELAY1  
LD A,(STORE)  
OUT (WPP1A),A  
LD A,(FLND)  
CP 09H  
JP NZ,OPT10N  
  
JP WRAM\_TEST  
}  
FMT\_ERR:  
LD HL,ERR\_MESS  
LD DE,VFNT+0\*32+7  
LD IX,4+7  
JP DNEG  
JP XXXX  
}  
FFFF: DB 'ROM TEST OK',00H  
AAAA: DB 'SYSTEM ROM TEST',00H  
ERR\_MESS: DB 'ROM TEST FAIL',00H  
SUM1 EQU 003BH .  
SUM2 EQU 04B0H .  
}  
ALL\_TEST:  
JP ROM  
}  
}  
{<< FUNCTION JUMP TABLE >>  
}  
FUN\_TBL:  
DW ROM .  
DW WRAM\_TEST  
DW SRAM\_TEST  
DW FSB\_TEST  
DW INT\_TEST  
DW KEY\_TEST  
DW JOY\_TEST  
DW FRONT\_TEST  
DW CAS\_TEST  
DW ALL\_TEST  
  
TEXT1: DB '01 SYSTEM ROM TEST',00  
TEXT2: DB '11 VIDED RAM TEST',00  
XXXX: DB '21 SYSTEM RAM TEST',00H  
TEXT3: DB '31 PSG AND SOUND TEST',00

0219' 53 5A 00  
021C' 3A 29 20 20  
0220' 56 44 50 20  
0224' 49 4E 54 45  
0228' 52 52 55 50  
022C' 54 20 54 45  
0230' 53 5A 00  
0233' 35 29 20 20  
0237' 4B 45 59 42  
023B' 4F 41 52 44  
023F' 20 54 45 53  
0243' 54 00  
0245' 36 29 20 20  
0249' 4A 4F 59 54  
024D' 49 43 4B 53  
0251' 20 54 45 53  
0255' 54 00  
0257' 37 29 20 20  
025B' 50 52 49 4E  
025F' 54 45 52 20  
0263' 54 45 53 54  
0267' 00  
026B' 38 29 20 20  
026C' 43 41 53 53  
0270' 45 54 54 45  
0274' 20 54 45 53  
0278' 54 00  
027A' 39 29 20 20  
027E' 41 4C 4C 20  
0282' 54 45 53 54  
0286' 00  
0287' 20 53 45 4C  
028B' 45 43 54 20  
028F' 20 3D 20 2B  
0293' 20 3D 20 2D  
0297' 2D 3E 20 39  
029B' 20 29 20 20  
029F' 3F 20 00  
  
02A2' CD 02AB'  
02A5' C3 02E9'  
02A8' 21 05E6'  
02AB' CD 0000\*  
02AE' 11 0000  
02B1' 01 4000  
02B4' F3  
02B5' CD 0000\*  
02B8' FB  
  
02B9' 21 0000\*  
02BC' 11 0100  
02BF' 01 02FF  
02C2' CD 0000\*  
  
02C5' 21 0000\*  
02C8' 11 0900  
02CB' 01 02FF  
02CE' CD 0000\*  
  
02D1' 21 0000\*  
02D4' 11 1100  
02D7' 01 02FF  
02DA' CD 0000\*

TEXT4: DB '4) VOP INTERRUPT TEST',00  
TEXT5: DB '5) KEYBOARD TEST',00  
TEXT6: DB '6) JOYSTICKS TEST',00  
TEXT7: DB '7) PRINTER TEST',00  
TEXT8: DB '8) CASSETTE TEST',00  
TEXT9: DB '9) ALL TEST',00  
TEXT10: DB 'SELECT = (0 -> 9) ?',00  
  
}  
-----  
}  
J. PSG TESTING ROUTINE  
}  
-----  
}  
----- WRITE DATA TO THE PSG REGISTER -----  
}  
PSG\_TEST:  
LD EE,E  
JP FF,F  
EE\_E1:  
LD HL,VOPFB  
CALL VOPSET  
  
LD DE,0000  
LD EC,4000H  
DI  
CALL CWMLK  
EI  
  
LD HL,AGC11  
LD DE,VFNT+201\*8  
LD EC,2FH  
CALL WAPLK  
  
LD HL,AGC11  
LD DE,VFNT+204B+201\*8  
LD EC,2FH  
CALL WAPLK  
  
LD HL,AGC11  
LD DE,VFNT+204B+2+201\*8  
LD EC,2FH  
CALL WAPLK



```

000' 3E F1          LD A,0FH          ; SET AMPLITUDE TO MAX. VALUE
F' 11 2000         LD DE,VFCT        ; OUTPUT A TONE
02E2' 01 1800     LD BC,1000        ;
02E3' CD 0000*    CALL SMLK         ;
;
02E8' C9          RET
;
02E9'             FF_F1
02E9' 21 05EE'    LD HL,1HSG1       ;
02EC' 11 3947     LD DE,VFNT+10*32+7 ;
02EF' DD 21 02F6' LD IX,4+7         ;
02F3' C3 0000*    JP DMSB          ;
;
02F6' 3E 00          LD A,00H          ; AMPLITUDE = 0
02F8' CD 0000*    CALL AFPA        ; WRITE AMPLITUDE A
02FB' CD 0000*    CALL AFPB        ; WRITE AMPLITUDE B
02FE' CD 0000*    CALL AFPC        ; WRITE AMPLITUDE C
;
0301' 11 0194     LD DE,0194H       ; DE = TONE PERIOD
0304' CD 0000*    CALL TONEA       ; WRITE TUNE A PERIOD
0307' CD 0000*    CALL TONEB       ; WRITE TUNE B PERIOD
030A' CD 0000*    CALL TONEC       ; WRITE TUNE C PERIOD
;
0300' 3E 0F          LD A,0FH          ; A = NOISE PERIOD
030F' CD 0000*    CALL NOISE        ; WRITE NOISE PERIOD
;
0312' 11 3800     LD DE,3800H       ; DE = ENVELOPE PERIOD
0315' 3E 0C          LD A,00H          ; SELECT ENVELOPE REGISTER
;
0317' CD 0000*    CALL PSMWZ       ; WRITE ENVELOPE PERIOD
031A' 1E 00          LD E,00H          ; E = ENVELOPE SHAPE CYCLE
031C' 3E 00          LD A,00H          ; SELECT ENVELOPE SHAPE CYCLE REG.
031E' CD 0000*    CALL PSMW1       ; E -> RI3
;
;----- TIME DELAY FOR THREE SECONDS -----
;
0322' 01 0888     LD EC,3000        ;
0325' CD 0000*    CALL DELAY        ;
0328' C1          POP BC          ;
;
;----- READ DATA FROM REGISTER 0 TO REGISTER 13 -----
;
0329' 06 03          LD B,03H          ; TUNE PERIOD REGISTER COUNTER
032B' 3E 01          LD A,01H          ; TUNE A PERIOD REGISTER
032D' 4F          LD C,A          ; SAVE IT IN C
032E' CD 0000*    CALL PSMR2       ; READ TUNE A PERIOD REGISTER
0331' 7A          LD A,D          ; MOVE COARSE TUNE INTO A
0332' FE 01          CP 01H          ; CHECK COARSE TUNE VALUE
0334' C2 03D2'     JP NZ,FRNT3     ; IF INCORRECT, PRINT ERROR MESSAGE
0337' 7B          LD A,E          ; CORRECT, MOVE FINE TUNE PERIOD
; INTO A
0338' FE 94          CP 94H          ; AGAIN CHECK VALUE
033A' C2 03D2'     JP NZ,FRNT3     ; IF INCORRECT, PRINT ERROR MESSAGE
033D' 79          LD A,C          ; CORRECT, MOVE THE ORIGINAL VALUE
; INTO A
033E' C6 02          ADD A,02H         ; POINT TO NEXT TUNE PERIOD REG.
0340' 05          DEC B          ; UPDATE COUNTER
0341' 20 EA          JR NZ,NXTRG     ; DO UNTIL B = 00
;
0343' 3E 06          LD A,06H          ; NOISE PERIOD REGISTER
0345' CD 0000*    CALL PSMR1       ; READ NOISE PERIOD
0348' 7B          LD A,E          ; MOVE IT INTO A
0349' FE 0F          CP 0FH          ; A-7FH
034B' C2 03D2'     JP NZ,FRNT3     ; NONZERO, PRINT ERROR MESSAGE
;
034E' 06 03          LD B,03H          ; ZERO, SET REGISTER COUNTER
0350' 3E 08          LD A,08H          ; CHANNEL A AMPLITUDE REGISTER
0352' 4F          LD C,A          ; SAVE A IN C
0353' CD 0000*    CALL PSMR1       ; READ CHANNEL A AMPLITUDE
0356' 7B          LD A,E          ; MOVE IT INTO A
0357' FE 00          CP 00H          ; A-00H
0358' C2 03D2'     JP NZ,FRNT3     ; NONZERO, PRINT ERROR MESSAGE
;
0359' 0C          INC C          ;
035D' 79          LD A,C          ; NEXT AMPLITUDE REGISTER
035E' 05          DEC B          ; UPDATE COUNTER
035F' 20 F1          JR NZ,NXTRG2     ; DO UNTIL B = 00
;
0361' 3E 0F          LD A,0FH          ;
0363' CD 0000*    CALL AFPA        ;
0366' C5          PUSH BC         ;
0367' 01 03EB     LD BC,1000        ;
036A' CD 0000*    CALL DELAY        ; SET 1 SECOND TIME DELAY
036D' C1          POP BC         ;
036E' CD 0000*    CALL OFFCHA       ; TURN OFF CHANNEL A
;
0371' C5          PUSH BC         ;
0372' 01 03EB     LD BC,1000        ;
0375' CD 0000*    CALL DELAY        ; SET 1 SECOND TIME DELAY
0378' C1          POP BC         ;
0379' 3E 0F          LD A,0FH          ;
037B' CD 0000*    CALL AFPA        ;
037E' C5          PUSH BC         ;
037F' 01 03EB     LD BC,1000        ;
0382' CD 0000*    CALL DELAY        ; SET 1 SECOND TIME DELAY
0385' C1          POP BC         ;
0386' CD 0000*    CALL OFFCHB       ; TURN OFF CHANNEL B
0389' C5          PUSH BC         ;
038A' 01 03EB     LD BC,1000        ;
038D' CD 0000*    CALL DELAY        ; SET 1 SECOND TIME DELAY
0390' C1          POP BC         ;
0391' 3E 0F          LD A,0FH          ;
0393' CD 0000*    CALL AFPA        ;
0396' C5          PUSH BC         ;
0397' 01 03EB     LD BC,1000        ;
039A' CD 0000*    CALL DELAY        ; SET 1 SECOND TIME DELAY
039D' C1          POP BC         ;
039E' CD 0000*    CALL OFFCHC       ; TURN OFF CHANNEL C
;
03A1' 3E 0C          LD A,00H          ; ENVELOPE PERIOD REGISTER
03A3' CD 0000*    CALL PSMR2       ; READ ENVELOPE PERIOD
03A6' 7A          LD A,D          ; MOVE COARSE TONE ENVELOPE PERIOD
03A7' FE 38          CP 38H          ; CHECK
03A9' C2 03D2'     JP NZ,FRNT3     ; NONZERO, PRINT ERROR MESSAGE
03AC' 7B          LD A,E          ; ZERO, MOVE FINE TONE ENVELOP PER.
03AD' C1          CP 00H          ; CHECK
03AF' C2 03D2'     JP NZ,FRNT3     ; NONZERO, PRINT ERROR MESSAGE
;
03B2' 3E 00          LD A,00H          ; ZERO, SET ENVELOP SHAPE CYCLE REG
03B4' CD 0000*    CALL PSMR1       ;
03B7' 7B          LD A,E          ; MOVE IT INTO A
03B8' FE 00          CP 00H          ; CHECK
03BA' C2 03D2'     JP NZ,FRNT3     ;
;
03BD' 21 0635'     LD HL,NXTRG     ;HL = 'PSG OK'
03C0' 11 3987     LD DE,VFNT+12*32+7 ;DE = VFMH ADDR
03C3' DD 21 03CA' LD IX,4+7         ;
03C7' C3 0000*    JP DMSB          ;
;
03CA' 21 069E'     LD HL,1HSG1       ;HL = 'PSG OK'
03CD' CD 0000*    CALL FRNT3B       ;PRINT PSG OK
;
03D0' 18 13          JR SMD          ; GOTO SOUND TESTING ROUTINE
;
;----- PRINT PSG ERROR MESSAGE -----
;
03D2' 21 0642'     LD HL,NXTRG     ;HL = 'PSG ERROR'
03D5' 11 3987     LD DE,VFNT+13*32+7 ;
03D8' DD 21 03DF' LD IX,4+7         ;
03DC' C3 0000*    JP DMSB          ;
;
03DF' 21 06A7'     LD HL,1HSG2       ;HL = 'PSG ERROR'
;
03E2' CD 0000*    CALL FRNT3B       ;PRINT PSG ERROR
;
;----- SOUND TESTING ROUTINE -----
;
03E5' 01 0800     SMD: LD TC,800H          ;
03E8' CD 0000*    CONTINU: CALL SBLND        ;
03EB' 0B          DEC TC          ; UPDATE COUNTER
03EC' 7B          LD A,B          ;
03ED' 81          OR C          ;
03EE' 20 FB          JR NZ,CONTINU     ; DO UNTIL BC = 00
;
;----- PSG TESTING COMPLETED -----
;
03F0' 3A 000E*    LD A,(RUM)        ;
03F3' FE 09          CP 09H          ;

```

```

; **** test interrupt ****
00C3 JMP EQU OC3H

INT_TEST:
03F8 F3 DI ; DISABLE INTERRUPT

;
03F9 DB A8 IN A, (RPP1A) ; FORCE 0000-3FFF AS SLOT 1
03FB E6 FC AND 1111100B
03FD F6 01 OR 0000001B
03FF D3 A8 OUT (NP1A),A

;
0401 3E C3 LD A, JMP ; INSERT JMP VECTOR
0403 32 003B LD (INTVECT),A ;
0406 21 04A7 LD HL, VDPINT ; MOVE INT. VECTOR TO RST3B (FOR MS1)
0409 22 0039 LD (INTVECT+1),HL ;

;
040C 3E C0 LD A, C00H ;
040E 0E 01 LD C, 01H ;
0410 CD 0000 CALL WDFR8 ; DISABLE VDP INT

;
0413 3E 55 LD A, 55H ;
0415 32 0000 LD (IFLAG),A ; SET INT FLAG

;
0418 3E E0 LD A, 0E0H ; ENABLE VDP INTERRUPT
041A 0E 01 LD C, 01H ; VDP REB. 1
041C CD 0000 CALL WDFR8 ;
041F 06 3C LD B, 60 ; SET TIMER1 COUNTER
0421 FB EI ; ENABLE INTERRUPT

;
0422 C5 TIMER1: PUSH BC ;
0423 01 0001 LD BC, 0001H ;
0426 CD 0000 CALL DELAY ; DELAY FOR 1 MSEC
0429 C1 POP BC ;
042A 10 F6 DJNZ TIMER1 ;

;
042C F3 DI ;
042D 3E 0C LD A, 0CH ;
042F 0E 01 LD C, 01H ;
0431 CD 0000 CALL WDFR8 ; DISABLE VDP INT

;
0434 3A 0000 LD A, (IFLAG) ;
0437 FE 57 CP 57H ; CHECK IS TWO OR MORE INT IN 60 MSEC
0439 D2 04C9 JP NC, INTOK ; IF OK, JMP AND PRINT 'INT OK' MESSAGE

;
043C 21 05E6 LD HL, VDPREB ;
043F CD 0000 CALL VDPSET ;

;
0442 11 0000 LD DE, 0000 ;
0445 01 4000 LD BC, 4000H ;
0448 F3 DI ;
0449 CD 0000 CALL CVMBLK ;
044C FB EI ;

;
044D 21 0000 LD HL, ASCII ;
0450 11 0100 LD DE, VPGT+20H+B ;
0453 01 02FF LD BC, 2FFH ;
0456 CD 0000 CALL WMBLK ;
0459 21 0000 LD HL, ASCII ;
045C 11 0900 LD DE, VPGT+204B+20H+B ;
045F 01 02FF LD BC, 2FFH ;
0462 CD 0000 CALL WMBLK ;
0465 21 0000 LD HL, ASCII ;
0468 11 1100 LD DE, VPGT+204B+2+20H+B ;
046B 01 02FF LD BC, 2FFH ;
046E CD 0000 CALL WMBLK ;

;
0471 3E F1 LD A, 0FH ;
0473 11 2000 LD DE, VPCT ;
0476 01 1800 LD BC, 256+B*3 ;
0479 CD 0000 CALL SMBLK ;

;
047C 21 0673 LD HL, XMSG9 ;
047F 11 3947 LD DE, VINT+10*32+7 ;
0482 DD 21 04B9 LD IX, #+7 ;
0486 C3 0000 JP DISEB ;

;
0489 21 0652 LD HL, XMSG7 ; HL = 'INT ERROR'
048C 11 39E7 LD DE, VINT+15*32+7 ;
048F DD 21 0496 LD IX, #+7 ;
0493 C3 0000 JP DISEB ;

;
0496 21 06B3 LD HL, XMSG3 ; HL = 'INT ERROR'
0499 CD 0000 CALL PRINT8 ; PRINT INT ERROR

;
049C 3A 000E LD A, 0EH ;
049F FE 09 CP 09H ;
04A1 C2 0020 JP NZ, OPTION ;
04A4 C3 0000 JP KEY_TEST ;

; ----- VDP INTERRUPT ROUTINE -----
;
VDPINT: PUSH AF ; SAVE ALL REGISTER
; PUSH BC ;
; PUSH DE ;
; PUSH HL ;
LD A, (IFLAG) ;
CP 0C3H ; IS FRIST INT IN 60 MSEC
JR NZ, VDP2 ; NO, JMP
INC A ;
LD (IFLAG),A ; YES, RE-INIT INT FLAG
JR VDP1 ;

;
VDP2: CP 56H ; IS SECOND INT IN 60 MSEC
JR NZ, VDP1 ; NO, JMP
INC A ;
LD (IFLAG),A ; YES, RE-INIT INT FLAG

;
VDP1: CALL WDFR8 ; READ VDP STATUS
EI ;
POP HL ;
POP DE ;
POP BC ;
POP AF ;
RET ;

;
INTOK: LD HL, VDPREB ;
CALL VDPSET ;

;
LD DE, 0000 ;
LD BC, 4000H ;
DI ;
CALL CVMBLK ;
EI ;

;
LD HL, ASCII ;
LD DE, VPGT+20H+B ;
LD BC, 2FFH ;
CALL WMBLK ;
LD HL, ASCII ;
LD DE, VPGT+204B+20H+B ;
LD BC, 2FFH ;
CALL WMBLK ;
LD HL, ASCII ;
LD DE, VPGT+204B+2+20H+B ;
LD BC, 2FFH ;
CALL WMBLK ;

;
LD A, 0FH ;
LD DE, VPCT ;
LD BC, 256+B*3 ;
CALL SMBLK ;

;
LD HL, XMSG8 ; HL = 'VDP INT OK'

```

```

0519' 11 3A07          LD DE,VPNT+16*32+7
051C' 00 21 0523'    LD IX,*+7
0520' C3 0000*      JP DMSG

0523' 21 06C3'      LD HL,FM564      ;HL = INT OK'
0526' 00 0000*      CALL FRM564     ;PRINT INT OK

0529' F3            KEYCK1: DI
052A' 21 0686'      LD HL,XMSG10
052D' 11 3A81      LD DE,VPNT+20*32+1
0530' 00 21 0537'  LD IX,*+7
0534' C3 0000*      JP DMSG

; --- CHECK CTRL-STOP ---
;
; WAIT: IN A,(PPICR) ;
; AND OF0H ;
; OR 06H ;
; OUT (WPPI),A ;
; IN A,(RPP1B) ;SCAN LINE DATA
; CPL ;
; CP ZH ;
; JP Z,KEY_TEST ;JMP TO KEYBOARD TEST
; JR WAIT ;

0537' DB AA          WAIT: IN A,(PPICR) ;CHECK IF CTRL-STOP (FOR MS1 USE)
0539' E6 F0          AND 11110000B ;
053B' F6 06          OR 00001100B ;SCAN LINE 6
053D' 03 AA          OUT (WPPI),A ;
053F' DB A9          IN A,(RPP1B) ;
0541' FE FD          CP 11111010B ;IS CTRL KEY PRESSED
0543' 20 F2          JR NZ,WAIT ;NO, WAIT AGAIN

0545' DB AA          IN A,(PPICR) ;
0547' E6 F0          AND 11110000B ;
0549' F6 07          OR 00001110B ;SCAN LINE 7
054B' 03 AA          OUT (WPPI),A ;
054D' DB A9          IN A,(RPP1B) ;
054F' FE EF          CP 11101111B ;IS STOP KEY PRESSED
0551' 20 E4          JR NZ,WAIT ;NO, WAIT AGAIN

0553' 3A 000E*      LD A,(FUND)
0556' FE 09          CP 09H
0558' C2 0020*     JP NZ,OPTION
055B' C3 0000*      JP KEY_TEST ;JMP TO KEYBOARD TEST

; --- CASSETTE TESTING ROUTINE ---
;
; --- DETECT READY SIGNAL ---
;
CAS_TEST:
LD HL,VDFREB
CALL VDFSET

LD DE,0000
LD BC,4000H

056A' F3            DI
056B' 00 0000*     CALL DWBLK

056E' F3            DI
056F' 21 0000*     LD HL,ASC11
0572' 11 0100     LD DE,VPNT+201H*8
0575' 01 02FF     LD BC,2FFH
0578' 00 0000*     CALL WMBLK
057B' 21 0000*     LD HL,ASC11
057E' 11 0900     LD DE,VPNT+204B*201H*8
0581' 01 02FF     LD BC,2FFH
0584' 00 0000*     CALL WMBLK
0587' 21 0000*     LD HL,ASC11
058A' 11 1100     LD DE,VPNT+204B*2+201H*8
058D' 01 02FF     LD BC,2FFH
0590' 00 0000*     CALL WMBLK

0593' 3E F1          LD A,OF1H
0595' 11 2000     LD DE,VPCT
0598' 01 1800     LD BC,256*8+3
059B' 00 0000*     CALL SWBLK

059E' FB            EI

059F' 21 03FF'      LD HL,MESS1
05A2' 11 3947      LD DE,VPNT+10*32+7
05A5' 00 21 059C'  LD IX,*+7
05A9' C3 0000*      JP DMSG

05AC' 21 05D1'      LD HL,MESS2
05AF' 11 3984      LD DE,VPNT+12*32+4
05B2' 00 21 0599'  LD IX,*+7
05B6' C3 0000*      JP DMSG

05B9' 00 0000*     CALL DELAY1
05BC' C3 0020'     JP OPTION

05BF' 43 41 53 53  ;MESS1: DB 'CASSETTE MUST BE ',00
05C3' 45 54 54 45 ;
05C7' 20 40 55 53 ;
05CB' 54 20 42 45 ;
05CF' 20 00 ;
05D1' 54 45 53 54 ;MESS2: DB 'TESTED THROUGH BASIC',00
05D5' 45 44 20 54 ;
05D9' 48 52 4F 55 ;
05DD' 47 4B 20 42 ;
05E1' 41 53 49 43 ;
05E5' 00 ;

; COMMENT %
;
; << SKIP CASSETTE TEST >>
;
;
; --- CHECK IS CASSETTE READY ---
;
CAS1: IN A,(RPP1A) ;A = IOIXXXXX
BIT 6,A ;
JR NZ,CAS1 ;

; --- OUTPUT A CASION SIGNAL TO THE CASSETTE ---
;
LD A,0FH ;
OUT (WPPI),A ;A = 11101111

LD HL,XMSG2
LD DE,VPNT+10*32+1
LD IX,*+7
JP DMSG

LD A,0FH ;
OUT (SERLAT),A ;SET FSG LATCH

; --- FLASH CAP LAMP ---
;
FLASH: IN A,(RPP5G) ;
XOR 00100000B ;
OUT (WPPI),A ;

LD BC,8000H
DI: DEC BC
LD A,C
OR B
JR NZ,DI

; --- CHECK CTRL-STOP ---
;
IN A,(PPICR) ;
AND OF0H ;
OR 06H ;
OUT (WPPI),A ;
IN A,(RPP1B) ;SCAN LINE DATA
CPL ;
CP ZH ;
JP Z,OPTION ;
JR FLASH Z ;

```

THE WHOLE TESTING ROUTINE COMPLETED

DATA DEFINE

VDP REGISTER DATA

05E6' 02
05E7' 00
05E8' CE
05E9' FF
05EA' 03
05EB' 76
05EC' 03
05ED' 00
05EE' 20 20 20 20
05F2' 20 20 50 53
05F6' 47 20 54 45
05FA' 53 51 00
05FD' 50 52 45 53
0601' 53 20 43 54
0605' 52 4C 20 53
0609' 54 4F 50 20
060D' 54 4F 20 52
0611' 45 20 53 54
0615' 41 52 54 00
0619' 20 20 20 20
061D' 20 20 50 52
0621' 45 53 53 20
0625' 50 4C 41 59
0629' 20 4F 4E 20
062D' 54 41 50 45
0631' 20 20 20 00
0635' 20 20 20 20
0639' 20 20 50 53
063D' 47 20 4F 4B
0641' 00
0642' 20 20 20 20
0646' 20 20 50 53
064A' 47 20 45 52
064E' 52 4F 52 00
0652' 20 20 20 20
0656' 20 20 49 4E
065A' 54 20 45 52
065E' 52 4F 52 00
0662' 20 20 20 20
0666' 20 20 56 44
066A' 50 20 49 4E
066E' 54 20 4F 4B
0672' 00
0673' 56 44 50 20
0677' 49 4E 54 45
067B' 52 52 53 50
067F' 54 20 54 45
0683' 53 54 00
0686' 50 52 45 53
068A' 53 20 43 54
068E' 52 4C 20 53
0692' 54 4F 50 20
0696' 54 4F 20 45
069A' 58 49 54 00
069E' 50 53 47 20
06A2' 4F 4B
06A4' 00 0A 00
06A7' 50 53 47 20

VDPREG: DB VR0
DB VR1
DB VR2
DB VR3
DB VR4
DB VR5
DB VR6
DB VR7
XMSG1: DB 'PSG TEST',00
XMSG2: DB 'PRESS CTRL-STOP TO RE-START',00
XMSG4: DB 'PRESS PLAY ON TAPE ',00
XMSG5: DB 'PSG OK',00
XMSG6: DB 'PSG ERROR',00
XMSG7: DB 'INT ERROR',00
XMSG8: DB 'VDP INT OK',00
XMSG9: DB 'VDP INTERRUPT TEST',00
XMSG10: DB 'PRESS CTRL-STOP TO EXIT',00
XMSG11: DB 'PSG OK'
DB 00H,00H,00H
XMSG12: DB 'PSG ERROR'

06AB' 45 52 52 4F
06AF' 52
06B0' 00 0A 00
06B3' 56 44 50 20
06B7' 49 4E 54 20
06EB' 45 52 52 4F
06EF' 52
06F0' 00 0A 00
06F3' 56 44 50 20
06F7' 49 4E 54 20
06FB' 4F 4B
06FD' 00 0A 00
06D0'
0000"
0001"
0004"
0007"
0008"
000E" 01

DB 00H,00H,00H
XMSG3: DB 'VDP INT ERROR'
DB 00H,00H,00H
XMSG4: DB 'VDP INT OK'
DB 00H,00H,00H
dseg
IFLAG: DS 1 ;INT FLAG
POST1: DS 3 ;CLR 1 POS
POST2: DS 3 ;CLR 2 POS
STORE: DS 01H
RVALUE: DS 6 ;ASCII CONVERT
FUN: DS 01H ;FUNCTION SELECTION
END START

Macros:

Symbols:

01B3' ALL\_TEST
0374' ANTC
053E1' CAS\_TEST
0427' DELAY
05B7' DMSG
02E9' FF\_F
0105' FUN\_SEL
0000' INTOK
03B' INT\_TEST
01C2' JUV\_TEST
00E7' KEY\_IN
01F1' KEY\_K
01B7' MPM
0352' NKTNG
03F7' OFFCHB
06A7' MSG2
0179' INT\_ERR
00B2' P1CHD
0000' PRINT
0527' PRINTD
000F' PSEPTB
031F' PSMI
000B1' RAMPUF
013E1' RM\_IT
0098' RVDATA
0000' RMEET
1B00' SGPTR
01B0' SWM\_TEST
001B' START2
04B0' SINC2
059C' SWBLK
01DE' TEXT2
0233' TEXT5
026B' TEXTD
0305' TIMEA
04C0' VDP1
05E61' VDPREG
0001' VBT
0000' VR0
0003' VR4
0000' VR7
1B00' VERGT
000A' WPTC
0432' WVDTRB
0000' WMEET
03F0' XMSG2
0642' XMSG6
0673' XMSG9
0364' AMPA
050B' ASCII
03EB' CONTINU
05B0' DELAY1
024B' EE\_E
0000' FIRE
0107' FUN\_SEL1
04C9' INTOK
00C3' JNP
0000' KEYOVS
00EE' KEY\_IN1
050F' MESS1
0310' NOISE
036F' OFFCHA
00201' OPTION
06B3' PMSB3
00011' POST1
00A8' P1CHP
01C4' PRINT\_TEST
0000' PSELAT
03B5' PSEL1
031B' PSEL2
00A2' RDPSS
000B' RYPIA
04C1' RVDFRG
0079' RVSTAT
03E5' SNO
0011' START
0007' STORE
0079' SWADDR
01CA' TEXT1
0205' TEXT3
0245' TEXT6
027A' TEXT9
030B' TONEB
04B0' VDP2
0562' VDPSET
3B001' VPT
000E' VR2
0076' VR5
01B8' WVM\_TEST
0537' WAIT
00A1' WPSB
0591' WMBLK
05EE' XMSG1
0619' XMSG4
0652' XMSG7
0163' XXXX

```

|-----|
| RMTST | sv-318/sv-328 ram/vram test program
|-----|
|       | by Raymond Y. Cheung
|
|       |
|       | REVISD BY W.K. LEUNG FOR MSX 728 TEST
|-----|
|
| .780
| TITLE RMTST.MAC RAM/VRAM test of SVTEST.MAC
| EXTERNAL ASCII,t,s28,PSB_TEST,OPTION,FUN,ROM
|
| entry VRAM_TEST,SRAM_TEST,DHSS,oldsit
|
|-----VDP INIT. PARAMETERS-----|
|
| VR0 EQU 02H ;KS=1
| VR1 EQU 0C2H ;16K,BLANK=1,INT=0,MODE 2,SIZE=0,MAG=X2
| VR2 EQU 0EH ;VFNT=3B00H
| VR3 EQU 0FFH ;VPCT=2000H(80H)
| VR4 EQU 03H ;VFST=3B00H(00H)
| VR5 EQU 7EH ;VSTAT=3B00H
| VR6 EQU 03H ;VSPGT=1B00H
| VR7 EQU 00H ;BACKDROP=TRANSPARENT
|
|-----VRAM MAP-----|
|
| VPST EQU (VR4 AND 04H)+800H ;PATTERN GENERATOR TABLE BASE
| VSPBT EQU VR6+800H ;SPRITE PATTERN GEN. BASE
| VPCT EQU (VR3 AND 80H)+40H ;COLOUR TAB. BASE
| VFNT EQU VR2+400H ;PATTERN NAME TAB. BASE
| VSTAT EQU VR5+80H ;SPRITE ATTRIBUTE TABLE BASE
|
| psglat equ 010H ; psg latch port
| rdpsg equ 0A2H ; read psg data port
| wrpsg equ 0A1H ; write psg data port
|
|
| ppicmd equ 82h ; ppi cmd word
| ppicrp equ 0abh ; ppi cmd port
| ppiopa equ 0a8h ; ppi port a
| ppiopb equ 0a7h ; ppi port b
| ppiopc equ 0aah ; ppi port c write
| ppiorc equ 0aah ; ppi port c read
|
|----- test area parameter -----|
|
| vstart equ 0000h ; vram test area
| vepage equ 40h ; end page
| start equ 8000h ; start
| epage equ 00h ; end page
| trans equ 0c000h ; transfer loc of test module
| tsize equ 200h ; size of test module
|
| oldsit equ trans*tsize ; temp. storage of slot assignment
|
|-----HARDWARE INITIALIZATION-----|
|
| ;
| ; jp 0000h ; dummy
| 0000' F3 VRAM_TEST: DI
| ;
| ;----- VDP INITIALIZATION -----|
| ;
| 0001' 21 019C' begin: LD HL,VDPREG
| 0004' 00 21 000B' ld ix,#+7 ; load return addr
| 0008' C3 03F5' jp VDPSET ;VDP INITIALIZATION
|
|-----|
| test vram from 0000 to 3fff
|-----|

```

```

0008' 06 00          ;
                    ld    b,0          ; pattern modifier
                    ;
0000' 11 0000        ckvram: ld    de,vstart        ; wr vram addr set
0010' 00 21 0017'    ld    ix,$+7
0014' C3 0418'      jp    wvset
                    ;
                    ;--- fill vram as data = 1 xor h xor b ---
                    ;
0017' 7B            vfill: ld    a,e          ; get a = 1 xor h xor b
0018' AA            xor    d
0019' AB            xor    b
001A' 03 9B        out    (wdata),a        ; test data-->(vram) (8 u sec need)
001C' 13            inc    de          ; inc addr
001D' 7A            ld    a,d          ; check end of page
001E' FE 40        cp    vepage
0020' 20 F5        jr    nz,vfill        ; fill pattern loop
                    ;
                    ;--- cap lamp "on" ---
                    ;
                    ;    ld    a,15          ; latch psq at reg 15
                    ;    out    (psqlat),a
                    ;    in    a,(rdpsq)        ; read bank data, cap lamp bit
                    ;    or    00100000b        ; set cap lamp bit on
                    ;    out    (wrpsq),a        ; turn on led
                    ;
                    ;--- cap lamp on (for mx) ---
                    ;
0022' DB AA        in    a,(ppirc)        ; read ppi port c
0024' E6 EF        and    10111111b        ; force cap lamp on
0026' 03 AA        out    (ppipoc),a
                    ;
                    ;--- check vram from vstart to vepage ---
                    ;
0028' 11 0000        ld    de,vstart        ; rd vram addr set
002B' 00 21 0032'    ld    ix,$+7
002F' C3 0424'      jp    rvset
                    ;
0032' DD 23        inc    ix          ; delay for read vram addr set
0034' DD 2B        dec    ix
                    ;
0036' DB 9B        vchecks: in    a,(rvdata)        ; read vram data (8 u sec need)
0038' 4F            ld    c,a          ; temp save read data
0039' 7B            ld    a,e
003A' AA            xor    d          ; a = 1 xor h xor b
003B' AB            xor    b
003C' B9            cp    c          ; is error ?
003D' C2 0272'      jp    nz,verror        ; vram error jmp
0040' 13            inc    de          ; next addr
0041' 7A            ld    a,d
0042' FE 40        cp    vepage        ; is end of page ?
0044' 20 F0        jr    nz,vcheck        ; loop back if no error
                    ;
                    ;--- cap lamp off ---
                    ;
                    ;    in    a,(rdpsq)        ; psq latching at reg. 15
                    ;    and    11011111b        ; turn off cap lamp bit
                    ;    out    (wrpsq),a
                    ;
                    ;--- cap lamp off (for mx) ---
                    ;
0046' DB AA        in    a,(ppirc)        ; read ppi port c
0048' F6 40        or    01000000b        ; force cap lamp on
004A' 03 AA        out    (ppipoc),a
                    ;
                    ;--- update pattern modifier ---
                    ;
004C' 7B            ld    a,b          ; is b= 0 ?
004D' B7            or    a
004E' 20 04        jr    nz,nzvra        ; nz - flow
                    ;
                    ;..... z - flow ....
                    ;
0050' 04            inc    b          ; b=0, => make B = 0000 0001
0051' C3 0000'      jp    ckvram        ; do again

```

```

).... nz - flow ....
)
0054' CB 20          nzvrat sla b          ; (cy) <- /... B .../ <-0
0056' C2 0000'      jp      nz,ckvram ; do again, until the "I" shift to (cy)
)
) .... else, finish ....
)
0059' 3A 0000#     LD A,(FUN)      ; GET FUNCTION SELECT NO.
005C' FE 09       CP 09H
005E' C2 0000#     JP NZ,OPTION
)
)-----
)      display message on screen
)-----
)
0061'              BRAM_TEST:
)
)-----CLEAR ALL VRAM-----
)
0061' 3A 0000#     LD A,(FUN)
)
)
0064' 5F          LD E,A
0065' D9          EIX
0066' 01 4000     LD BC,4000H      ;CLEAR 16 K VRAM
0069' 11 0000     LD DE,VPGT
006C' D0 21 0073' ld ix,*+7
0070' C3 03C8'   jp      CVMBLK
)
)
0073' 21 0000#     LD HL,ASCII      ; SET DWR PAT IN FILE: DWRSET
0076' 11 0100     LD DE,VPGT+20H#B ; PNT BECOMES ASCII PATTERNS
0079' 01 02FF     LD BC,2FFH
007C' D0 21 00E3' ld ix,*+7
0080' C3 03E0'   jp      WVMBLK
)
)
0083' 21 0000#     LD HL,ASCII      ; 2ND PORTION
0086' 11 0900     LD DE,VPGT+2048+20H#B
0089' 01 02FF     LD BC,2FFH
008C' D0 21 0093' ld ix,*+7
0090' C3 03E0'   jp      WVMBLK
)
)
0093' 21 0000#     LD HL,ASCII      ; 3RD PORTION
0096' 11 1100     LD DE,VPGT+2048+2+20H#B
0099' 01 02FF     LD BC,2FFH
009C' D0 21 00A3' ld ix,*+7
00A0' C3 03E0'   jp      WVMBLK
)
)
00A3' 3E F1       ld a,0fh         ; text char color
00A5' 11 2000     ld de,vpct      ; pattern color tbl
00A8' 01 1800     ld bc,256*8+3   ; 3 portions
00AB' D0 21 00B2' ld ix,*+7
00AF' C3 03C9'   jp      svmb     ; set vram block
)
)
)-----determine whether sv-318 or sv-328 is running-----
)
)
00B2' 3E AA       ld a,0aah       ; write a pattern to ram
00B4' 32 B000     ld (B000h),a    ; can it be write ?
00B7' 3A B000     ld a,(B000h)    ; read A
)
)
)      ld a,5ch         ; force to check upper 16k only (for msx)
)
)
00BA' FE AA       cp      0aah     ; is same value ?
00BC' 20 0F       jr      nz,sys318 ; sv-318 system if not same
)
)
)      ld a,055h       ; double varify
)
)
00C0' 32 B000     ld (B000h),a
00C3' 3A B000     ld a,(B000h)    ; is same as write ?
00C6' FE 55       cp      055h
00C9' 20 03       jr      nz,sys318 ; no, it is not sv328
00CA' C3 0173'   jp      sys328
)
)-----
)      sv-318 system determined
)-----
)
)
00CD' 21 0444'   sys318: ld hl,asglx ; 'sv-318 functional test'
00D0' 11 3847     ld de,vpnt+2*32+7

```

```

0003' D0 21 00DA'      ld    ix,0+7
0007' C3 03B2'        jp    msg          ; disp message on screen
}
000A' 11 3B47          ld    de,vpnt+2*32+7
000D' D0 21 00E4'      ld    ix,0+7
00E1' C3 03B2'        jp    msg          ; disp message on screen
}
00E4' 21 04E4'          ld    hl,msg4x          ; 'loc: C000-ffff testing'
00E7' 11 3BC7          ld    de,vpnt+6*32+7
00EA' D0 21 00F1'      ld    ix,0+7
00EE' C3 03B2'        jp    msg          ; disp message
}
00F1' 21 04B6'          ld    hl,msg3x          ; '16k ran pat rd wr addr'
00F4' 11 3747          ld    de,vpnt+10*32+7
00F7' D0 21 00FE'      ld    ix,0+7
00FB' C3 03B2'        jp    msg
}
}----- memory test from C000 to FFFF only -----
}
00FE' 06 00            ld    b,0            ; clear b patrn modifier
}
}----- load up memory -----
}
loopx: ld    c,b            ; c = data to disp on screen
0100' 4B                ld    de,vpnt+12*32+15 ; pattern modifier disp
0101' 11 370F          ld    ix,0+7
0104' D0 21 010B'      jp    wdata          ; disp sub
}
010B' 21 C000          ld    hl,0c000h      ; get starting addr
010E' 7D                fillx: ld    a,l          ; low byte to accm
010F' AC                xor    h              ; xor with high byte
0110' A8                xor    b              ; xor with pattern
}
0111' 77                ld    (hl),a          ; store in test addr
0112' Z3                inc    hl              ; increment addr
0113' 7C                ld    a,h              ; load high byte of addr
0114' FE 00            cp    epage           ; compare with stop addr
0116' C2 010E'        jp    nz,fillx       ; not done, go back
}
}----- delay the refresh -----
}
0119' 2E 10            ld    l,10h          ; outer loop
}
leakx: xor    a            ; a = 00 to 12B
011B' 7F                refhx: ld    r,a          ; a -> refresh address
011C' ED 4F            ld    r,a          ; delay refresh addr
011E' ED 4F            ld    r,a          ; it makes the end refresh addr wait longer
0120' ED 4F            inc    a              ; a = a + 1
0122' 3C                cp    12B            ; is a = end of one refresh cycle ?
0123' FE 80            jr    nz,refhx       ; loop back
0125' 20 F3            dec    l              ; l reg as counter of outer loop
0127' 2D                jr    nz,leakx
012B' 20 F1
}
}----- cap lamp on -----
}
}
}    in    a,(rdpsg)      ; psq latching at reg. 15
}
}    or    00100000b      ; turn on cap lamp bit
}    out    (wrpsg),a
}
}----- cap lamp on (for nsx) -----
}
}
012A' DB A0            in    a,(ppirdc)     ; read ppl port c
012C' E6 EF            and    10111111b     ; force cap lamp on
012E' D3 A0            out    (pploc),a
}
}----- read and check test data -----
}
}
0130' 21 C000          ld    hl,0C000h      ; get starting addr
0133' 7D                testx: ld    a,l          ; load low byte
0134' AC                xor    h              ; xor with high byte
0135' A3                xor    b              ; xor with modifier
0136' 4E                ld    c,(hl)
0137' B7                cp    c              ; compare with memory loc
0138' C2 0343'        jp    nz,error       ; error exit
0138' Z3                inc    hl              ; update memory address
013C' 7C                ld    a,h              ; load high byte
013D' FE 00            cp    epage           ; compare with stop addr
013F' C2 0133'        jp    nz,testx       ; loop back

```



```

}
} --- cap lamp off ---
}
}   in   a,(rdpsg)   ; psg latching at reg. 15
}   and  11011111b  ; turn off cap lamp
}   out  (wrpsg),a   ;
}
} --- cap lamp off (for msx) ---
}
0142' DB AA           ;   in   a,(ppirdc)   ; read ppi port c
0144' F6 40           ;   or   01000000b   ; force cap lamp on
0146' D3 AA           ;   out  (ppipoc),a   ;
}
}
0148' 78              ;   ld   a,b           ;
0149' B7              ;   or   a             ;
014A' C2 0151'        ;   jp   nz,nzfillx   ; if not zero, else if zero
}
}
014D' 04              ;   inc  b             ; B = 0000 0001
014E' C2 0100'        ;   jp   nz,loopx     ; rst with new modifier
}
}
0151' C8 20           ;   nzfillx: sla    b   ; (cy) (- /.... B ....) (- 0
0153' C2 0100'        ;   jp   nz,loopx     ; loop back if not done
}
} --- disp memory test ok ---
}
0156' 11 38D6         ;   ld   de,vpnt+6*32+22 ;
0159' 21 0512'        ;   ld   hl,msg4b       ; 'O.K.'
015C' D0 21 0163'     ;   ld   ix,$+7         ;
0160' C3 03B2'        ;   jp   dmsg           ; disp message
0163' D9              ;   EXX                ;
0164' 7B              ;   LD A,E              ;
0165' 32 0000*        ;   LD (FUN),A         ;
}
}
0168' 3A 0000*        ;   EXIT: LD A,(FUN)   ;
016B' FE 09           ;   CP 09H              ;
016D' C2 0000*        ;   JP NZ,OPTION        ;
0170' C3 0000*        ;   JP PSG_TEST         ;
}
}
} -----
}   sv-32B determined
} -----
}
0173' 21 042D'        ;   sys32B: ld   hl,msg1   ; 'sv-32B functional test'
}
}
0176' 11 3847         ;   ld   de,vpnt+2*32+7 ;
0179' D0 21 0180'     ;   ld   ix,$+7         ;
017D' C3 03B2'        ;   jp   dmsg           ; disp message on screen
}
}
0180' 21 04CD'        ;   ld   hl,msg4         ; 'loc: B000-ffff testing'
0183' 11 38C7         ;   ld   de,vpnt+6*32+7 ;
0186' D0 21 018D'     ;   ld   ix,$+7         ;
018A' C3 03B2'        ;   jp   dmsg           ; disp message
}
}
018D' 21 049F'        ;   ld   hl,msg3         ; '32k ram pat rd wr addr'
0190' 11 3947         ;   ld   de,vpnt+10*32+7 ;
0193' D0 21 019A'     ;   ld   ix,$+7         ;
0197' C3 03B2'        ;   jp   dmsg           ;
}
}
019A' 18 08           ;   jr   select         ; skip the following data
}
} --- VDP REGISTER DATA ---
}
019C' 02             ;   VDPREG: DB   VR0     ; /*000000/MS/EV/*
019D' C2             ;   DB   VR1     ; /* 4/16K /BLANK/IE/M1/MS2/0/SIZE/MMS/*
019E' 0E             ;   DB   VR2     ; /*0/0/0/0/NAME TABLE BASE ADDR/*
019F' FF             ;   DB   VR3     ; /*COLOUR TABLE BASE ADDR/*
01A0' 03             ;   DB   VR4     ; /*00000/PAT GEN BASE ADDR/*
01A1' 76             ;   DB   VR5     ; /*0/SPRITE ATTRIBUTE TABLE BASE ADDR/*
01A2' 03             ;   DB   VR6     ; /*00000/SPRITE PAT GEN BASE ADDR/*
01A3' 00             ;   DB   VR7     ; /*TEXT COLOUR 1/ TEXT COL. 0 OR BACKGROUND COL/*
}
}
} --- memory test ---
}
01A4' 06 00           ;   select: ld   b,0     ; clear b patrn modifier
}

```

```

|---- load up memory ----
|
01A6' 48      loop: ld  c,b          ; c = data to disp on screen
01A7' 11 39F  ld  de,vpnt+12*32+15 ; pattern modifier disp
01AA' D0 21 01B1' ld  ix,8+7
01AE' C3 037E'  jp  wdata          ; disp sub
|
01B1' 21 B000  ld  hl,start       ; get starting addr
01B4' 7D      fill: ld  a,l          ; low byte to accm
01B5' AC      xor  h          ; xor with high byte
01B6' AB      xor  b          ; xor with pattern
|
01B7' 77      ld  (hl),a       ; store in test addr
|
01B8' 23      inc  hl          ; increment addr
01B9' 7C      ld  a,h          ; load high byte of addr
01BA' FE 00   cp  epage        ; compare with stop addr
01BC' C2 01B4' jp  nz,fill       ; not done, go back
|
|---- delay the refresh ----
|
01BF' 2E 10   ld  l,10h         ; outer loop
|
01C1' AF      leak: xor  a          ; a = 00 to 128
01C2' ED 4F   refht: ld  r,a         ; a-> refresh address
01C4' ED 4F   ld  r,a         ; delay refresh addr
01C6' ED 4F   ld  r,a         ; it makes the end refresh addr wait longer
01C8' 3C      inc  a          ; a = a + 1
01C9' FE 80   cp  128         ; is a = end of one refresh cycle ?
01CB' 20 F5   jr  nz,refh       ; loop back
01CD' 2D      dec  l          ; l reg as counter of outer loop
01CE' 20 F1   jr  nz,leak
|
|---- cap lamp on ----
|
|   in  a,(rdpsg) ; psg latching at reg. 15
|   or  00100000b ; turn on cap lamp bit
|   out (wrpsg),a
|
|---- cap lamp on (for rx) ----
|
01D0' D8 A1   in  a,(ppirdc) ; read ppi port c
01D2' E6 DF   and  10111111b ; force cap lamp on
01D4' D3 A1   out (ppipoc),a
|
|---- read and check test data ----
|
01D6' 21 B000  ld  hl,start       ; get starting addr
01D9' 7D      tests: ld  a,l          ; load low byte
01DA' AC      xor  h          ; xor with high byte
01DB' AB      xor  b          ; xor with modifier
01DC' 4E      ld  c,(hl)
01DD' B9      cp  c          ; compare with memory loc
01DE' C2 0343' jp  nz,error       ; error exit
01E1' 23      inc  hl          ; update memory address
01E2' 7C      ld  a,h          ; load high byte
01E3' FE 00   cp  epage        ; compare with stop addr
01E5' C2 01D9' jp  nz,test       ; loop back
|
|---- cap lamp off ----
|
|   in  a,(rdpsg) ; psg latching at reg. 15
|   and  11011111b ; turn off cap lamp
|   out (wrpsg),a
|
|---- cap lamp off (for rx) ----
|
01E8' D8 A1   in  a,(ppirdc) ; read ppi port c
01EA' F6 40   or  01000000b ; force cap lamp on
01EC' D3 A1   out (ppipoc),a
|
01EE' 78      ld  a,b          ; check for B
01EF' B7      or  a
01F0' 20 04   jr  nz,nzfill ; if B not 0, to shift 1
|
01F2' 04      inc  b          ; B=0 to make "1"
01F3' C2 01A6' jp  nz,loop       ; rst with new modifier
|
01F6' CB 20   nzfill: sla  b          ; to shift (cy) (- /...B.../ (- 0

```

```

01FB' C2 01A6'          jp    nz,loop    ;
;
;----- disp memory test ok -----
;
01FB' 11 3806          ld    de,vpnt+6*32+22 ;
01FE' 21 0512'          ld    hl,msg4b    ; 'loc: 0000-ffff o.k.'
0201' D0 21 0208'          ld    ix,$+7
0205' C3 0382'          jp    msg    ; disp message
;
;===== switch to bk 21 and test =====
;
0208' 11 3907          ld    de,vpnt+8*32+7
020B' 21 04FB'          ld    hl,msg4a    ; 'loc: 0000-7fff testing'
020E' D0 21 0215'          ld    ix,$+7
0212' C3 0382'          jp    msg
;
;----- transfer test module -----
;
0215' 21 0900*          ld    hl,ts328    ; test module to test 0000-7fff
0218' 11 C000          ld    de,trans    ; transfer loc
021B' 01 0200          ld    bc,tsize    ; size of module
021E' E0 D0          ldir
;
;
; ld    a,11111101b    ; bk 21 on, game off
;
0220' 0B A8          in    a,(ppipoa)    ; read slot assignment (for msg)
0222' 32 C200          ld    (oldsl),a    ; save it
;
;
; ld    iy,$+7          ; return addr
; jp    trans          ; go to test module in ram
;
;
; ld    hl,msg4b    ; 'o.k.'
; ld    de,vpnt+8*32+22 ; disp message
; ld    ix,$+7
; jp    msg    ; disp message
;
;
; EXX
; LD A,E
; LD (FLN),A
; JP EXIT
;
;-----
; test ppi read/write
;-----
;
;
;
; ld    a,ppicnd    ; init ppi
;
0241' 3E B2          ld    a,ppicnd    ; init ppi
;
; out    (ppicnp),a    ; port a,b as input, port c as output
;
0243' D3 A8          out    (ppicnp),a    ; port a,b as input, port c as output
;
; ld    b,15          ; access b=15 to 0 to port c
;
0245' 06 0F          ld    b,15    ; access b=15 to 0 to port c
0247' 7B          ld    a,b
;
; ppichk: ld    a,b
; out    (ppipoc),a    ; data -> port c
; ncp
; ncp
; ncp
; in    a,(ppirdc)    ; read a back & check
; cp    b              ; is the same ?
; jp    nz,perror    ; ppi error
; djnz  ppichk        ; b=b-1, loop back
;
;
;----- ppi ok -----
;
; ld    de,vpnt+15*32+7
; ld    hl,msg6    ; 'ppi ok'
; ld    ix,$+7
; jp    msg    ; disp message on screen
;
;----- delay for reading a while -----
;
; ld    h,3          ; delay multiplier
;
0262' 26 03          ld    h,3    ; delay multiplier
0264' 01 FFFF          ld    bc,0ffffh
0267' 00          ld    bc,0ffffh
;
; delay1: ncp
; delay1: ncp
; delay1: ncp
;
; dec    bc
; ld    a,b          ; delay
; or    c            ; check end
;
026A' 0B          dec    bc
026B' 7B          ld    a,b    ; delay
026C' B1          or    c    ; check end
;
; jr    nz,delay
; dec    h
;
026D' 20 F8          jr    nz,delay
026F' 23          dec    h
;
0270' 20 F2          jr    nz,delay1    ; delay multiplier

```

```

=====
|
|-----
|
| error entry
|-----
|
0272' D9          verror: exx          ; vram error save error message
|
0273' 01 4000    ld      bc,4000h      ; clear 16 k vram
0276' 11 0000    ld      de,vpgt
0279' D0 21 0280' ld      ix,#+7
0270' C3 03C8'   jp      cvmbik
|
0280' 21 0000*   ld      hl,ascii      ; set ascii patterns
0283' 11 0100    ld      de,vpgt+20h*8
0286' 01 0300    ld      bc,300h
0289' D0 21 0290' ld      ix,#+7
0280' C3 03E0'   jp      wmbik
|
0290' 21 0000*   ld      hl,ascii      ; set ascii patterns
0293' 11 0900    ld      de,vpgt+20h*8+2048
0296' 01 0300    ld      bc,300h
0299' D0 21 02A0' ld      ix,#+7
0290' C3 03E0'   jp      wmbik
|
02A0' 21 0000*   ld      hl,ascii      ; set ascii patterns
02A3' 11 1100    ld      de,vpgt+20h*8+2048*2
02A6' 01 0300    ld      bc,300h
02A9' D0 21 02D0' ld      ix,#+7
02A0' C3 03E0'   jp      wmbik
|
02B0' 3E F1      ld      a,0fh         ; text char color
02B2' 11 2000    ld      de,vpct       ; pattern color tbl
02B5' 01 1800    ld      bc,256*8*3    ; 3 portions
02B8' D0 21 02B0' ld      ix,#+7
02B0' C3 03C7'   jp      svmb        ;
|
|----- error message heading -----
|
|.... CHECK BEFORE DISP EITHER 318 OR 328
|
02B0' 21 045B'   ld      hl,msg1k      ; ' functional test error'
02C2' 11 3847    ld      de,vpnt+2*32+7
02C5' D0 21 02CC' ld      ix,#+7
02C9' C3 03B2'   jp      dmsg
|
02CC' 21 0481'   ld      hl,msg2a      ; ' video ram '
02CF' 11 3907    ld      de,vpnt+8*32+7
02D2' D0 21 02D9' ld      ix,#+7
02D6' C3 03B2'   jp      dmsg
|
02D9' 21 0488'   ld      hl,msg2b      ; ' error pat rd wr addr '
02DC' 11 3947    ld      de,vpnt+10*32+7
02DF' D0 21 02E6' ld      ix,#+7
02E3' C3 03B2'   jp      dmsg
|
02E6' 21 0537'   ld      hl,msg8       ; 'pw- up pass:'
02E9' 11 3AA7    ld      de,vpnt+21*32+7
02EC' D0 21 02F3' ld      ix,#+7
02F0' C3 03B2'   jp      dmsg
|
02F3' 21 0000    ld      hl,0000      ; disp # of passes
02F6' 39        add     hl,sp
02F7' 4C        ld      c,h          ; high byte
02F8' 11 3A84    ld      de,vpnt+21*32+20
02FB' D0 21 0302' ld      ix,#+7
02FF' C3 037E'   jp      wdata
|
0302' 40        ld      c,l          ; low byte
0303' 11 3A86    ld      de,vpnt+21*32+22
0306' D0 21 0300' ld      ix,#+7
030A' C3 037E'   jp      wdata
|
|----- disp error data -----
|
0300' D9        exx          ; restore error data
030E' EB        ex      de,hl      ; in vram de->hl as err addr
030F' 0B        ex      AF,AF'     ; temp save wr data

```

```

0310' 4B          ld  c,b          ; disp pattern
0311' 11 39FF     ld  de,vpnt+12*32+15
0314' D0 21 031B' ld  ix,$+7
031B' C3 037E'   jp  wdata          ; disp reg. c
031B' 0B

031C' C3 0343'   ;
;
;
031F' 4F         perror: ld  c,a
0320' 11 39F3     ld  de,vpnt+15*32+19      ; disp rd data
0323' D0 21 032A' ld  ix,$+7
0327' C3 037E'   jp  wdata

032A' 4B         ;
032B' 11 39F6     ld  c,b          ; disp wr data
032E' D0 21 0335' ld  de,vpnt+15*32+22
0332' C3 037E'   jp  wdata

0335' 21 052B'   ;
033B' 11 39E7     ld  hl,msg6a         ; 'ppi error !'
033B' D0 21 0342' ld  de,vpnt+15*32+7
033F' C3 03B2'   jp  dmsg

0342' 76         ;
;
;
;----- error of system ram -----
;
0343' 0B         errors: EI  AF,AF'      ; temp save right data
0344' 11 3993     ld  de,vpnt+12*32+19 ; error pattern disp reg. c
0347' D0 21 034E' ld  ix,$+7
034B' C3 037E'   jp  wdata

034E' 0B         ;
034F' 4F         EX  AF,AF'      ; a= right addr
0350' 11 3996     ld  c,a
0353' D0 21 035A' ld  de,vpnt+12*32+22 ; right pattern disp
0357' C3 037E'   jp  wdata

035A' 4C         ;
035B' 11 3999     ld  c,h          ; disp addr (ll)
035E' D0 21 0365' ld  de,vpnt+12*32+25 ;
0362' C3 037E'   jp  wdata

0365' 4D         ;
0366' 11 399B     ld  c,l          ; disp addr (l.)
0369' D0 21 0370' ld  de,vpnt+12*32+27
036D' C3 037E'   jp  wdata

0370' 21 051A'   ;
0373' 11 3987     ld  hl,msg5         ; 'error !'
0376' D0 21 037D' ld  de,vpnt+12*32+7
037A' C3 03B2'   jp  dmsg

037D' 76         ;
;
;
;----- flag operator -----
;
;----- write a byte on screen -----
;
; inputs: de -> vpnt + row * 32 + col
;         c = data to display
; destroy: af
;
;-----
;
037E' 7B         wdata: ld  a,e          ; set screen write
037F' D3 99       out  (svaddr),a
0381' 7A         ld  a,d
0382' E6 3F       and  3fh
0384' F6 40       or   40h
0386' D3 99       out  (svaddr),a

;
;----- display write data -----
;
038E' 79         ;
038F' C0 3F       ld  a,c          ; write data
038B' C0 3F       srl  a
038B' C0 3F       srl  a

```



03C8'	AF
03C9'	08
03CA'	7B
03CB'	D3 99
03CC'	7A
03CD'	E6 3F
03CE'	F6 40
03CF'	D3 99
03D4'	08
03D5'	5F
03D6'	D3 98
03D9'	08
03DA'	7B
03DB'	B1
03DB'	7B
03DC'	20 FB
03DE'	D0 E9

```

;
; WMBLK: XOR A ; A=00 (FOR VRAM CLEAR)
;
;
; SVB: EX AF,AF'
;       ld a,e ; lower byte addr of vram loc
;       out (svaddr),a ; set a7....a0 of vram
;       ld a,d ; higher byte addr of vram loc
;       and 3fh ; a = 00xx xxxx
;       or 40h ; a = 01xx xxxx (2nd byte wr vram format)
;       out (svaddr),a ; set up a13.....a8 for vram write
;       EX AF,AF' ;
;
;
; LD E,A ; E=CHVR
; CSLOOP: OUT (MVDATA),A ; 00->VRAM (BUS NEED EACH WRITE)
;          DEC BC ;
;          LD A,B ; DO UNTIL BC=0
;          OR C ;
;          LD A,E ; A=CHVR TO STORE
;          JR NZ,CSLOOP
;          JP (ix) ; return addr
;
;-----
; WMBLK: WRITE VRAM BLOCK FROM CPU MEMORY
;
; INPUT: HL = SOURCE ADDR OF CPU MEMORY
;        DE = DEST ADDR OF VDP VRAM
;        BC = BYTE COUNT
;
; OUTPUT: -
;
; CALLS: WMBSET (WRITE VRAM ADDR SET-UP)
;
; DESTROY: AF,EC, DE, HL
;
;-----
;
; WMBLK: ld a,e
;        out (svaddr),a ; SETUP STARTING ADDR OF VRAM
;        ld a,d ; DE=VRAM ADDR
;        and 3fh
;        or 40h
;        out (svaddr),a
;
;
; WMBLK: LD A,(HL) ; A = SOURCE DATA FROM CPU MEMORY
;        OUT (MVDATA),A ; DATA ->(VRAM)
;        INC HL ; UPDATE SOURCE PTR
;        DEC BC ; UPDATE COUNTER
;        LD A,C
;        OR B ; DO UNTIL BC=0
;        JR NZ,WMBLK1
;        JP (ix)
;
;-----
; WDPSET: SET UP 8 VDP REGISTERS (0-7)
;
; INPUT: HL -> R0 DATA FOR VDP
;        ;
;        ;
;        ; R7 DATA FOR VDP
;
; OUTPUT: HL = HL+8
;
; CALLS: WDPFRG (MOVE 1 BYTE TO VDP REGISTER)
;
; DESTROY: af-hl
;
;-----
;
; WDPSET: LD BC,0800H ; B = 08 COUNTER, C=00 VDP REG # REL
;
;
; vdpsti: LD A,(HL) ; FETCH A = VDP DATA
;        out (svdprg),a ; data wr to vdp
;        ld a,c ; reg #
;        and 07h ; a = 0000 0xxx
;        or B0h ; a = 1000 0xxx (wr vdp reg format)
;        out (svdprg),a ; reg select
;
;
; INC C ; UPDATE REG# PTR OF VDP
; INC HL ; UPDATE DATA PTR AT CPU MEMORY
; DJNZ vdpsti ; DO UNTIL B=0
;

```

03E0'	7B
03E1'	D3 99
03E3'	7A
03E4'	E6 3F
03E6'	F6 40
03EB'	D3 99
03EA'	7E
03EB'	D3 98
03ED'	23
03EE'	08
03EF'	79
03F0'	B0
03F1'	20 F7
03F3'	D0 E9

03F5'	01 0800
03F8'	7E
03F9'	D3 99
03FB'	79
03FC'	E6 07
03FE'	F6 B0
0400'	D3 99
0402'	0C
0403'	23
0404'	10 F2

```

0406' DD E9          jp      (ix)          ; return
;-----
; WMBYT: WRITE 1 BYTE INTO VOP RVM
;-----
; INPUT: DE -> VRAM LOC
;        A = DATA INTO VRAM
; OUTPUT: (DE) <- A, WHERE DE->VRAM LOC
; CALLS: WMBSET (WRITE VRAM ADDR SET-UP)
; DESTROY: af-hl
;-----
;
040B' 0B
0409' 7D             ld      a,e
040A' D3 99         out      (svaddr),a      ; set a7....a0
040C' 7A             ld      a,d
040D' E6 3F         and     3fh
040F' F6 40         or      40h              ; a = 01xx xxxx
0411' D3 99         out      (svaddr),a      ; set a13....a8
0413' 0B
;-----
;
0414' D3 98         out      (WADATA),A      ; TO WRITE A BYTE
0416' DD E9          jp      (ix)          ; return
;-----
; wvnsset: vram write addr set
;-----
; Input: de ->vram addr, ix=ret addr
; DESTROY: af
;-----
;
041B' 7B
0419' D3 99         ld      a,e
041A' 7A             ld      a,d
041C' E6 3F         and     3fh
041E' F6 40         or      40h              ; a=01xx xxxx
0420' D3 99         out      (svaddr),a      ; a13....a0
0422' DD E9          jp      (ix)
;-----
;
; rvnsset: vram read addr set
;-----
; Input: de->vram addr, ix=ret addr
; DESTROY: af
;-----
;
0424' 7B
0425' D3 99         ld      a,e
0427' E6 3F         and     3fh              ; a=00xx xxxx
0429' D3 99         out      (svaddr),a      ; set a13....a8
042B' DD E9          jp      (ix)
;-----
;----- message -----
;
042D' 4D 53 58 20   msg1:  db      'MS1 64K AT SLOT 2 TEST',00
;-----
;
0431' 36 34 4B 20
0435' 41 54 20 53
0439' 4C 4F 54 20
043D' 32 20 54 45
0441' 53 54 00
0444' 4D 53 58 20   msg1x: db      'MS1 16K AT SLOT 2 TEST',00
0448' 31 36 4B 20
044C' 41 54 20 53
0450' 4C 4F 54 20
0454' 32 20 54 45
0458' 53 54 00
045B' 46 55 4E 43   msg1k: db      'FUNCTIONAL TEST ERROR',00
045F' 54 4F 4F 4E
0463' 41 4C 20 54
0467' 45 53 54 20
046B' 45 52 52 4F
046F' 52 00
0471' 31 36 4B 20   msg2:  db      '16K VRAM O.K. !',00
0475' 56 52 41 4D
0479' 20 4F 2E 4B
047D' 2E 20 21 00
0481' 56 49 44 45   msg2a: db      'VIDEO ',00
0485' 4F 20 00
048B' 56 52 41 4D   msg2b: db      'VRAM PAT RD WR ADDR',00

```



```

048C' 20 20 20 20
0490' 50 41 54 20
0494' 52 44 20 57
0498' 52 20 41 44
049C' 44 52 00
049F' 33 32 48 20
04A3' 52 41 40 3A
04A7' 50 41 54 20
04AB' 52 44 20 57
04AF' 52 20 41 44
04B3' 44 52 00
04B6' 31 36 48 20
04BA' 52 41 40 3A
04BE' 50 41 54 20
04C2' 52 44 20 57
04C6' 52 20 41 44
04CA' 44 52 00

04CD' 4C 4F 43 3A
04D1' 20 38 30 30
04D5' 30 20 46 46
04D9' 46 46 20 54
04DD' 45 53 54 49
04E1' 4E 47 00
04E4' 4C 4F 43 3A
04E8' 20 43 30 30
04EC' 30 20 46 46
04F0' 46 46 20 54
04F4' 45 53 54 49
04F8' 4E 47 00

04FB' 4C 4F 43 3A
04FF' 20 30 30 30
0503' 30 20 37 46
0507' 46 46 20 54
050B' 45 53 54 49
050F' 4E 47 00
0512' 4F 2E 48 2E
0516' 20 20 20 00
051A' 45 52 52 4F
051E' 52 20 21 00
0522' 50 50 49 20
0526' 4F 2E 48 2E
052A' 00
052B' 50 50 49 20
052F' 45 52 52 4F
0533' 52 20 21 00
0537' 50 57 52 20
053B' 55 50 20 50
053F' 41 53 53 3A
0543' 00

```

```

msg3r db '32K RMM:PAT RD WR ADDR',00
msg3x1 db '16K RMM:PAT RD WR ADDR',00
msg4i db 'LOC: 8000-FFFF TESTING',00
msg4x1 db 'LOC: C000-FFFF TESTING',00
msg4at db 'LOC: 0000-7FFF TESTING',00
msg4b1 db 'O.K. ',00
msg5r db 'ERROR !',00
msg6i db 'PPI O.K.',00
msg6at db 'PPI ERROR !',00
msg8i db 'PWR UP PASS!',00

```

```

1
END

```

Macros:

Symbols:

02A1# ASCII	0001' BEGIN	0000' CKVRAM
03D6' CSLDUP	03D8' CMWBLK	0267' DELAY
0264' DELAY1	03B21' DMSG	03DC' DMSG1
03C6' DMSGX	0377' EI	030B' E2
0000 EFNDE	0343' ERROR	0168' EXIT
01E4' FILL	010E' FILLX	023C# FUN
01C1' LEAK	011B' LEAKX	01A6' LOOP
0100' LOOPY	042D' MERR	0450' MSG1K
0444' MSG1X	0471' MSG2	0481' MSG2A
048B' MSG2B	049F' MSG3	04D6' MSG3X
04CD' MSG4	04F8' MSG4A	0512' MSG4B
04E4' MSG4X	051A' MSG5	0522' MSG6
052B' MSG6A	0537' MSG6B	01F6' NZFILL
0151' NZFILLX	0054' NZVARI	C200I OLSLT
016E# OPTION	031F' PERRNR	0247' PPICHK
00B2 PPICMD	000B PPICOMP	000B PPIFDB
00A9 PPIFDB	000A PPIFDC	000A PPIFDC
0000 PSOLAT	0171# PSQL_TEST	0002 RDRSG
01C2' REFH	011C' REFH	0000# ROM
007B RMMATA	0474' RMMXRT	0099 RVSTAT
01A4' SELECT	00611' SRAM_TEST	8000 START
0099 SMDUR	0099 SMMRIG	03C9' SMMB
00CD' SYS31B	0173' SYS32B	01B9' TEST
0133' TESTX	C000 TTRNG	0216# TS32B
0200 TSIZE	0036' VDRCHK	019C' VDRFEG
03F5' VDRSET	03FB' VDRSH	0040 VDFAGE
0272' VERRDR	0017' VFILL	2000 VFCT
0000 VFGT	3000 VINT	0002 VRO
00C2' VRI	000E VR2	00FF VR3
0003 VR4	0076 VR5	0003 VR6
0000 VR7	00001' VRAM_TEST	3B00 VSAT
1B00 VSFGT	0000 VSTART	037E' WDATA
00A1 WFRSB	009B WVDATA	03EA' WMRK1
03E0' WMBLK	040B' WMBYT	041B' WMRSET

No Fatal error(s)





```

0114' 4A 4F 59 53      TEXT1: DB 'JOYSTICK TEST',00
0118' 54 49 43 4B
011C' 20 54 45 53
0120' 54 00
0122' 4A 4F 59 53      TEXT2: DB 'JOYSTICK TEST OK ',00
0126' 54 49 43 4B
012A' 20 54 45 53
012E' 54 20 20 4F
0132' 4B 20 00
0135' 4A 4F 59 53      TEXT3: DB 'JOYSTICK 1 ERROR',00
0139' 54 49 43 4B
013D' 20 20 20 31
0141' 20 20 20 45
0145' 52 52 4F 52
0149' 00
014A' 4A 4F 59 53      TEXT4: DB 'JOYSTICK 2 ERROR',00
014E' 54 49 43 4B
0152' 20 20 20 32
0156' 20 20 20 45
015A' 52 52 4F 52
015E' 00

```

```

          |
          |                               0091
          |                               0090
          |                               0090
          |                               000A
          |                               00A9
          |                               000A
          |
          | FINISH#
          |                               0000'
          |                               0000' 01 4000
          |                               0003' 11 0000#
          |                               0006' 0D 0000#
          |
          |                               0009' 21 0000#
          |                               000C' 11 0100#
          |                               000F' 01 0300
          |                               0012' 0D 0000#
          |
          |                               0015' 21 0000#
          |                               0018' 11 0900#
          |                               001B' 01 0300
          |                               001E' 0D 0000#
          |
          |                               0021' 21 0000#
          |                               0024' 11 1100#
          |                               0027' 01 0300
          |                               002A' 0D 0000#
          |
          |                               002D' 3E F1
          |                               002F' 11 0000#
          |                               0032' 01 1800
          |                               0035' 0D 0000#
          |
          |                               0038' 21 0119'

```

Macros:

```

Symbol#          000B' CHK_OIE      0080' CHK_TRL
000D# CVMBLK      00B4' DATA_IN      00C8' DATA_OUT
000B' DATA_OUT1  00DB1' DELAY1      00DB' DELAY2
00E1' DELAY3      010C# DMSG         0007 ENABLE
007D' EXIT        015F' FINISH      00EF FLAG1
000F FLAG2       007E# FLN          00AB' IN_TBL1
00AC' IN_TBL2    00EC' JOY_ERR      0104' JOY_ERR1
00001' JUY_TEST  00C3# OPTION      00B6# PRINT_TEST
000E PSG_A       000F PSG_B        00A0 PSG_LAT
00A2 PSG_RD      00A1 PSG_WR       00BF SELECT1
000F SELECT2     0000' START       003C# SMBOLK
0114' TEXT1      0122' TEXT2      0135' TEXT3
014A' TEXT4      0001# VDFREB     0004# VDFSET
2000 VCT         0000 VFGT        2800 VFNT
000E VR2        00FF VR3         0003 VR4
0031# WMBLK

```

No Fatal error(s)

```

.Z80
|
| COMMENT %
|
| .....,X
|
| PRINTER TESTING
|
| DATE --- AUG 11, 1984
|
| BY W.K. LEUNG
|
| .....,X
|
| EXT CVMBLK,WMBLK,SMBOLK,ASC11,DMSG
| EXT FUN,OPTION,VFGT,VFNT,VFCT,CAS_TEST
| ENTRY PRINT_TEST
|
| << PORT ADDRESS >>
|
| P_DATA EQU 91H        | DATA PORT
| P_STRB EQU 90H        | STROBE PORT
| P_STAT EQU 90H        | STATUS PORT
|
| FP1CH EQU 0A9H       | FPI PORT C WRITE
| FP1CR EQU 0A9H       | FPI PORT B READ
| FP1CR EQU 0A9H       | FPI PORT C READ
|
| << PRINTER TESTING >>
|
| PRINT_TEST:
| LD DC,4000H          | CLEAR 16K RAM
| LD DE,VFGT
| CALL CVMBLK
|
| LD HL,ASC11
| LD DE,VFGT+201H+B
| LD DC,3001
| CALL WMBLK
|
| LD HL,ASC11
| LD DE,VFGT+2040+2+201H+B
| LD DC,3001
| CALL WMBLK
|
| LD HL,ASC11
| LD DE,VFGT+2040+2+201H+B
| LD DC,3001
| CALL WMBLK
|
| LD A,C011H
| LD DE,VFGT
| LD DC,256H*B+3
| CALL SMBOLK
|
| LD HL,TEXT1

```

```

003B' 11 00B7#          LD DE,VPNT+4#32+7
003E' D0 21 0045#      LD IX,#+7
0042' C3 0000#          JP DMSG

0045' 21 0126#          LD HL,TEXT2
004B' 11 0107#          LD DE,VPNT+8#32+7
004B' D0 21 0052#      LD IX,#+7
004F' C3 0000#          JP DMSG

0052' 21 013E#          LD HL,TEXT3
0055' 11 0144#          LD DE,VPNT+04#32+4
005B' D0 21 005F#      LD IX,#+7
005C' C3 0000#          JP DMSG

005F' 21 014B#          LD HL,TEXT6
0062' 11 01AA#          LD DE,VPNT+00H#32+10
0065' D0 21 006C#      LD IX,#+7
0069' C3 0000#          JP DMSG

006C' 21 014E#          LD HL,TEXT4
006F' 11 02C7#          LD DE,VPNT+16H#32+7
0072' D0 21 0079#      LD IX,#+7
0076' C3 0000#          JP DMSG

0079' 21 0119#          LD HL,TEXT1
007C' C0 00C3#          CALL P_STRING
007F' C0 00CD#          CALL P_LFOR ; PRINT LINE FEED AND RETURN

0082' 21 0126#          LD HL,TEXT2
0085' C0 00C3#          CALL P_STRING
0088' C0 00CD#          CALL P_LFOR

008B' 21 013E#          LD HL,TEXT3
008E' C0 00C3#          CALL P_STRING
0091' C0 00CD#          CALL P_LFOR

0094' 21 014B#          LD HL,TEXT6
0097' C0 00C3#          CALL P_STRING
009A' C0 00CD#          CALL P_LFOR

009D' 21 0170#          LD HL,TEXT7
00A0' 11 0247#          LD DE,VPNT+12H#32+7
00A3' D0 21 00AA#      LD IX,#+7
00A7' C3 0000#          JP DMSG

;
; << GET REPLY >>
;
KEY_IN:
00AA' DB AA
00AB' DB AA
00AC' E6 F0
00AE' 4F

;
; KEY_IN1:
00AF' 79
00B0' D3 AA
00B2' DB A9

00B4' FE FE ; ZERO PRESS?
00B6' 20 F2
00B8' 3A 0000#
00B9' FE 09
00BD' C2 0000#
00CD' C3 0000#

;
; << PRINT STRING >>
;
P_STRING:
00C3' LD A,(HL)
00C4' LD E,A
00C5' OR A
00C6' DB ; END OF STRING
00C7' C0 00D7# CALL P_CWR ; PRINT A CHR
00CA' 23
00CB' 1B F6

;
; << PRINT LINE FEED AND RETURN >>
;
00CD' LD DE,VPNT+4#32+7
00CD' 21 00D4#
00DD' C0 00C3#
00D3' C9

00D4' 00 0A 00

00D7' 01 FFFF
00DA' DB 90
00DC' E6 02
00DE' 2B 0A
00E0' 00
00E1' 79
00E2' 87
00E3' 20 F5
00E5' 10 F3
00E7' C3 0109#

00EA' 7B
00EB' D3 91
00ED' AF
00EE' D3 90
00F0' 3C
00F1' D3 90
00F3' C9

00F4' 11 FFFF
00F7' C5
00FB' 01 FFFF
00FB' 00
00FC' 79
00FD' 87
00FE' 20 FB
0100' 10 F9
0102' C1
0103' 1B
0104' 7A
0105' B3
0106' 20 EF
0108' C9

0109' 21 0160#
010C' 11 0247#
010F' D0 21 0116#
0113' C3 0000#
0116' C3 0000#

0119' 50 52 49 4E
011D' 5A 45 52 20
0121' 5A 45 53 5A
0125' 00
0126' 44 41 5A 41
012A' 20 44 49 53
012E' 50 4C 41 59
0132' 20 4F 4E 20
0136' 50 52 49 4E
013A' 5A 45 52 00
013E' 40 55 53 54
0142' 20 42 45 20
0146' 3A 00
014B' 50 2C 5A 2C
014C' 5F 00
014E' 50 52 45 53
0152' 53 20 20 30

;
; P_LFOR:
LD HL,LFOR
CALL P_STRING
RET

;
; LFOR:
DB 00H,00H,00H

;
; << PRINT A CHR >>
;
P_CWR:
LD IC,OFFFH
P_CWR1:
IN A,(P_STAT)
AND 0000100 ; CHECK FOR PRINTER READY
JR Z,DATA_OUT
IX: C
LD A,C
OR A
JR NZ,P_CWR1
DJNZ P_CWR1
JP P_ERR ; PRINT ERROR

;
; << DATA OUTPUT >>
;
DATA_OUT:
LD A,E ; GET DATA
OUT (P_DATA),A ; SEND TO CHIP
XOR A
OUT (P_STRB),A ; CLEAR DATA STROBE
INC A
OUT (P_STRD),A ; SET DATA STROBE
RET

;
; << DELAY >>
;
DELAY:
LD DE,(FFFF)
DELAY2: RPT# DC
LD IC,(FFFF)
DELAY1: DEC C
LD A,C
OR A
JR NZ,DELAY1
DJNZ DELAY1
RPT DC
DEC DE
LD A,D
OR E
JR NZ,DELAY2
RET

;
; << ERROR HANDLING ROUTINE >>
;
P_ERR:
LD HL,TEXT5
LD DE,VPNT+12H#32+7
LD IX,#+7
JP DMSG
JP KEY_IN

TEXT1: DB 'PRINTER TEST',00
TEXT2: DB 'DATA DISPLAY ON PRINTER',00
TEXT3: DB 'MUST BE 1',00
TEXT6: DB 05H, ',', 05AH, ',', 05FH,00
TEXT4: DB 'PRESS 0 TO EXIT',00

```

```

0156' 20 20 54 4F
015A' 20 45 5B 49
015E' 5A 00
0160' 4E 4F 20 52      TEXT5: DB 'NO READY SIGNAL',00
0164' 45 41 44 5F
0168' 20 53 49 47
016C' 4E 41 4C 00
0170' 50 52 49 4E      TEXT7: DB 'PRINTER OK',00
0174' 5A 45 52 20
0178' 4F 4B 00

```

```

017B'
      |
      | FINISH
      |
      | END

```

Macros:

Symbols:

0022# ASCII	00C1# CAS_TEST	0007# CMBLK
00EA# DATA_OUT	00F4# DELAY	00FB# DELAY1
00F7# DILAY2	0114# DMSS	017B# FINISH
00B9# FIN	00AA# KEY_IN	00AF# KEY_IN1
0004# L1DR	00BE# OPTION	00A9# PPRR
00VA# P1DR	00VA# P1CW	0001# PRINT_TEST
0007# P_CWR	00DA# P_CWR1	0091# P_DATA
0109# P_FRR	00CD# P_L1DR	0090# P_STAT
0090# P_STRB	00C3# P_STRING	0036# SWBLK
0119# TEXT1	0126# TEXT2	013E# TEXT3
014E# TEXT4	0160# TEXT5	0148# TEXT6
0170# TEXT7	0030# VECT	0025# VPBT
0100# VNT	002B# WMELK	

No Fatal error(s)

THE SVI-700 SERIES PERIPHERALS

FOR SVI-728

(1st Release)

Kimba Lau

July, 1985

CONTENTS:

- (1) SVI-707 MSX 5.25" DISK DRIVE
- (2) SVI-727 MSX 80 COLUMN CARTRIDGE
- (3) SVI-737 MODEM WITH RS232C INTERFACE CARTRIDGE
- (4) SVI-747 MSX 64 RAM CARTRIDGE
- (5) SVI-757 MSX RS-232C INTERFACE CARTRIDGE
- (6) SVI-767 MSX DATA CASSETTE



(1) SVI-707 MSX 5.25" DISK DRIVE

SVI-707 MSX Disk Drive is your gateway into the expanding universe of MSX computer software and programming tools. It is very useful as an external memory unit because of its large memory capacity and high access speed.

The 5.25" double-sided, double density disk drive provides 320K formatted memory capacity to utilize disk BASIC, most current CP/M, MSX DOS. The built-in disk drive controller allows it to hook up directly to the computer.

With a 13K spooler buffer when running in CP/M mode, it can read/write Kaypro II, Osborne I, Bondwell 12/14, and all single or double-sided disks formatted for the SVI-328.

1.1 SPECIFICATION

Drive Unit	:	single
Floppy Diskette	:	- Double sided - Double density - Soft sectored 5.25" diskette
Memory Capacity		
- Unformatted	:	500K bytes
- Formatted	:	326K bytes (CP/M) 355K bytes (MSX DOS)
Disk Operating System	:	- MSX DOS - CP/M DOS
Rotational Speed	:	300 rpm

Disk Format : CP/M  
 - 40 tracks / side  
 - 17 sectors / track  
 - 256 bytes / sector

MSX DOS  
 - 40 tracks / side  
 - 9 sectors / track  
 - 512 bytes / sector

Access Time : - Track - to - track 26 msec  
 - Setting time 20 msec  
 - Motor start time 350 msec

Recording Density : 5536 bpi

Track Density : 48 tpi

Encoding Method : MFM

Power source : 110V / 220V  
 (separate power supply)

Humidity : 20% - 80%

Dimensions : 162(W) x 240(D) x 63(H)

For trouble shooting, please refer to Service & Technical Manual for Disk Drive of SVI-707 - MSX Disk Drive.

1.2 SPARE PART LISTGENERAL RECOMMENDED SPARE PART LIST FOR SVI-707  
5 1/4" MSX DISK DRIVE

<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
PI31807	SVI WARRANTY	0.10
MM00007	MOISTURE ABSORBANT (10GM)	0.10
MM00020	DOUBLE SIDE DISKETTE	1.30
WP70705	CP/M 2.24 UTILITY LABEL	0.10
MM00020	DOUBLE SIDE DISKETTE	1.30
WP70706	SVI BLANK DISKETTE LABEL	0.10
PC707101-01	OUT/C 445X715X270MM	0.30
PC707102	SVI-707 SPACER F CARTON	0.10
PC707103	SPACER F MANUAL BOX	0.20
PC707104	SVI-707 MANUAL BOX	0.30
PC707101	G/B F MSX DISK DRIVE	0.50
PI609102	MSX DOS USER'S MANUAL	0.90
PI609103	MSX DISK BASIC MANUAL	0.50
PI707101-01	707 USER'S MANUAL	0.40
PI707201	ADDENDUM TO 707 MANUAL	0.10
PIBW1202	CP/M OPERATING SYSTEM	1.70
PIBW1203	CP/M COMMAND SUMMARY	0.50
PP70701	POLYFOAM F MSX DISK DRIVE	0.90
PP70702	POLYFOAM SHEET F 707	0.20
PZ01001	PLASTIC BAG 9.5X16.5 IN	0.10
PZ60503	PLASTIC BAG 445X305MM	0.10
PZ80102	PLASTIC BAG 153X178MM	0.10
WP70701	FCC LABEL	0.10
MJ016	DC CORD FOR 700 TX	0.50
WP70704	S/NO LABEL 707	0.10
WP70704G	S/NO LABEL F G/B	0.10
DL010	LED 2 X 5 (MR62D)	0.10
IG7805	REGULATOR UA7805	0.50
IG7812	REGULATOR UA7812	0.50
IM273235	EPROM 2732 350NS	2.60
IM276430	EPROM 2764 300NS	2.10
KE4003325	E.CAP 3300UF 25V	0.40
KE4004716	E.CAP 4700UF 16V	0.40
KR1068050	CERAMIC CAP 68PF 50V	0.10
MC80101	CRYSTAL 8MHZ	0.40
MJ114	UL1007 #24 BROWN 180MM	0.10
MJ115	UL1007 #24 RED 180MM	0.10
MJ116	UL1007 #24 ORANGE 180MM	0.10
MJ117	UL1007 #24 YELLOW 180MM	0.10
MJ118	UL1007 #24 BLACK 180MM	0.10
MJ70701	FLOPPY POW PLUG CAB130MM	0.40
MJ70704	34 WAY ROUND S CABLE	7.10
MJ90201	34 CORES FLAT CABLE #28	0.20
MQ80102	CARD EDGE CNTR 34WAY	1.20
MQ90202	STB CNTR 65495-028 34WAY	0.90
MS70701	3POLE 2THROW POWER SWITCH	0.20
WH60502	FIBRE WASHER D 3.2X8	0.10
WN70701	DISK BOTTOM CABINET	0.90

GENERAL RECOMMENDED SPARE PART LIST FOR SVI-707  
5 1/4" MSX DISK DRIVE

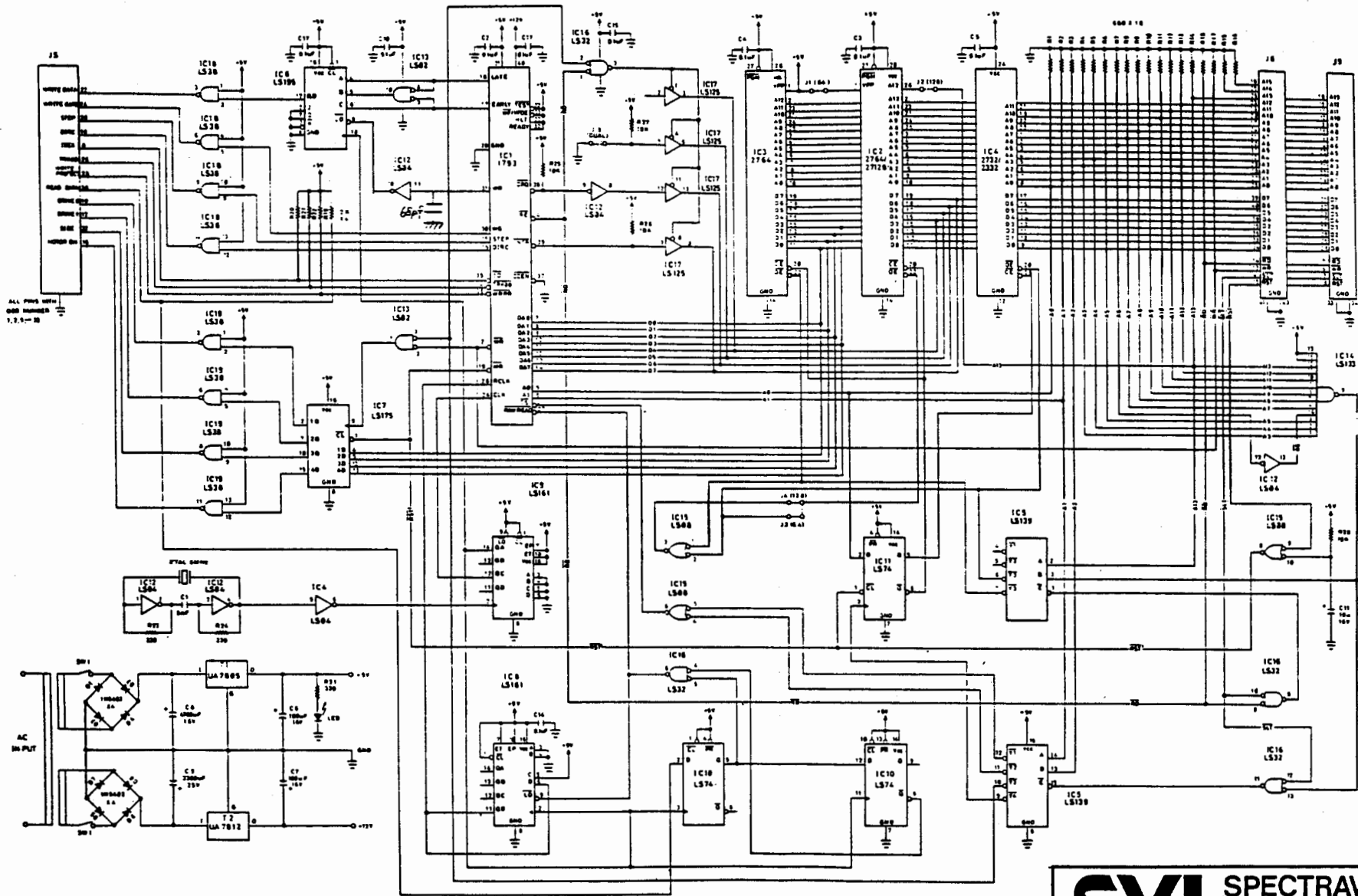
<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
WN70702	DISK TOP CABINET	0.70
WN70704	"POWER" INDICATOR	0.10
WN70705	FELT NET 70X135	0.10
WN70709	HEAT SHRINKABLE D2.0 TUB	0.10
WN70710	BALL 3/32"L4.2"WIRE TIES	0.10
WN70713	RUBBER 10X10X10MM	0.10
WN90203	RUBBER FOOT	0.10
WP70703	SVI-707 BACK LABEL	0.10
XC70701	MODEL NAME PLATE	0.10
XC70705	METAL COVER TOP	0.60
XC70706	METAL COVER BOTTOM	0.30
XC70708	HEAT SINK	0.50
XN015	M3 BRASS STUD	0.10
XS062	D3 X 10PT	0.10
XS069	D3 X 12PA	0.10
XS076N	M3X0.5X68M	0.10
XS082N	SCREW M3X0.5PX10KM	0.10
XS089	SCREW M2X6TTP	0.10
XS093N	M3X0.5PX5 BM	0.10
IC80101	IC FD1793	4.50
ICLS02	IC 74LS02	0.20
ICLS04	IC 74LS04	0.20
ICLS08	IC 74LS08	0.20
ICLS125	IC 74LS125	0.20
ICLS133	IC 74LS133	0.30
ICLS139	IC 74LS139	0.40
ICLS161	IC 74LS161	0.30
ICLS175	IC 74LS175	0.30
ICLS195	IC 74LS195	0.30
ICLS372	IC 74LS32	0.20
ICLS38	IC 74LS38	0.20
ICLS74	IC 74LS74	0.20
ID0009	RTFR DIODE IN5402	0.10
KE3010025	ELECT CAP 10UF 25V	0.10
KE3100016	ELECT CAP 100UF 16V	0.10
KM1472050	MYLAR CAP 4700PF 50V	0.10
KR2100025	CERAMIC CAP 0.1UF 25V	0.10
MP252	4-PINS POWER PLUG	0.40
RF1222	RESIST 220 OHM 1/4W +-5%	0.10
RF1332	RESIST 330 OHM 1/4W +-5%	0.10
RF1682	RESISTOR 680 OHM 1/4W	0.10
RF2202	RESIST 2K 1/4W +-5%	0.10
RF3102	RESISTOR 10K 1/4W +-5%	0.10
SP70701-03	MSX DRIVE PCB VER 1.3	5.70
XC60520	FIXING BRACKET F POWER S	0.10
YS70701	SA 455 DISK DRIVER	85.80
	TOTAL:	133.40

## RECOMMENDED SPARE PART LIST FOR SVI-707 POWER SUPPLY

<u>COUNTRY</u>	<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
MALTA U.K. KUWAIT MALAYSIA ISRAEL	TP707102	BS PW TX 240V W/O CABLE	5.50
AUSTRALIA NEW ZEALAND	TP707104	SAA PW TX 240V W/O CABLE	5.40
CANADA	TP707105	CSA PW TX 110V W/O CABLE	5.00
U.S.A ECUADOR	TP707301	UL PW TX 110V W/O CABLE	5.00
SWEDEN FINLAND NORWAY DENAMRK	TP707106	SEMKO DW TX 220V W/O CABLE	6.00
S. AFRICA FRANCE NETHERLAND W. GERMANY S. ARABIA AUSTRIA ITALY PORTUGAL SWITZERLAND TURKEY YEMEN ICELAND THAILAND LEBANON SPAIN HONG KONG GREECE BELGIUM TURKEY	TP707103	VDE PW TX 220V W/O CABLE	5.60

1.3 CIRCUIT DIAGRAM

- SVI-707 MSX 5.25" Disk Controller (Schematic Drawing)
- SVI-707 MSX 5.25" Disk Controller PCB Layout (Component Side)
- SVI-707 MSX 5.25" Disk Controller PCB Layout (Solder Side)
- SVI-707 MSX 5.25" Disk Controller Component Layout



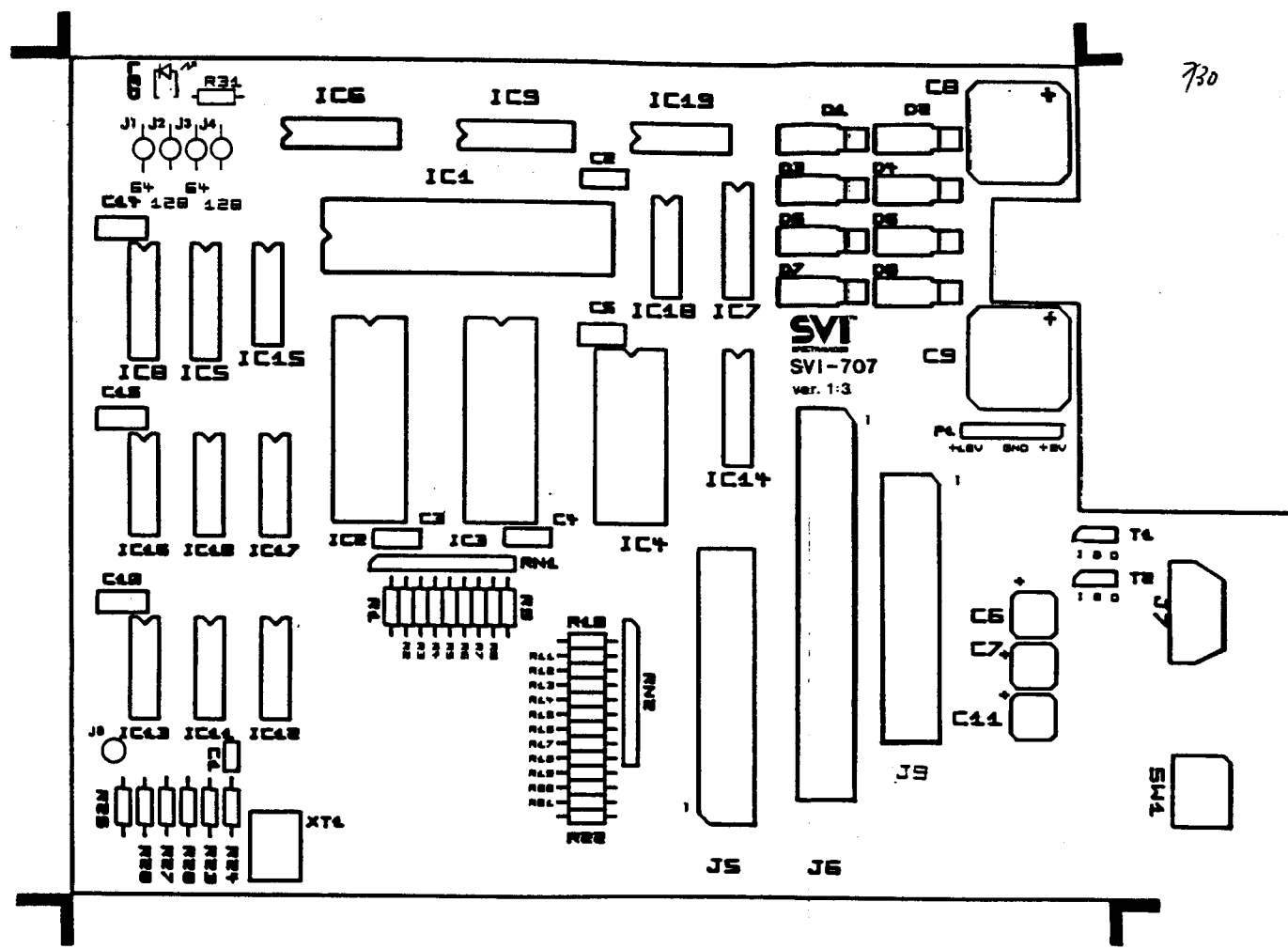
**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LIMITED

TITLE - SVI-707 5 1/4" DISK CONTROLLER (SCHEMATIC DIA.)	VERIFIED BY KIMBA LA.
	DATE 1-7-85
DOCUMENT NO STM-707-001	VERSION SHEET NO 1.3 OF 4

E  
D  
C  
B  
A

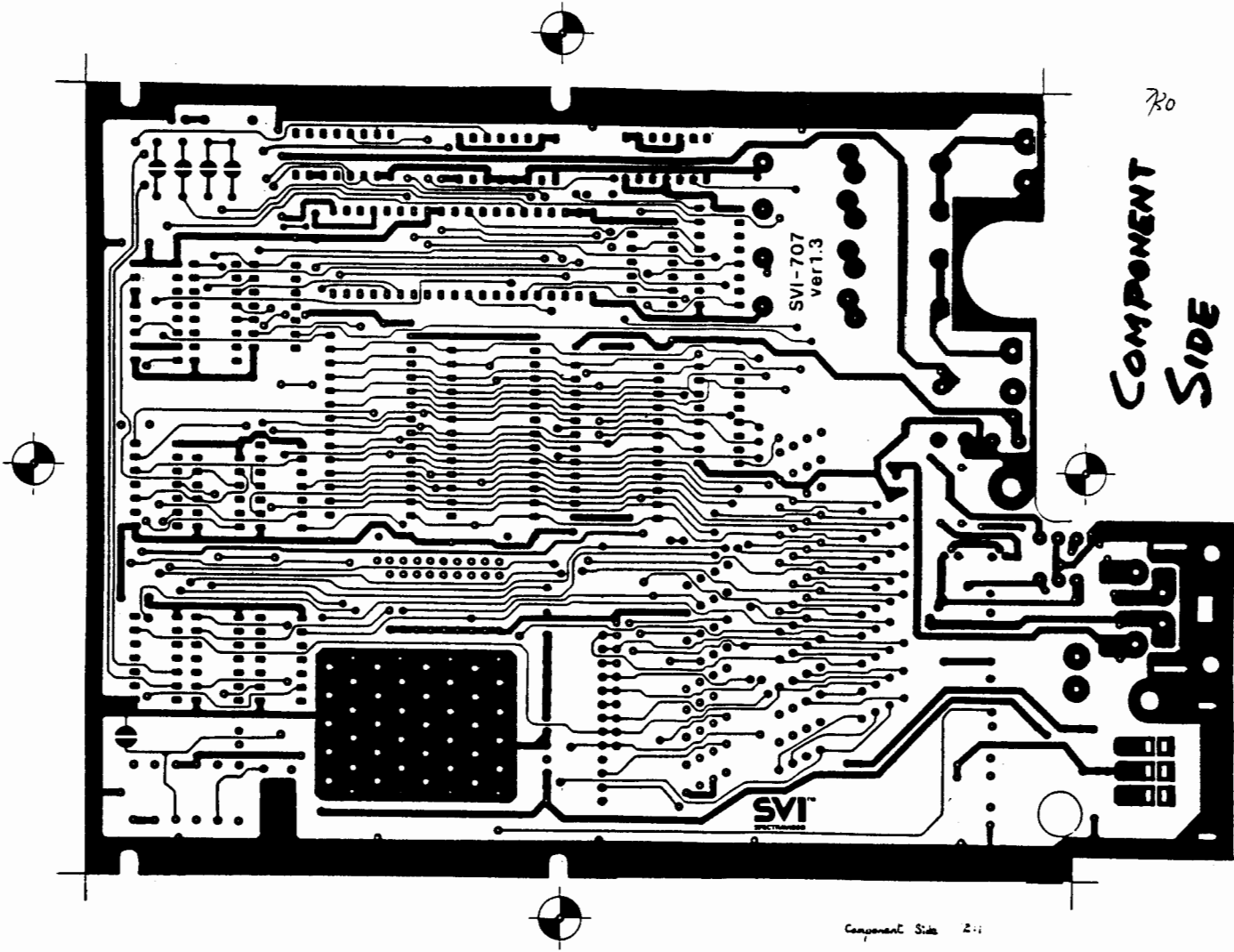


730

CONFIDENTIAL  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE  
PROPERTY OF SPECTRAVIDEO

<b>SVI SPECTRAVIDEO INTERNATIONAL</b> SPECTRAVIDEO LIMITED	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL	
TITLE - SVI-707 5 1/4" DISC DRIVE CONTROLLER COMPONENT LAYOUT	VERIFIED BY <u>KIMB</u> DATE <u>24-</u>
DOCUMENT NO. <u>STM-707-004</u>	VERSION NO <u>1.3</u>





Component Side 2:1

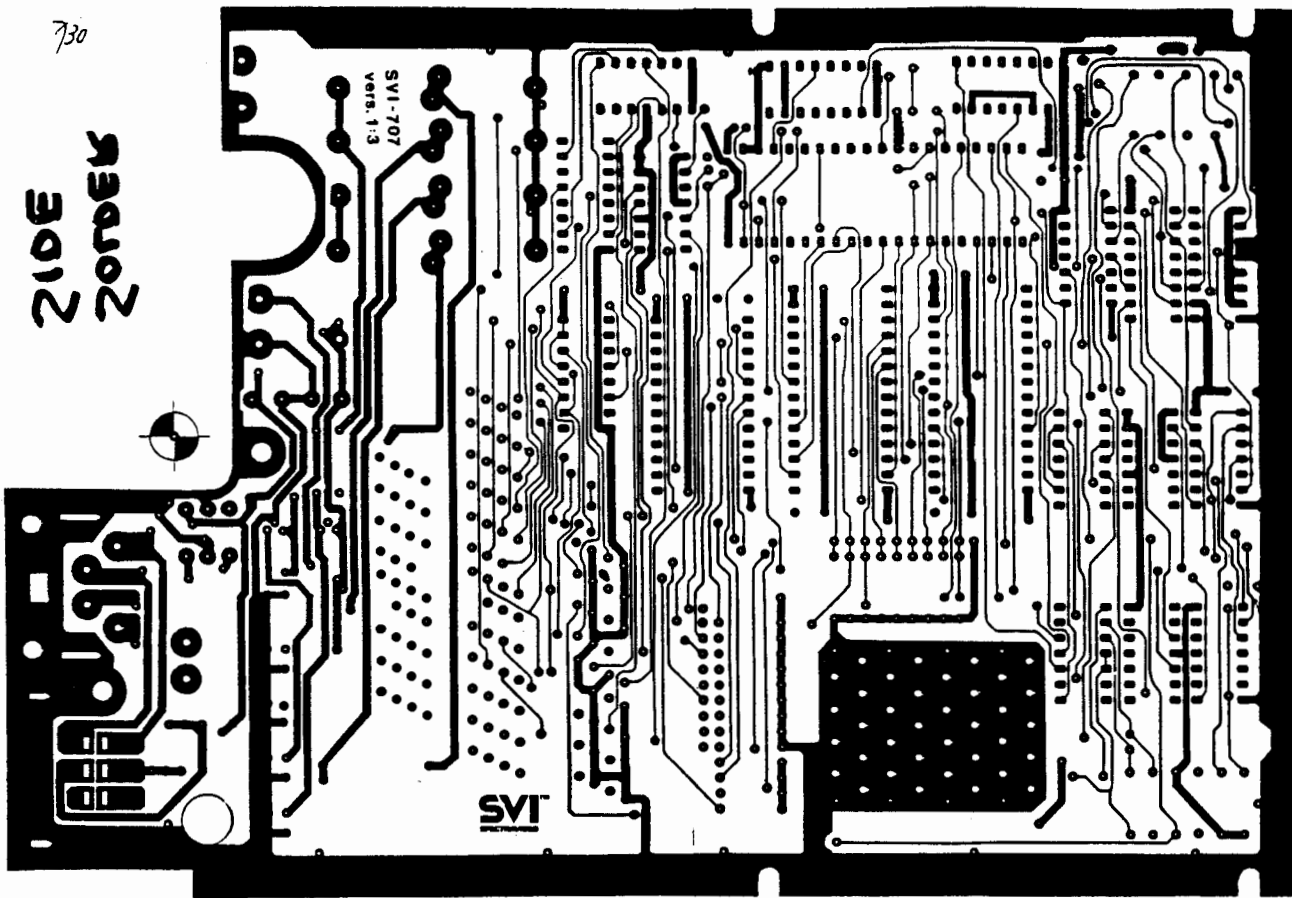
CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 INTERNATIONAL  
 SPECTRAVIDEO LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE — SVI-707 5 1/4" DISC DRIVE CONTROLLER BOARD PCB LAYOUT (COMPONENT SIDE)	VERIFIED BY <u>KIMBA LAU</u> DATE <u>24<sup>th</sup> JULY 1985</u>
--	---

E  
D  
C  
B  
A



Solder Side 2:1

CONFIDENTIAL  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

<b>SVI SPECTRAVIDEO INTERNATIONAL LIMITED</b> SPECTRAVIDEO LIMITED		
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.		
TITLE—SVI-707 5 1/4" Disc DRIVE CONTROLLER BOARD PCB LAYOUT (SOLDER SIDE)	VERIFIED BY <u>KIMBA LAU</u> DATE <u>24-7-85</u>	
DOCUMENT NO. <u>STM-707-003</u>	VERSION NO. <u>1.3</u>	SHEET <u>3</u> OF <u>4</u>

(2) SVI-727 MSX 80 COLUMN VIDEO CARTRIDGE

SVI-727 MSX 80 Column Video Cartridge allows you to switch the video display from 40 column to 80 column. The cartridge is MSX compatible, you can use it with the SVI-728 or other MSX computer.

With this cartridge, most of the 80 column CP/M programmes already available on the market can be run on your computer.

2.1 SPECIFICATION

Character	:	- 7 x 9 dot matrix with 2 blank dots between each character
		- 128 ASCII character set with inverse characters
Display Format	:	80 column x 24 row
Video Output	:	Composite video 2V p-p
Power Consumption	:	+ 5V 300mA + 12V 50mA
Dimensions (mm)	:	170(L) x 123(W) x 32(D)

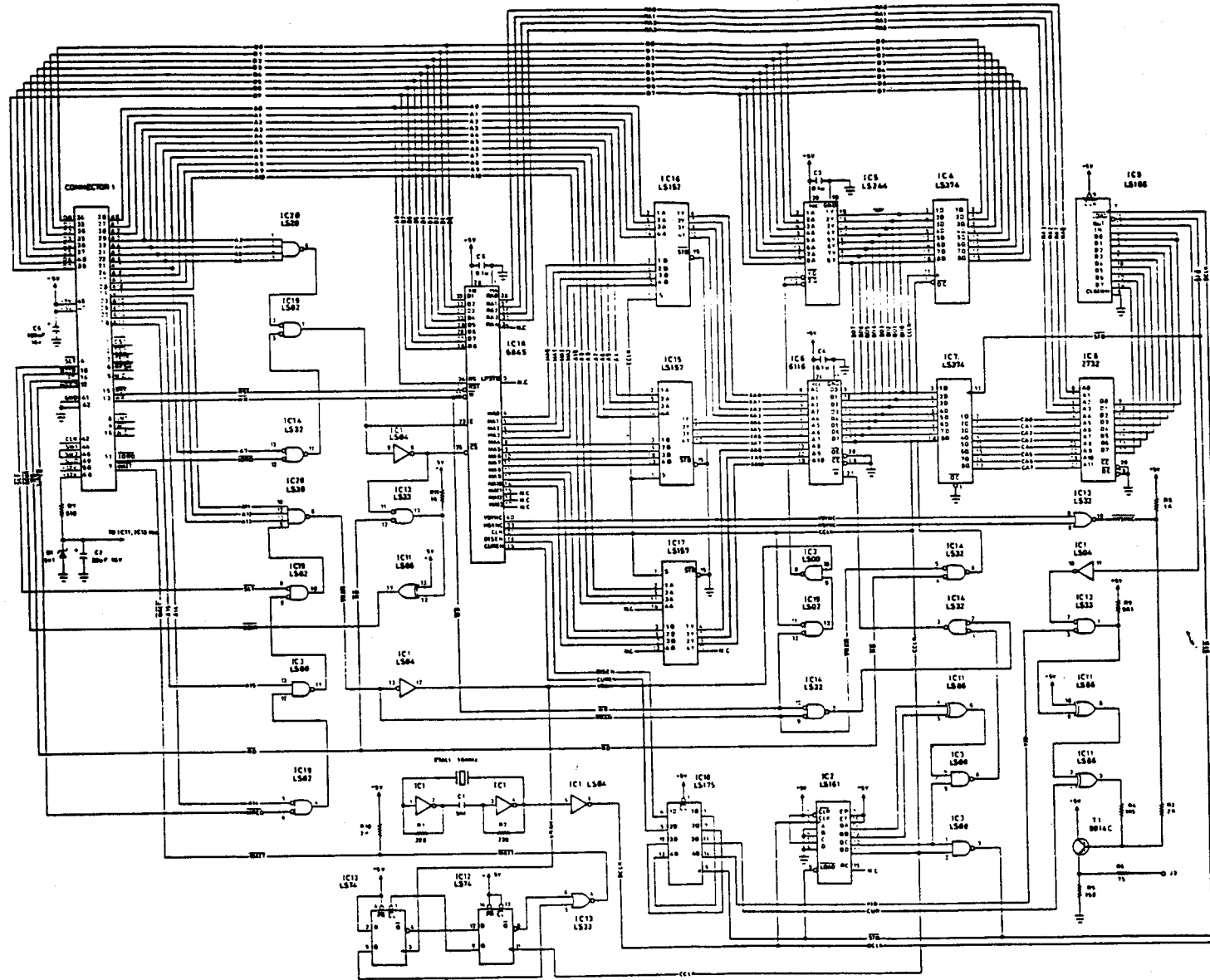
Specifications for product improvement are subject to change without notice.

2.2 SPARE PART LISTGENERAL RECOMMENDED SPARE PART LIST FOR SVI-727 MSX  
80 COLUMN CARTRIDGE


<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
IC6845	IC HD6845SP	2.90
IL00LS	74LS00	0.20
IL02LS	IC 74LS02	0.20
IL04	7404 HEX INVERTERS	0.40
IL157LS	74LS157	0.30
IL161LS	IC 74LS161	0.30
IL166LS	IC 74LS166	0.40
IL175LS	74LS175	0.30
IL20LS	74LS20	0.20
IL244LS	74LS244	0.50
IL32LS	74LS32	0.20
IL33LS	74LS33	0.20
IL374LS	IC 74LS374	0.60
IL74LS	74LS74	0.20
IL86LS	74LS86	0.20
IM2732-45	IC2732 EPROM 450NS	1.90
IM611620	6116 STATIC RAM 200NS	1.80
IT0010	TRANSISTOR 9014C	0.10
IZ0001	5V1 ZENER 1/2W 5%	0.10
KE3033016	33UF 16V E.CAP	0.10
KE3100016	E.CAP. 100UF 16V	0.10
KM1472050	MYLAR CAP 4700PF 50V-EST	0.10
KR2100016	0.1UF 16V CERAM CAP	0.10
MC32016	16MHZ X'TAL	0.50
MJ012	AUDIO-PLUG WITH CORD,5FT	0.50
MP242	24 PINS IC SOCKET	0.10
MP727	RCA VIDEO OUTPUT SOCKET	0.20
PC727101	EXPORT CARTON	0.10
PG727101	G/B FOR MSX 80-COLUMN	0.30
PI318104(S)	GUARANTEE CARD (SVI)	0.10
PI727101	MSX 80-COLUMN MANUAL	0.30
PP727101	POLYFOAM FOR MSX CARD	0.20
PP727102	POLYFOAM BOTTOM	0.20
PZ638002	MODEM PVC BAG	0.10
RF0753	75 OHM 1/4W 5% RESISTOR	0.10
RF1153	150 OHM 1/4W 5% RESISTOR	0.10
RF1223	220 OHM 1/4W +/-5%	0.10
RF1513	510 OHM 1/4W +/-5% RSTR	0.10
RF2103	1K OHM 1/4W +/-5%	0.10
RF2153	1.5K OHM 1/4W +/-5%	0.10
RF2203	2.0K 1/4W 5% RESISTOR	0.10
RF2513	5.1K OHM 1/4W 5% RSTR	0.10
SP72701-01	727 MAIN PCB	3.00
WA003	QC LABEL	0.10
WS019	QC LABEL (WHITE)	0.10
WA088	FCC LABEL	0.10
WA096	SERIAL NO LABEL	0.10
WN727001	MSX 80 COLUMN FRONT COVER	0.30
WN727002	REAR COVER	0.30
XS050	TS/PH SCREW D3X8	0.10
	TOTAL:	18.90

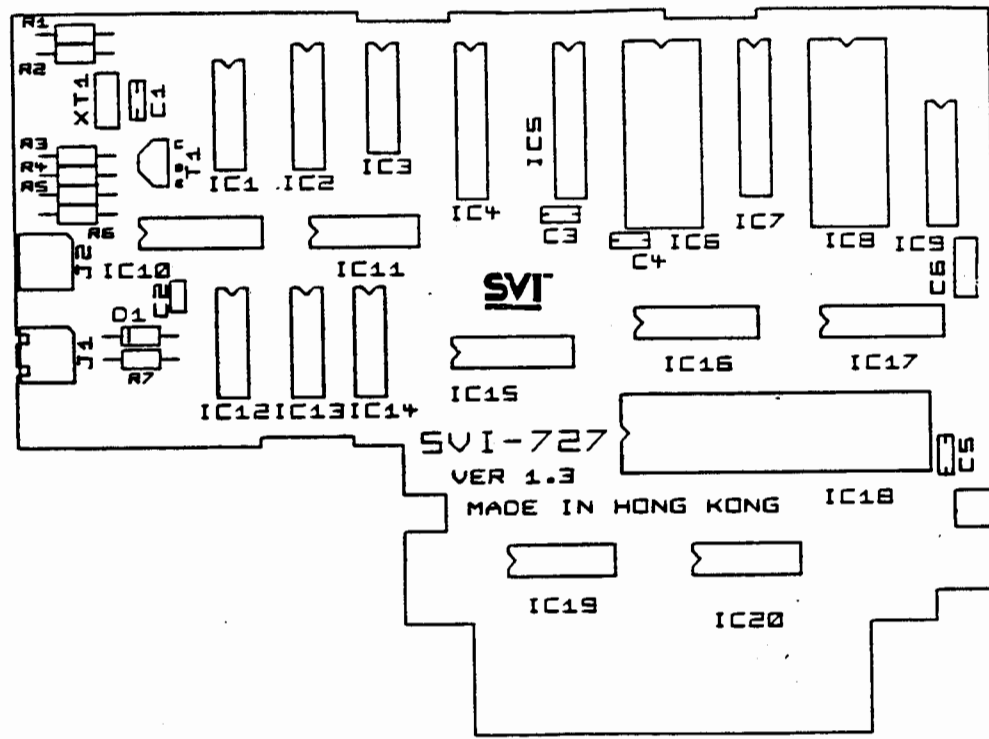
### 2.3 CIRCUIT DIAGRAM

- SVI-727 MSX 80 Column Cartridge (Schematic Diagram)
- SVI-727 MSX 80 Column Cartridge PCB Layout
- SVI-727 MSX 80 Column Cartridge Component Layout



**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

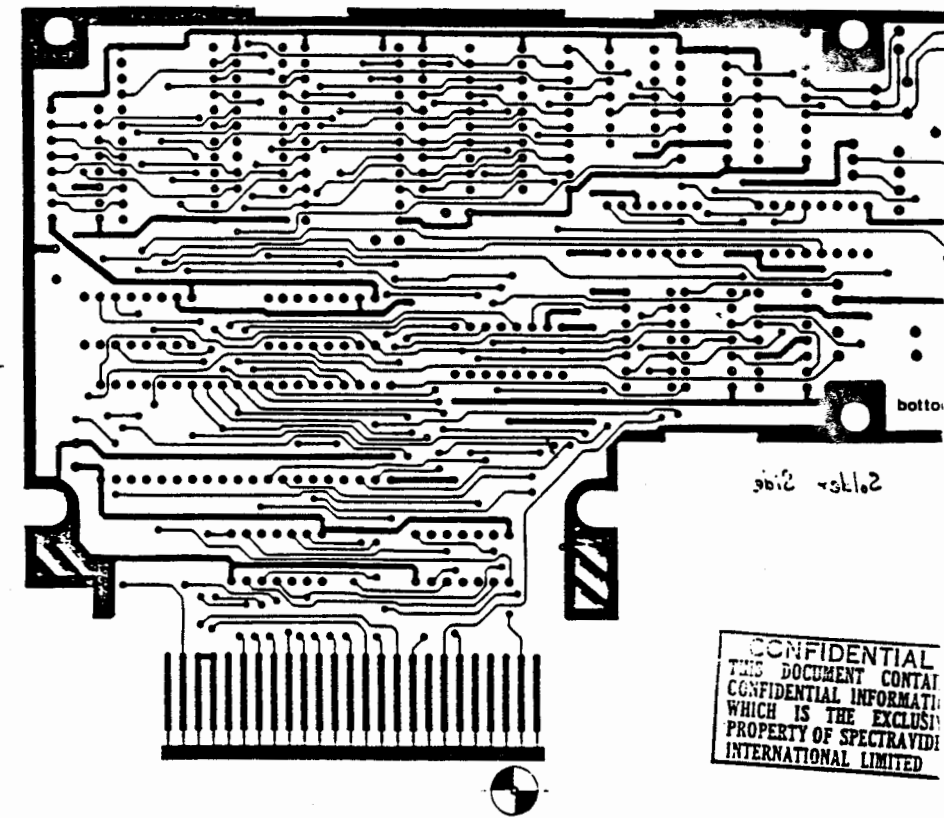
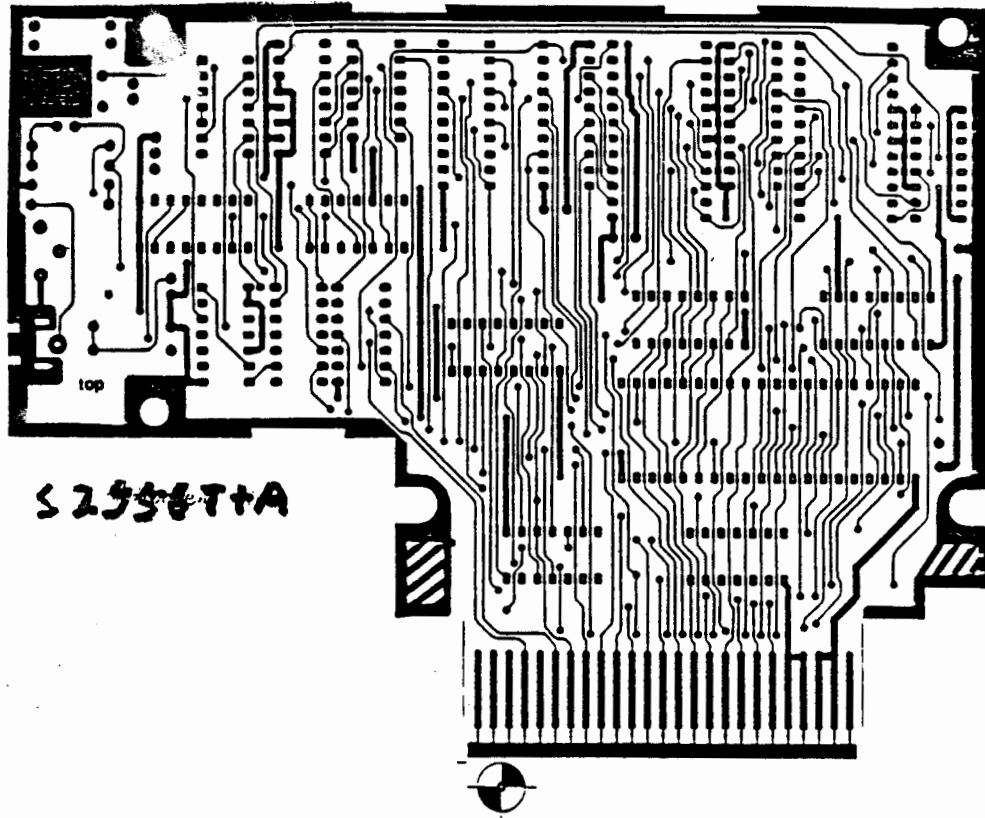
	
<b>SPECTRAVIDEO LIMITED</b>	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE - SVI-727 MSX 80 COLUMN CARTRIDGE (SCHEMATIC DIA.)	VERIFIED BY <u>KIMBA LAU</u> DATE <u>1-7-85</u>
DOCUMENT NO <u>STM-727-004</u>	VERSION NO <u>1.4</u> SHEET <u>4</u> OF <u>4</u>



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE <u>SVI - 727 COMPONENT</u> DRAWING	VERIFIED BY <u>KIMBA</u> DATE <u>17<sup>th</sup> DEC. 1984</u>
DOCUMENT NO <u>STM-727-002</u>	VERSION NO <u>1.3</u> SHEET <u>2</u> OF <u>4</u>

D  
C  
B  
A



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI</b> SPECTRAVIDEO INTERNATIONAL SPECTRAVIDEO - LIMITED -	
CONTROLLED ENGINEERING DOCUMENT THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL	
TITLE--SVI-727 PCB LAYOUT COMPONENT AND SOLDER SIDE	VERIFIED BY <u>KIM</u> DATE <u>27th</u>
DOCUMENT NO. STM-727-003	VERSION NO. 1.3 S.



7 6 5 4 3 2 1

E

D

C

B

A

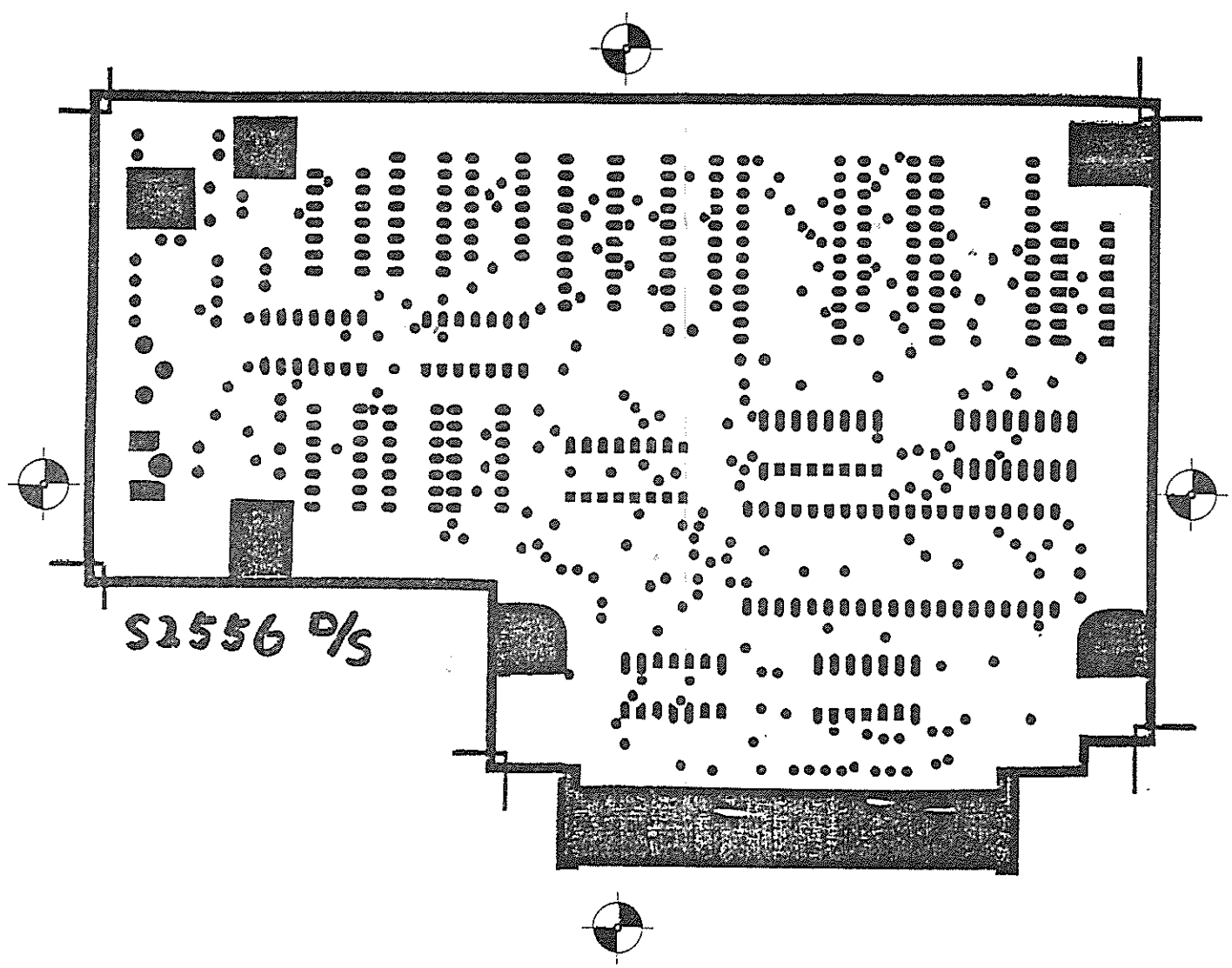
E

D

C

B

A



S2556 P/S

CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI</b> SPECTRAVIDEO SPECTRAVIDEO INTERNATIONAL LIMITED	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE -- SVI - 727 SOLDER MASK DRAWING	VERIFIED BY: <u>KIMBA LAU</u> DATE: <u>27<sup>th</sup> DEC. 1984</u>
DOCUMENT NO STM - 727 - 001	VERSION 1.3 SHEET 1 NO OF 1

7 6 5 4 3 2 1

7 6 5 4 3 2 1

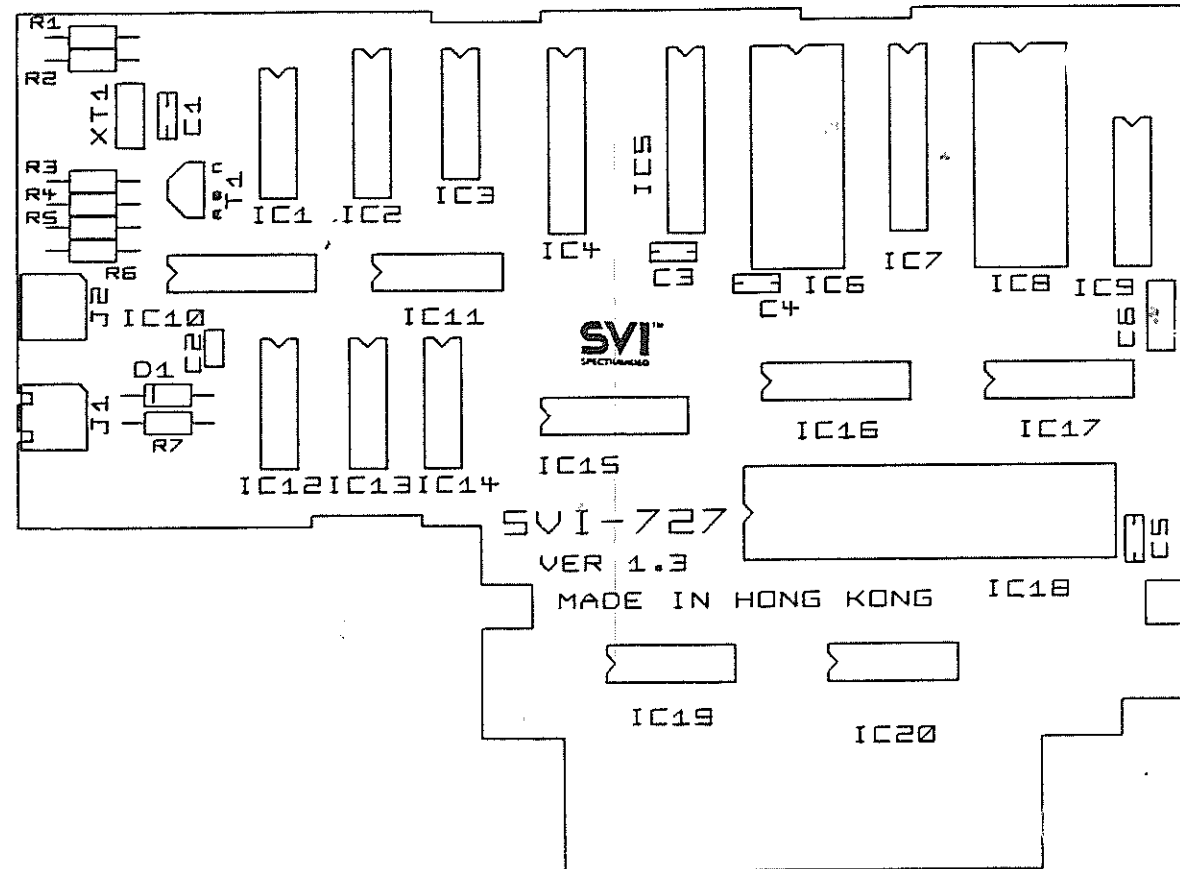
E

D

C

B

A



CONFIDENTIAL  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

<b>SVI SPECTRAVIDEO</b> <b>INTERNATIONAL</b> <b>SPECTRAVIDEO LIMITED</b>	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE—SVI - 727 COMPONENT DRAWING	VERIFIED BY: <u>KIMBA</u> DATE <u>27<sup>th</sup> DEC. 1984</u>
DOCUMENT NO <u>STM-727-002</u>	VERSION NO <u>1.3</u>
	SHEET <u>1</u> OF <u>1</u>

7 6 5 4 3 2 1

7

6

5

4

3

2

1

E

D

C

B

A

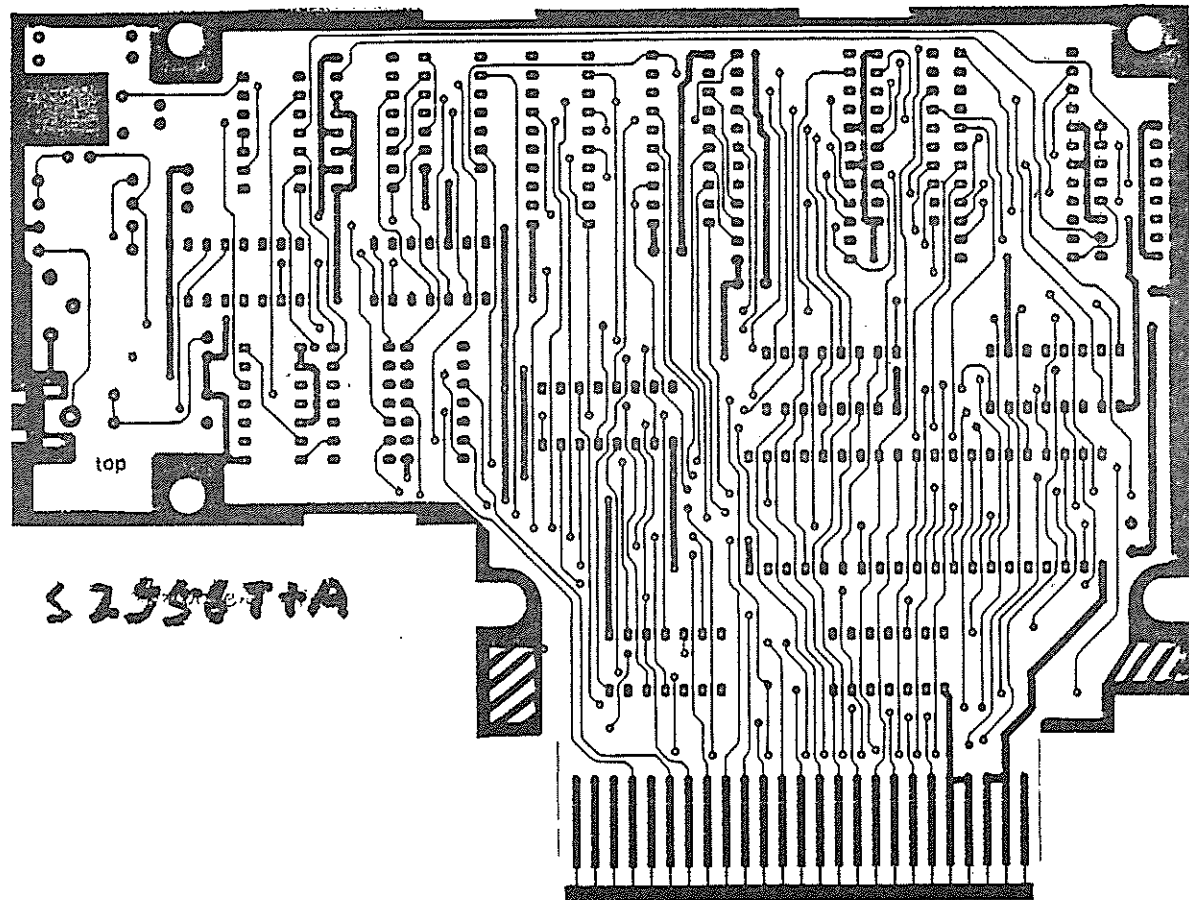
E

D

C

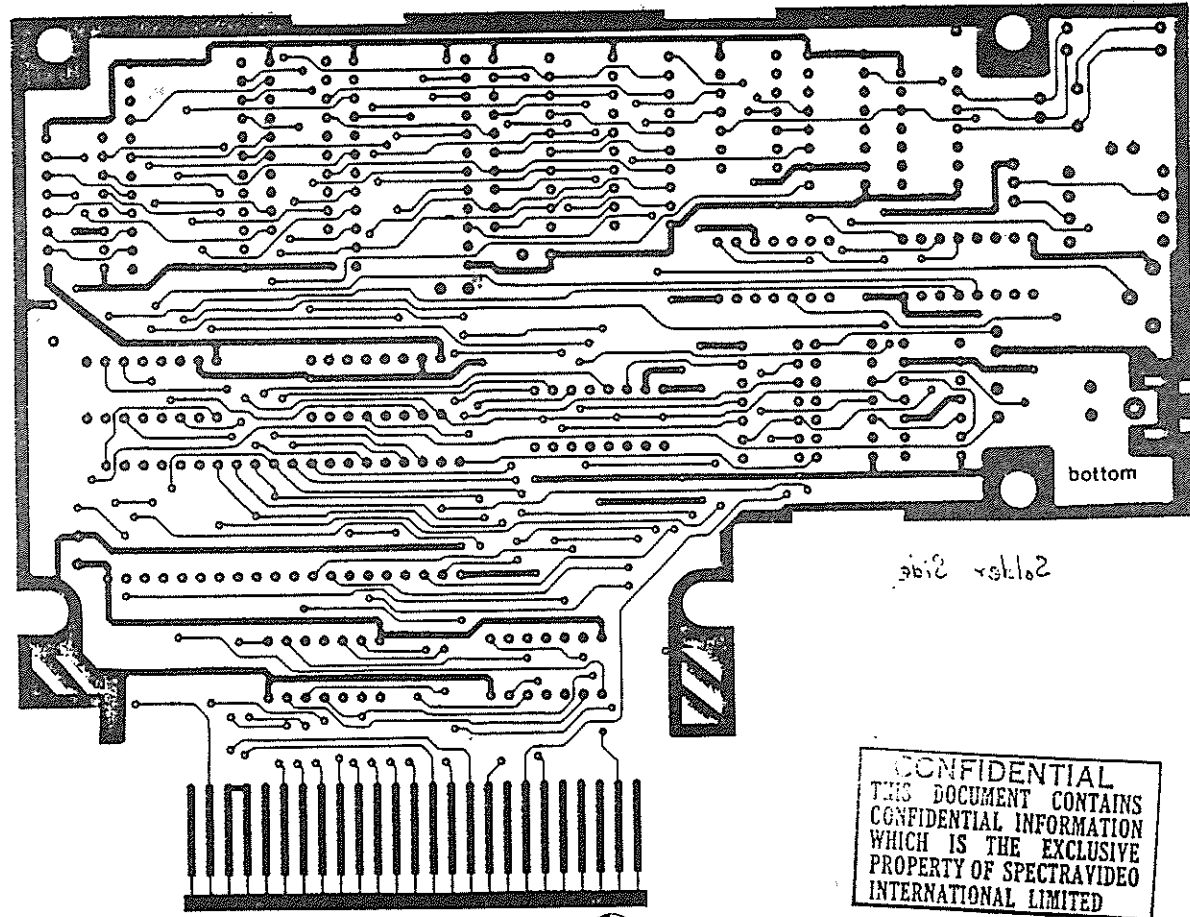
B

A



S 2354T+A

top



bottom

abi2 rsk1.2

CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 INTERNATIONAL  
 SPECTRAVIDEO LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE--SVI-727 PCB LAYOUT COMPONENT AND SOLDER SIDE	VERIFIED BY <u>KIMBA</u> DATE <u>27th DEC. 1984</u>
--	---

DOCUMENT NO STM-727-003	VERSION NO 1.3	SHEET OF 1
----------------------------	-------------------	---------------

7

6

5

4

3

2

1

(3) SVI-737 MSX 300 BAUD MODEM WITH RS-232 INTERFACE

SVI-737 MSX 300 Baud Modem with RS-232 is a modem plus built-in RS-232 interface.

The Modem which is CCITT V21 compatible, can be directly coupled to telephone line. Auto-answering and full-duplex operation is carried out at 300 bps transmission rate. The RS-232 is Electronic Industrial Association (EIA) standard with selectable baud rate, word length, stop bits and parity bit.

SVI-737 is MSX compatible, you can connect it to SVI-728 or other MSX computer.

2.1 SPECIFICATIONModem

Standard	:	CCITT V21 compatible
Data Rate	:	300 baud
Communication Mode	:	Asynchronous, full-duplex
Telephone Line Interface	:	FCC approved (pending) direct connection to 600 ohms line impedance
Modulation	:	Frequency shift keying (FSK)
Operation Functions	:	Originate - manual
		Answer - manual / automatic

Frequencies : Originate  
- mark : 978.8 Hz  
space: 1179.8 Hz  
  
Answer  
- mark : 1647.9 Hz  
space: 1847.7 Hz

Receiver Sensitivity : -48 dBm

Transmitter Output Level : 0.7746V at 600 ohms  
line impedance

RS-232 Interface

Baud Rate : 50 - 19200 bps  
(15 different baud  
rates)

Counter Channel Usage : CH0 - receive baud  
rate clock  
  
CH1 - transmit baud  
rate clock  
  
CH2 - used by appli-  
cation  
(e.g. time out  
counter)

Power consumption : + 5V 250mA  
+ 12V 30mA  
- 12V 28mA

Dimensions (mm) : 170(L) x 123(W) x  
32(D)

Specifications for product improvement are  
subject to change without notice.

3.2 SPARE PART LIST

GENERAL RECOMMENDED SPARE PART LIST FOR SVI-737 MODEM  
WITH RS232C INTERFACE CARTRIDGE

<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
IC1488	IC 1488	0.40
IC1489	IC 1489	0.40
IC324	QUAD-AMP 324	0.50
IC4N29	OPTOCOUPLER 4N29	0.60
IC6234	TONE DIAL CHIP 6234	0.80
IC6926	MSM 6926 MODEM CHIP	9.10
IC8251A	8251A USART	1.80
IC8253	8253A PROGRAMMABLE TIMER	1.80
ID0001	IN4148:SILICON DIODE	0.10
ID0005	IN 4001 DIODE	0.10
IL04LS	74LS04	0.20
IL08LS	74LS08	0.20
IL138LS	74LS138	0.30
IL139LS	74LS139	0.30
IL174LS	IC 74LS174	0.30
IL273LS	74 LS273	0.70
IL32LS	74LS32	0.20
IL367LS	74LS367	0.30
IL38LS	IC 74LS38	0.20
IL74LS	74LS74	0.20
IM27643	2764 EPRON 300NS	3.40
IT0010	TRANSISTOR 9014C	0.10
IZ0007	5V ZENER 1/2W +/- 4%	0.10
KE2001025	ELECT CAP 1UF 25V	0.10
KE22200016	2.2 UF 16V TAN.CAP	0.10
KE3010025	ELECT CAP 10UF 25V	0.10
KM2010100	MYLAR CAP 0.01UF 100V	0.10
KM2100250	0.1U 250V CAP	0.10
KR1100050	100PF 50V NPO CAP.	0.10
KR1330050	C.CAP. 330PF 50V	0.10
KR2100016	0.1UF 16V CERAM CAP	0.10
MC014	SPST REED RELAY <3MA	0.80
MC3200184S	1.8432 MHZ X'TAL-SMAL	1.20
MP242	24 PINS IC SOCKET	0.10
MP252-1	RS232 SOCKET	1.50
MP282	28 PIN IC SOCKET	0.20
MPT04-1	TEL. SOCKET 4 PINS	0.50
RF1183	180 OHM 1/4W RESISTOR	0.10
RF1823	820 OHM RSTR 1/4W 5%	0.10
RF2103	1KOHM 1/4W +/-5%	0.10
RF2203	2.0K 1/4W 5% RESISTOR	0.10
RF2273	2.7K 1/4W 5% RESISTOR	0.10

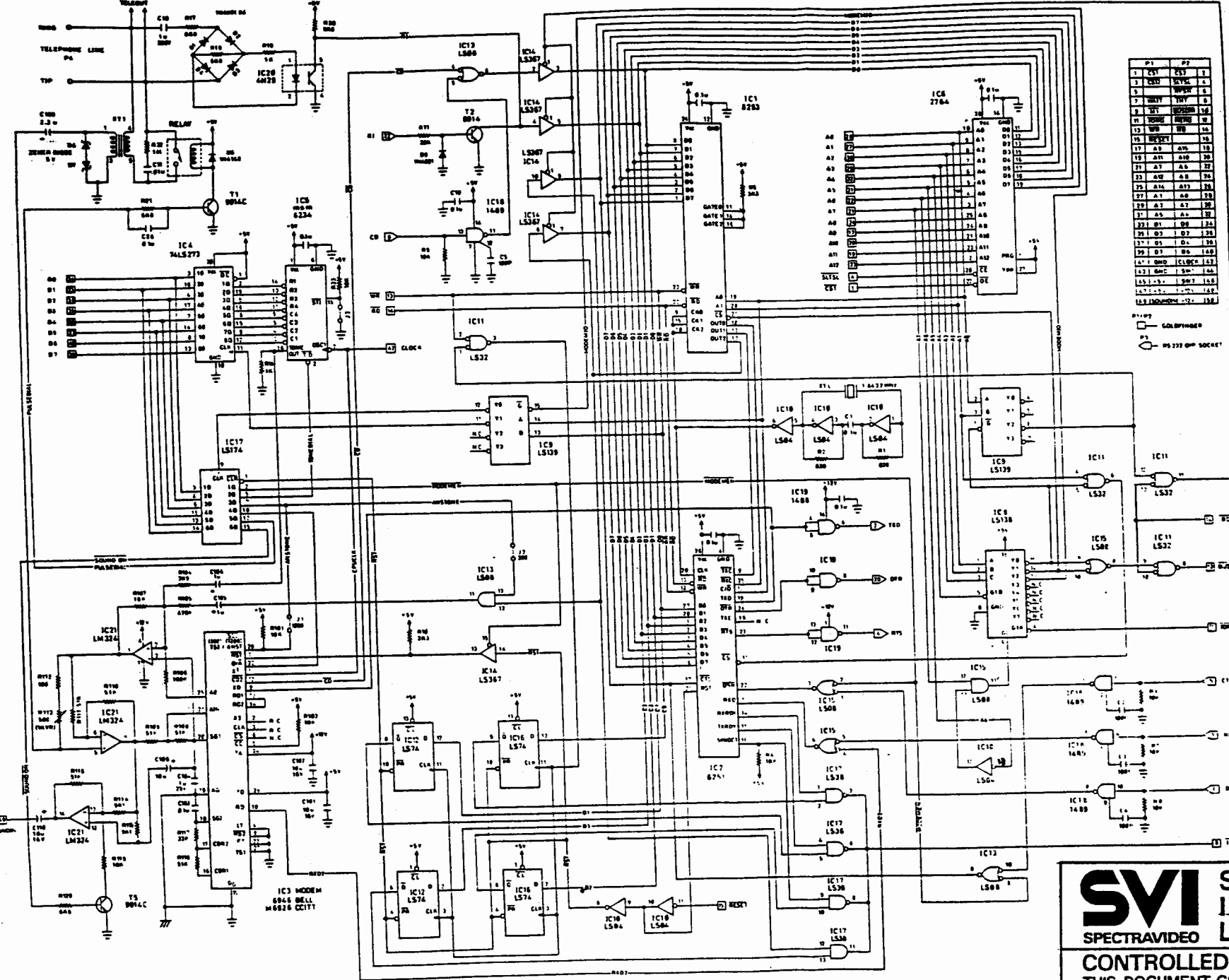
GENERAL RECOMMENDED SPARE PART LIST FOR SVI-737 MODEM  
WITH RS232C INTERFACE CARTRIDGE

<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
RF2333	3K3 1/4W 5% RESISTOR	0.10
RF2393	3.9K OHM 1/4W +/-5%	0.10
RF2683	6K8 OHM 1/4W +/-5%	0.10
RF3103	10K OHM 1/4W +/-5%	0.10
RF3153	15K OHM 1/4W +/-5% RESIST	0.10
RF3183	18K OHM 1/4W +/-5% RESIST	0.10
RF3203	20K OHM 1/4W +/-5%	0.10
RF3333	33K 1/4W 5% RESISTOR	0.10
RF4103	100K OHM 1/4W +/-5%	0.10
RF4473	470K OHM 1/4W +/-5%	0.10
RF5102	1M OHM +/-5% 1.4W	0.10
RV2101	1K OHM VARIABLE-RESISTOR	0.10
SP73701-01	PCB MSX CARD FORMAT	3.00
TA0005	X'FORMER 1:1, 600 MATCH	0.20
WA003	QC LABEL	0.10
WA019	QC LABEL (WHITE)	0.10
WA088	FCC LABEL	0.10
WN727002	REAR COVER	0.30
WN737001	MSXRS232 & MODEM F.COVER	0.30
XS050	TS/PH SCREW D3X8	0.10
XS054	MS/PH SCREW M3X0.5PX8	0.10
DD757101	5 1/4" BLANK DISK 48TPI	1.40
MJ73701	4 WIRE TEL. CABLE D.SIDE	0.40
PC737101	EXPORT CARTON	0.10
PG737101	G/B FOR MSX 232 & MODEM	0.30
PI318104(S)	GUARANTEE CARD (SVI)	0.10
PI737101	MSX 232 & MODEM MANUAL	0.50
PI757102	RS232 SOFTWARE MANUAL	0.40
PP727101	POLYFOAM FOR MSX CARD	0.20
PP727102	POLYFOAM BOTTOM	0.20
PZ638002	MODEM PVC BAG	0.10
WA125	SERIAL NO LABEL - 737	0.10
WA126	CP/M SYSTEM DISK LABEL	0.10
	TOTAL:	37.20

### 3.3 CIRCUIT DIAGRAM

- SVI-737 MSX RS232 & Modem Cartridge (Schematic Diagram)
- SVI-737 MSX RS232 & Modem Cartridge PCB Layout (Component Side)
- SVI-737 MSX RS232 & Modem Cartridge PCB Layout (solder side)
- SVI-737 MSX RS232 & Modem Cartridge Component Layout





P1	P2	P3
1	CS1	CS2
2	CS3	CS4
3	CS5	CS6
4	CS7	CS8
5	CS9	CS10
6	CS11	CS12
7	CS13	CS14
8	CS15	CS16
9	CS17	CS18
10	CS19	CS20
11	CS21	CS22
12	CS23	CS24
13	CS25	CS26
14	CS27	CS28
15	CS29	CS30
16	CS31	CS32
17	CS33	CS34
18	CS35	CS36
19	CS37	CS38
20	CS39	CS40
21	CS41	CS42
22	CS43	CS44
23	CS45	CS46
24	CS47	CS48
25	CS49	CS50
26	CS51	CS52
27	CS53	CS54
28	CS55	CS56
29	CS57	CS58
30	CS59	CS60
31	CS61	CS62
32	CS63	CS64
33	CS65	CS66
34	CS67	CS68
35	CS69	CS70
36	CS71	CS72
37	CS73	CS74
38	CS75	CS76
39	CS77	CS78
40	CS79	CS80
41	CS81	CS82
42	CS83	CS84
43	CS85	CS86
44	CS87	CS88
45	CS89	CS90
46	CS91	CS92
47	CS93	CS94
48	CS95	CS96
49	CS97	CS98
50	CS99	CS100

P1-P2  
 P3  
 P4  
 P5  
 P6  
 P7  
 P8  
 P9  
 P10  
 P11  
 P12  
 P13  
 P14  
 P15  
 P16  
 P17  
 P18  
 P19  
 P20  
 P21  
 P22  
 P23  
 P24  
 P25  
 P26  
 P27  
 P28  
 P29  
 P30  
 P31  
 P32  
 P33  
 P34  
 P35  
 P36  
 P37  
 P38  
 P39  
 P40  
 P41  
 P42  
 P43  
 P44  
 P45  
 P46  
 P47  
 P48  
 P49  
 P50

**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

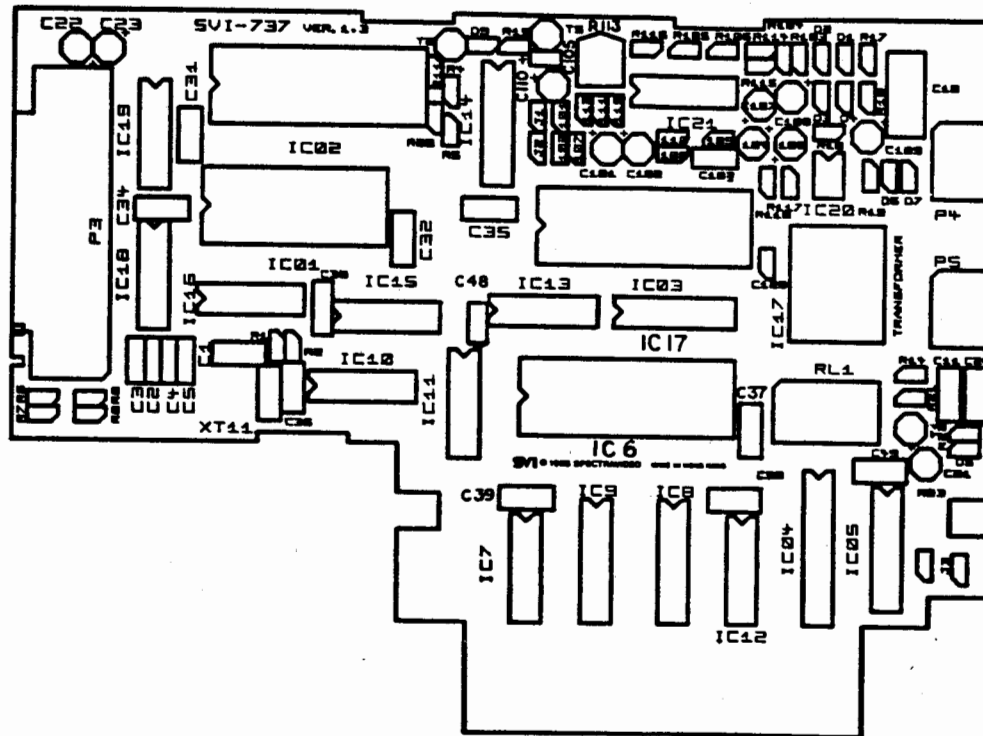
**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LIMITED

TITLE — SVI-737 MSX  
 RS232 & MODEM CARTRIDGE  
 (SCHEMATIC DIA.)

VERIFIED BY KIMBA LAU  
 DATE 1-7-85

DOCUMENT NO STM-737-001

VERSI SHEET 1  
 NO 1.3 OF 4

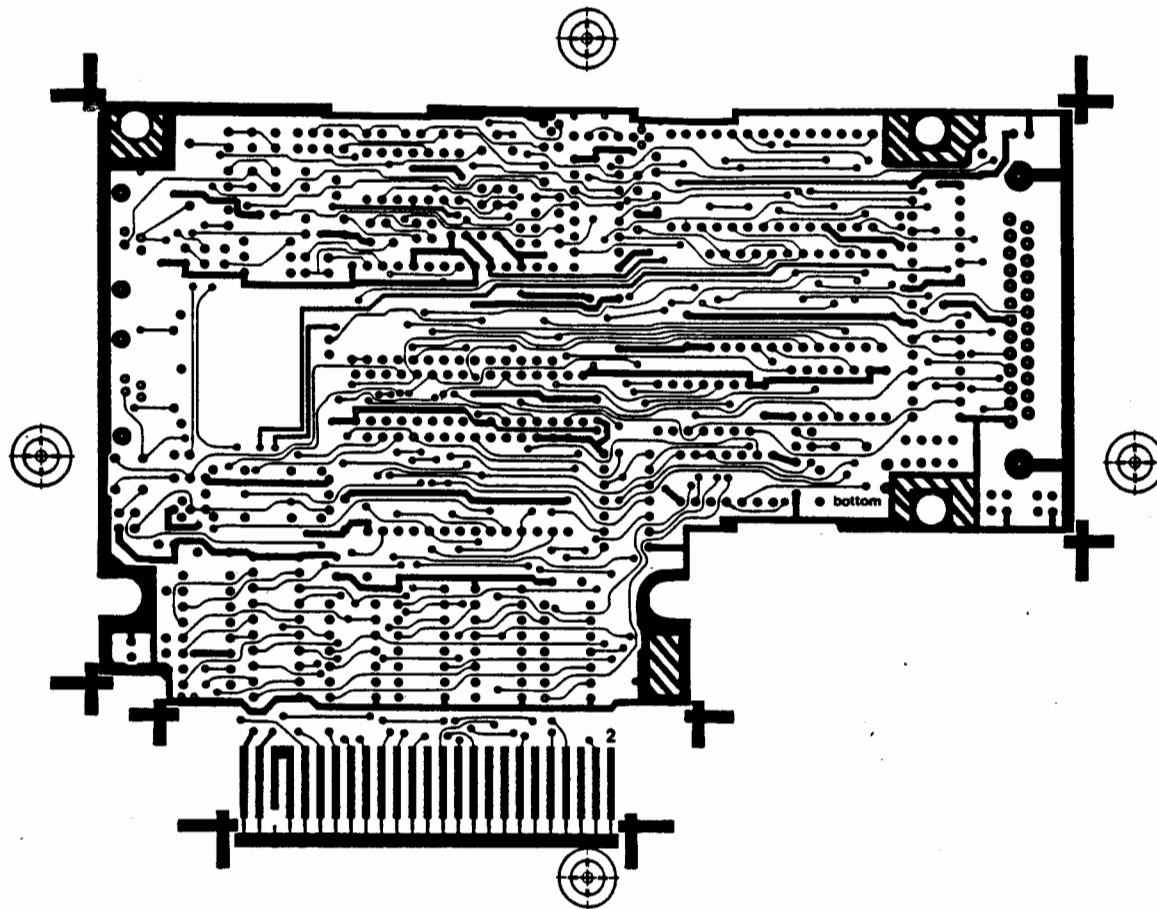


CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 INTERNATIONAL  
 SPECTRAVIDEO LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

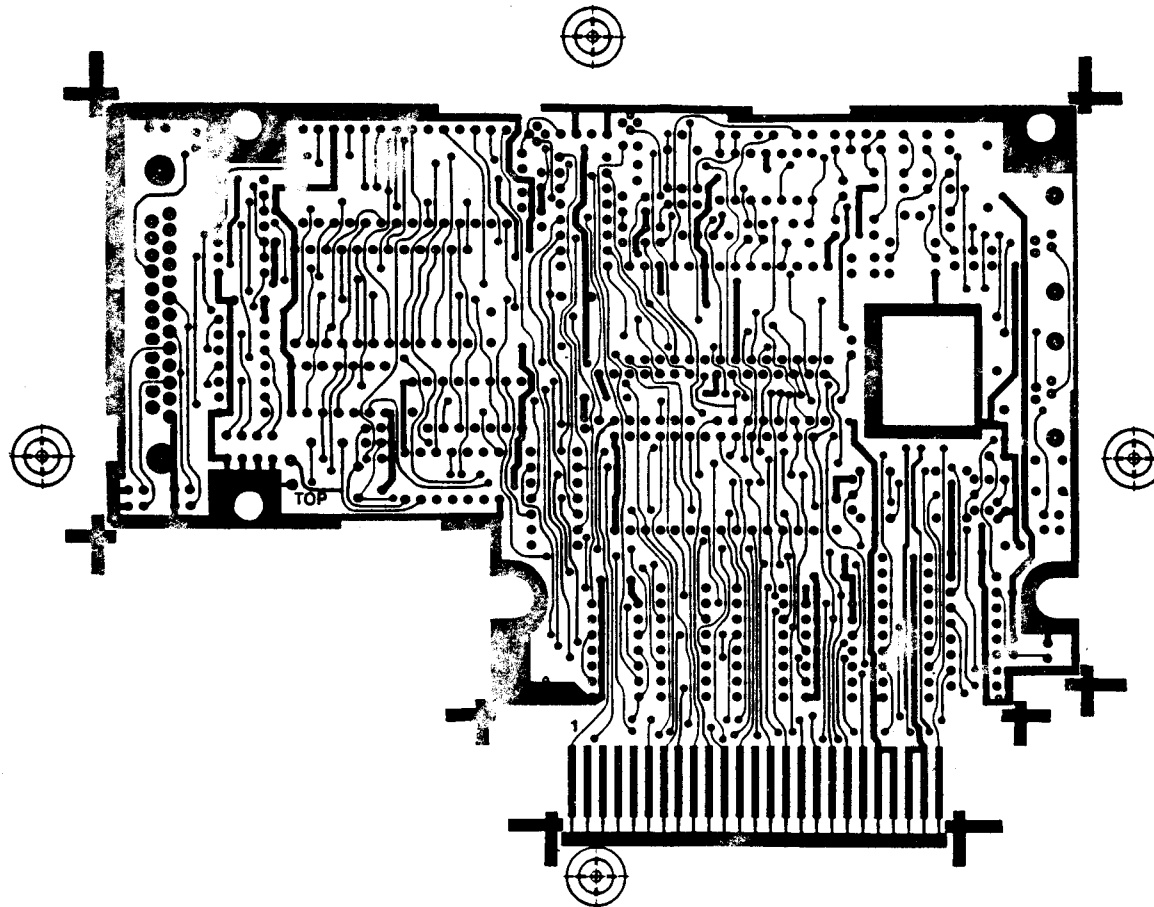
TITLE — SVI-737 MSX RS232 & MODEM CARTRIDGE COMPONENT LAYOUT	VERIFIED BY <u>KIMBA LAU</u> DATE <u>18-7-1985</u>
DOCUMENT NO. <u>STM-737-004</u>	VERSION NO. <u>4</u> SHEET OF <u>4</u>



**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

- \* NOTE: (1) Connect a jumper wire between IC 10 pin 4 to pin 5.
- (2) Connect a jumper wire between IC 8 pin 5 to A5 [31]
- \* This applies only to early version PCB.

<b>SVI SPECTRAVIDEO INTERNATIONAL LIMITED</b> SPECTRAVIDEO - LIMITED -		
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.		
TITLE — <b>SVI-737 MSX RS232 &amp; MODEM CARTRIDGE PCB LAYOUT (COMPONENT SIDE)</b>	VERIFIED BY <b>MBA LAU</b> DATE <b>-7-1985</b>	
DOCUMENT <b>STM-737-002</b>	VERSION <b>1.3</b>	SHEET <b>2</b> OF <b>4</b>



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 INTERNATIONAL  
 SPECTRAVIDEO LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE — SVI-737 MSX RS232  
 & MODEM CARTRIDGE PCB  
 LAYOUT (SOLDER SIDE)

VERIFIED BY KIMBA LAU  
 DATE 18-7-1985

(4) SVI-747 MSX 64K RAM MEMORY CARTRIDGE

SVI-747 MSX 64K RAM Memory Cartridge provides continuous 64K bytes memory to your computer. Since it is MSX compatible, it can be connected to SVI-728 or other MSX computer.

With the additional memory capacity, you can run a lot of software. It plugs into the cartridge or cartridge expansion slots of all MSX computers. It can be used as a RAM disk in CP/M system with the SVI-707.

4.1 SPECIFIATION

Memory : 64K RAM

Power Consumption : 5V 200mA

Dimensions (mm) : 170(L) x 123(W) x 32(D)

- N.B. - No additional power supply
- Adapt to game slot of SVI-728 MSX computer
  - Provide continuous 64K byte memory for those MSX computers which do not have 64K memory built-in

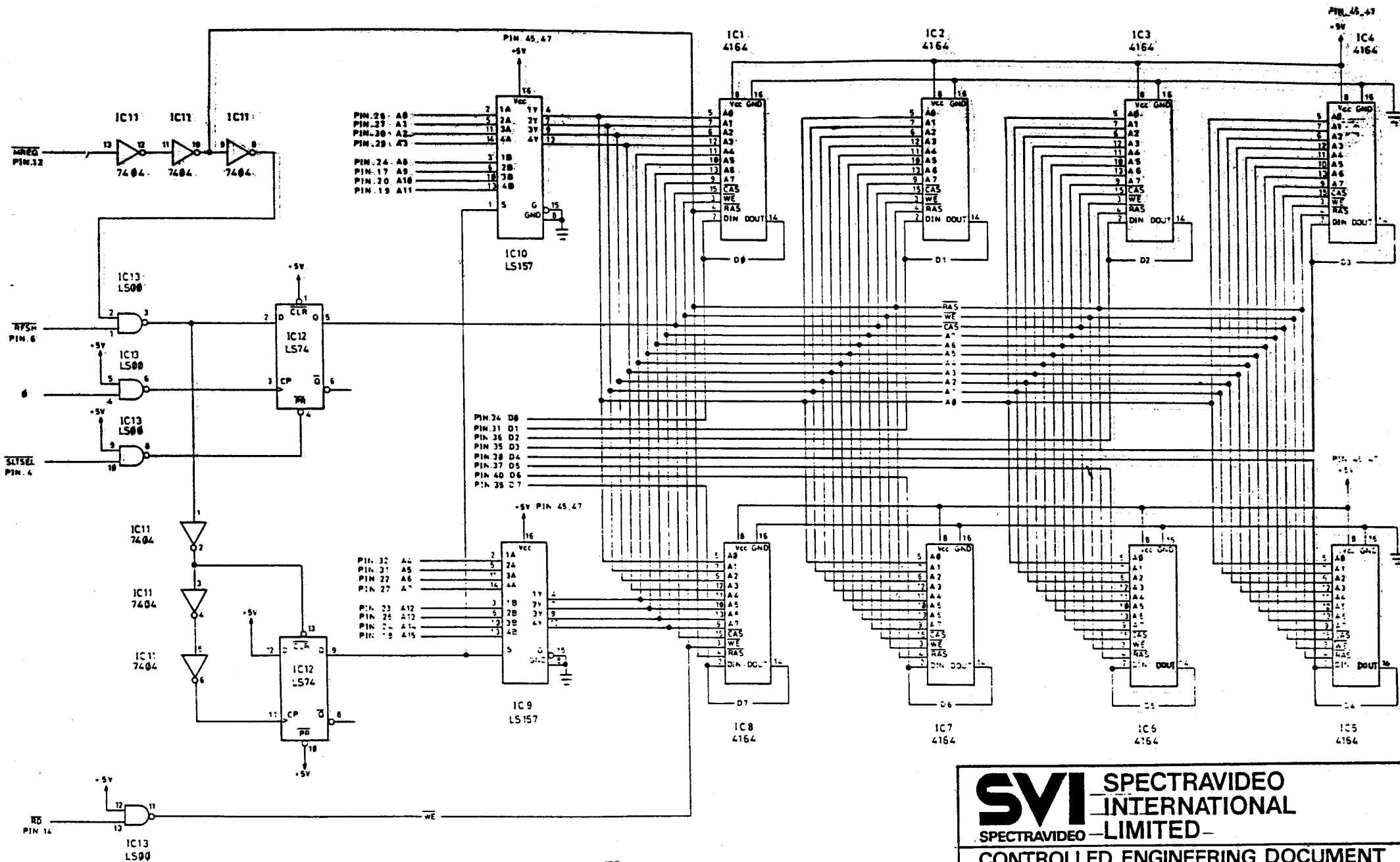
Specifications for product improvement are subject to change without notice.

4.2 SPARE PART LISTGENERAL RECOMMENDED SPARE PART LIST FOR SVI-747 MSX  
64K RAM CARTRIDGE

<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
PI31807	SVI WARRANTY	0.10
MM00001	MOISTURE ABSORBANT (5GM)	0.10
PC747101	OUTC 350X250X380MM	0.10
PG747101	G/B FOR MSX 64K RAM	0.30
PI747101	MSX 64K RAM MANUAL	0.30
PP72701	POLYFOAM FOR MSX CARD	0.40
PZ72701	PLASTIC BAG	0.10
WP72701	FCC LABEL	0.10
IM416425	D.RAM 4164 250NS	1.00
WN70002	REAR COVER	0.30
WN70003	MSX 64K FRONT COVER	0.30
XS061	M 3 X 6PT	0.10
ICLS00	IC 74LS00	0.20
ICLS04	IC 74LS04	0.20
ICLS157	IC 74LS157	0.30
ICLS74	IC 74LS74	0.20
KE3100016	ELECT CAP 100UF 16V	0.10
KR2100012	CERAMIC CAP 0.1UF 12V	0.10
SP74701-01	MAIN PCB MSX 64K RAM	3.90
	TOTAL:	8.20

4.3 CIRCUIT DIAGRAM

- SVI-747 MSX 64K RAM Cartridge (Schematic Diagram)
- SVI-747 MSX 64K RAM Cartridge PCB Layout
- SVI-747 MSX 64K



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
 INTERNATIONAL  
 SPECTRAVIDEO LIMITED

CONTROLLED ENGINEERING DOCUMENT  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
 THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LIMITED

TITLE - SVI-747 MSX 64K  
 RAM EXPANSION CARTRIDGE  
 (SCHEMATIC DIA.)

VERIFIED BY KIMBA LAU  
 DATE 1-7-85

DOCUMENT NO STM-747-003

VERSION NO 1.0 SHEET 3 OF 3



7 | 6 | 5 | 4 | 3 | 2 | 1

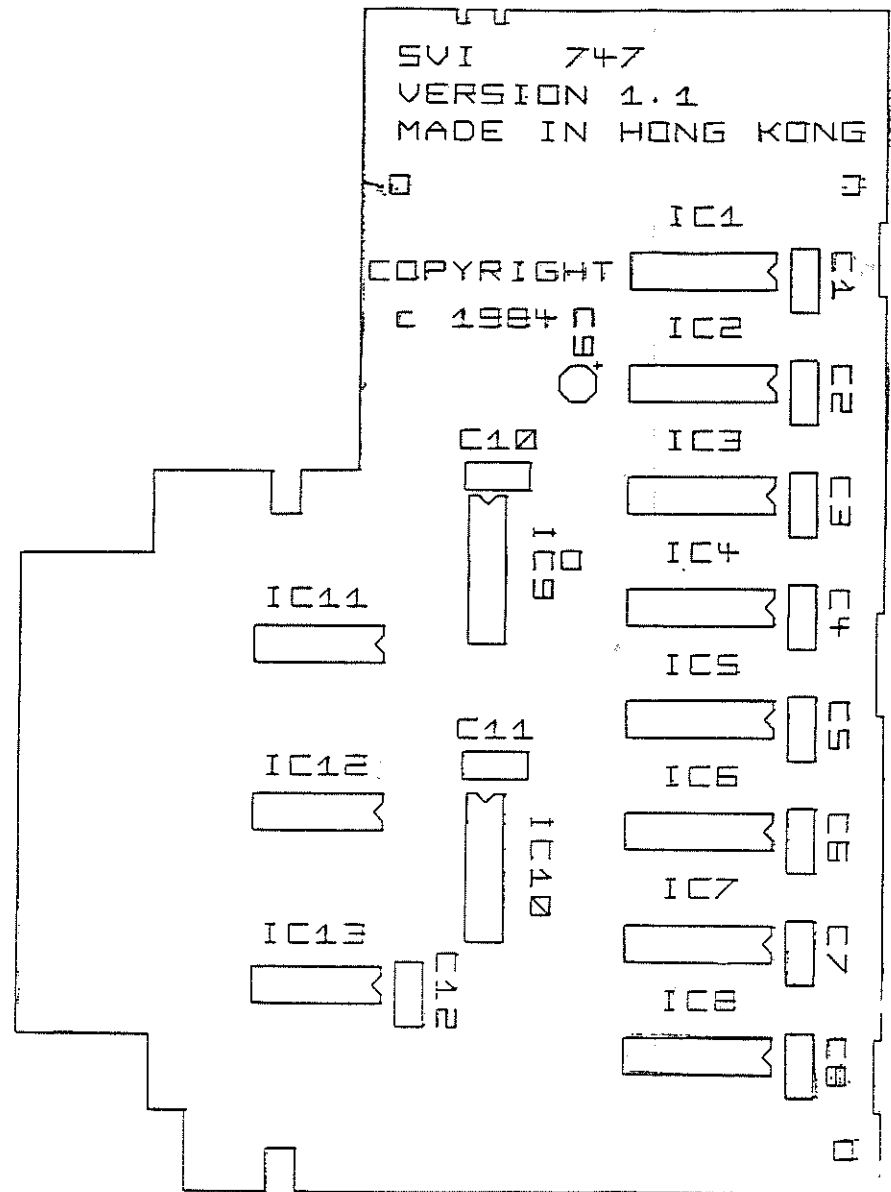
E

D

C

B

A



**CONFIDENTIAL**  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

<b>SVI SPECTRAVIDEO INTERNATIONAL LIMITED</b>	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE - SVI-747 COMPONENT DRAWING	VERIFIED BY <u>KIMBA LAU</u> DATE <u>27<sup>th</sup> DEC. 1984</u>
DOCUMENT NO <u>STM-747-φφ1</u>	VERSION NO <u>1.1</u> SHEET <u>1</u> OF <u>1</u>

7 | 6 | 5 | 4 | 3 | 2 | 1

7

6

5

4

3

2

1

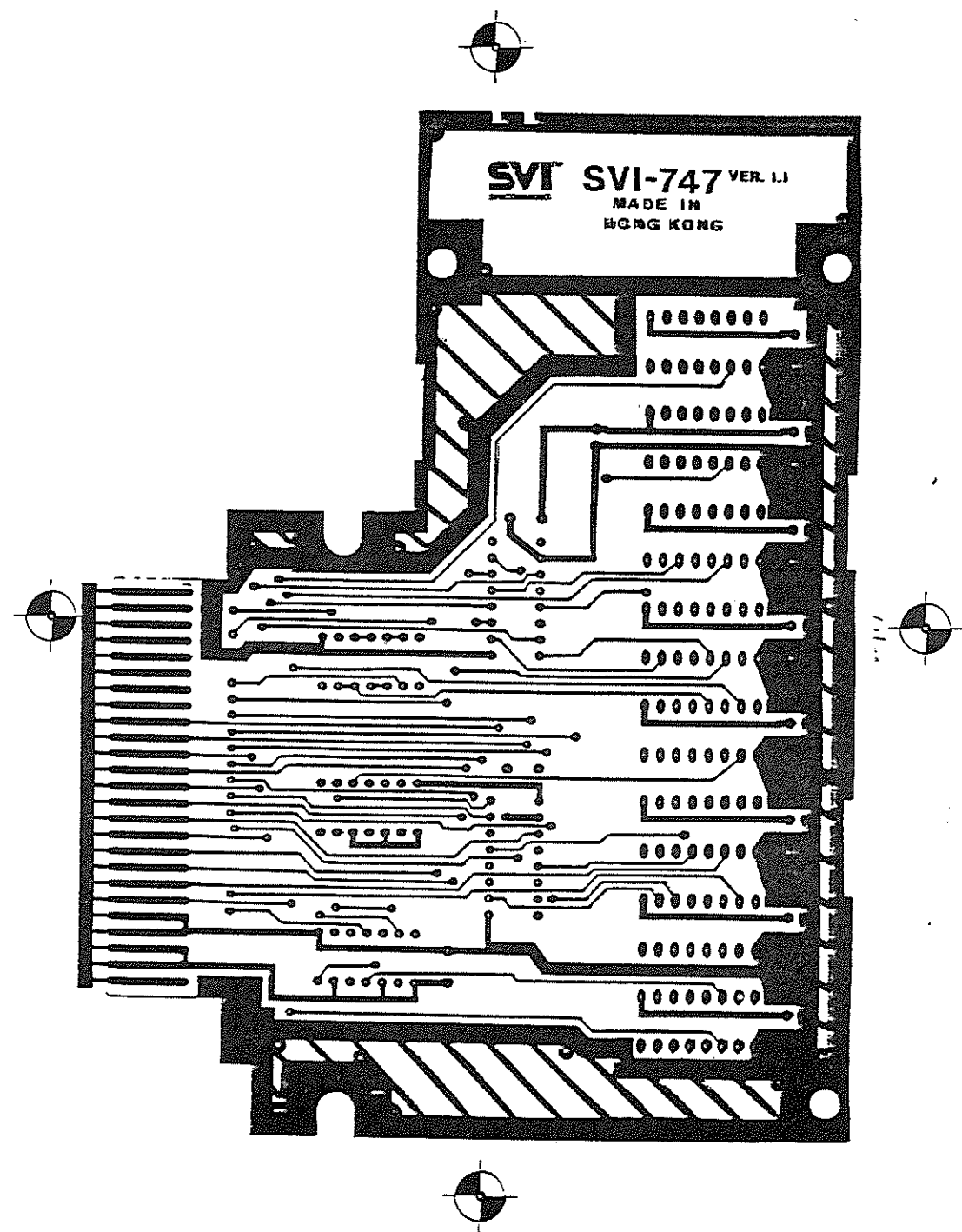
E

D

C

B

A



CONFIDENTIAL  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

7

6

5

4

3

2

1

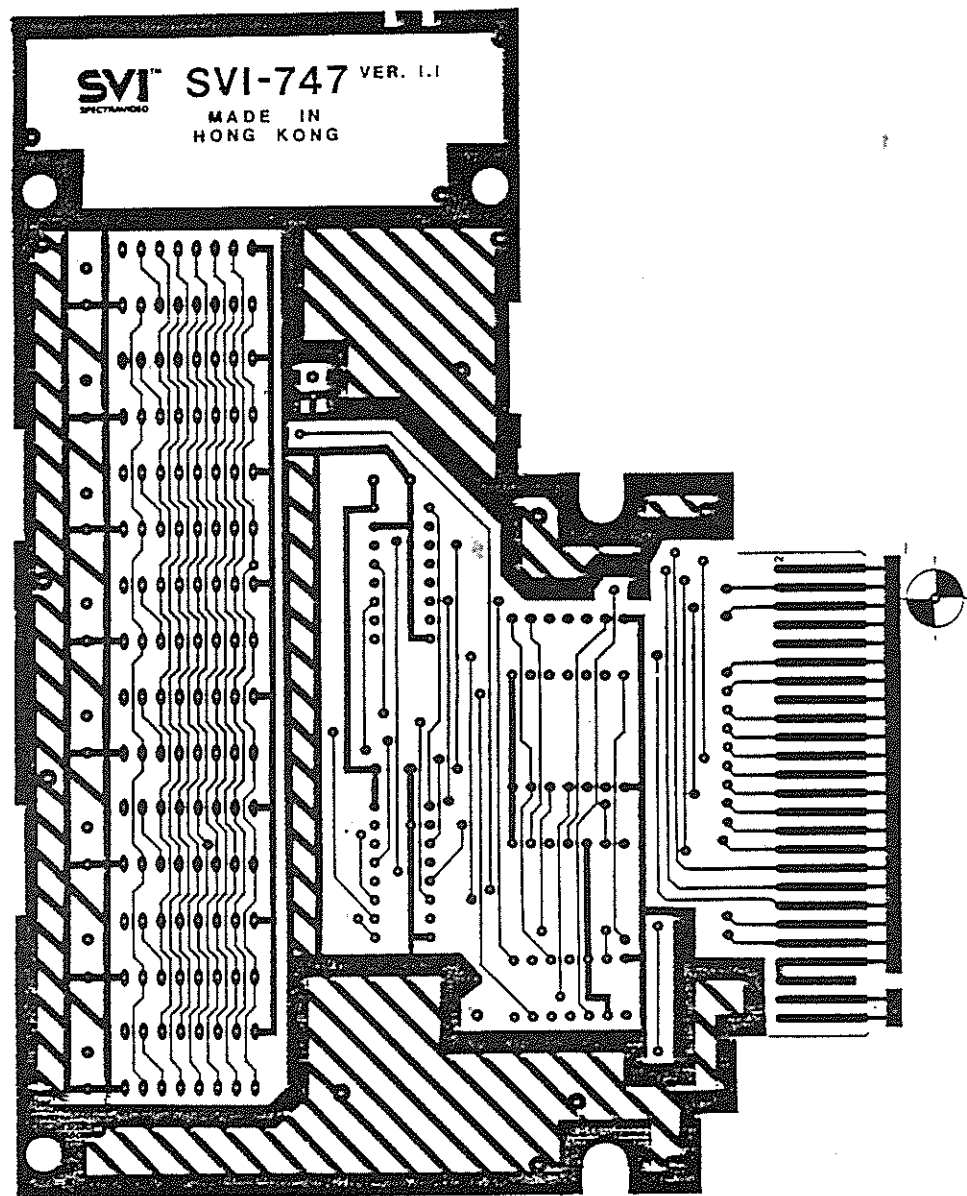
E

D

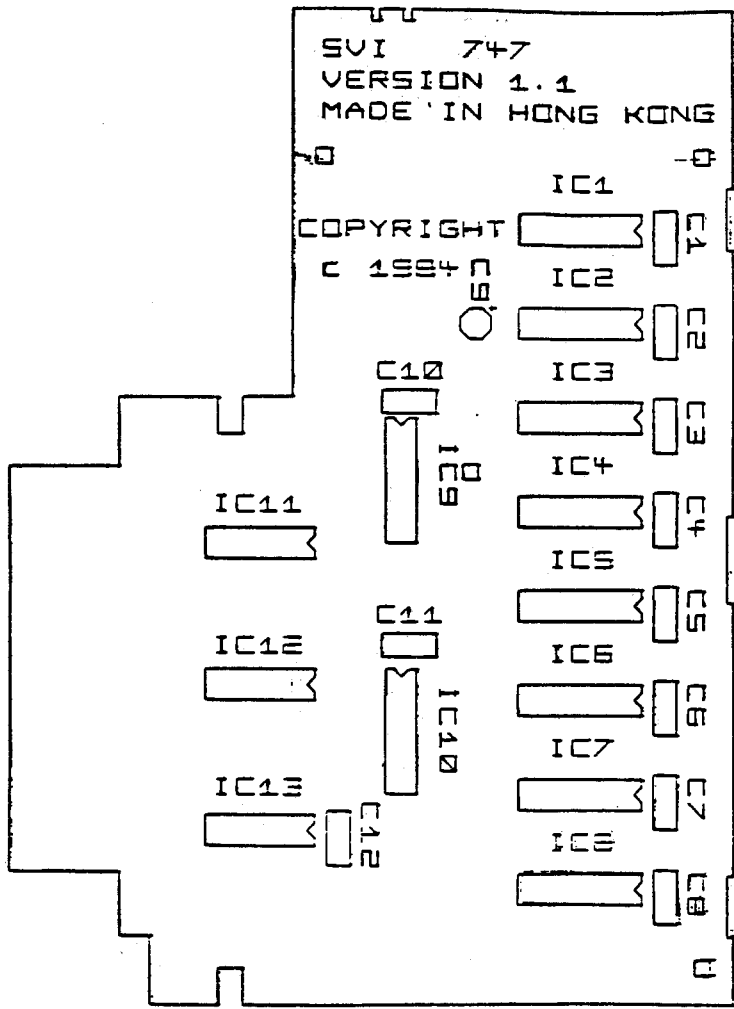
C

B

A

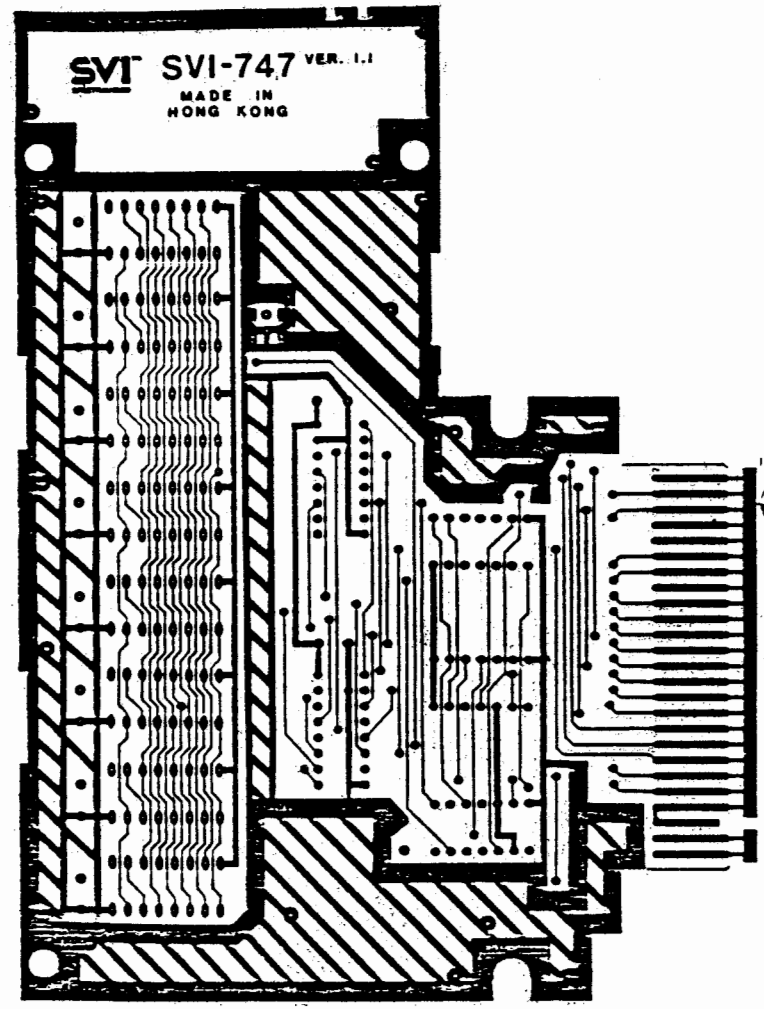
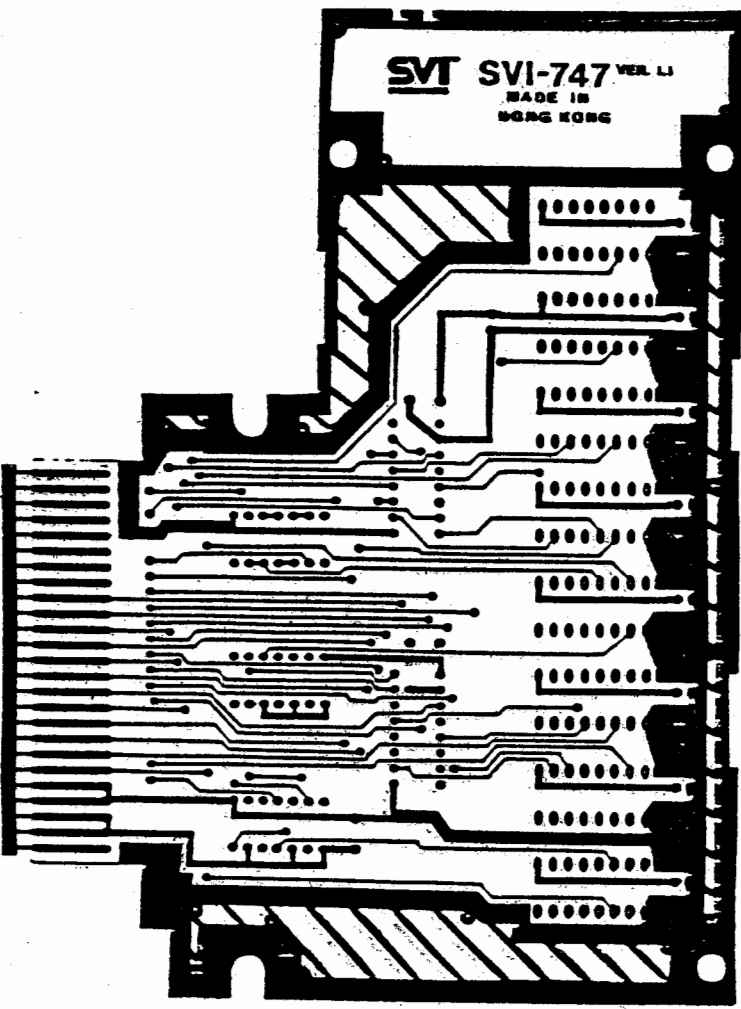


<b>SVI SPECTRAVIDEO INTERNATIONAL LIMITED</b>	
<b>CONTROLLED ENGINEERING DOCUMENT</b>	
THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE - SVI - 747 PCB LAYOUT (COMPONENT AND SOLDER SIDE)	VERIFIED BY <u>KIMBA LAU</u> DATE <u>27th Dec 1984</u>
DOCUMENT NO <u>STM-747-002</u>	VERSION NO <u>1.1</u> SHEET <u>1</u> OF <u>1</u>



**CONFIDENTIAL**  
 THE DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.		
TITLE - SVI-747 COMPONENT DRAWING	VERIFIED BY KIMBA LAU	DATE 27 <sup>th</sup> DEC. 1984
DOCUMENT NO STM-747-001	VERSION NO 1.1	SHEET 1 OF 3



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI SPECTRAVIDEO INTERNATIONAL LIMITED</b>	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD..	
TITLE - SVI - 747 PCB LAYOUT (COMPONENT AND SOLDER SIDE)	VERIFIED BY - <u>KIMBA LAU</u> DATE <u>27<sup>th</sup> Dec 1984</u>
DOCUMENT NO <u>STM-747-002</u>	VERSION NO <u>1.1</u>
	SHEET <u>2</u> OF <u>3</u>

(5) SVI-757 MSX RS-232 INTERFACE CARTRIDGE

SVI-757 MSX RS-232 Interface Cartridge provides Electronic Industrial Association (EIA) standard for data communication between your computer and external equipment.

With the installed cartridge, your computer can link up with another computer, modem, teletype equipment or printer etc. Since the cartridge is MSX compatible, you can connect it to the SVI-728 or other MSX computer.

5.1 SPECIFICATION

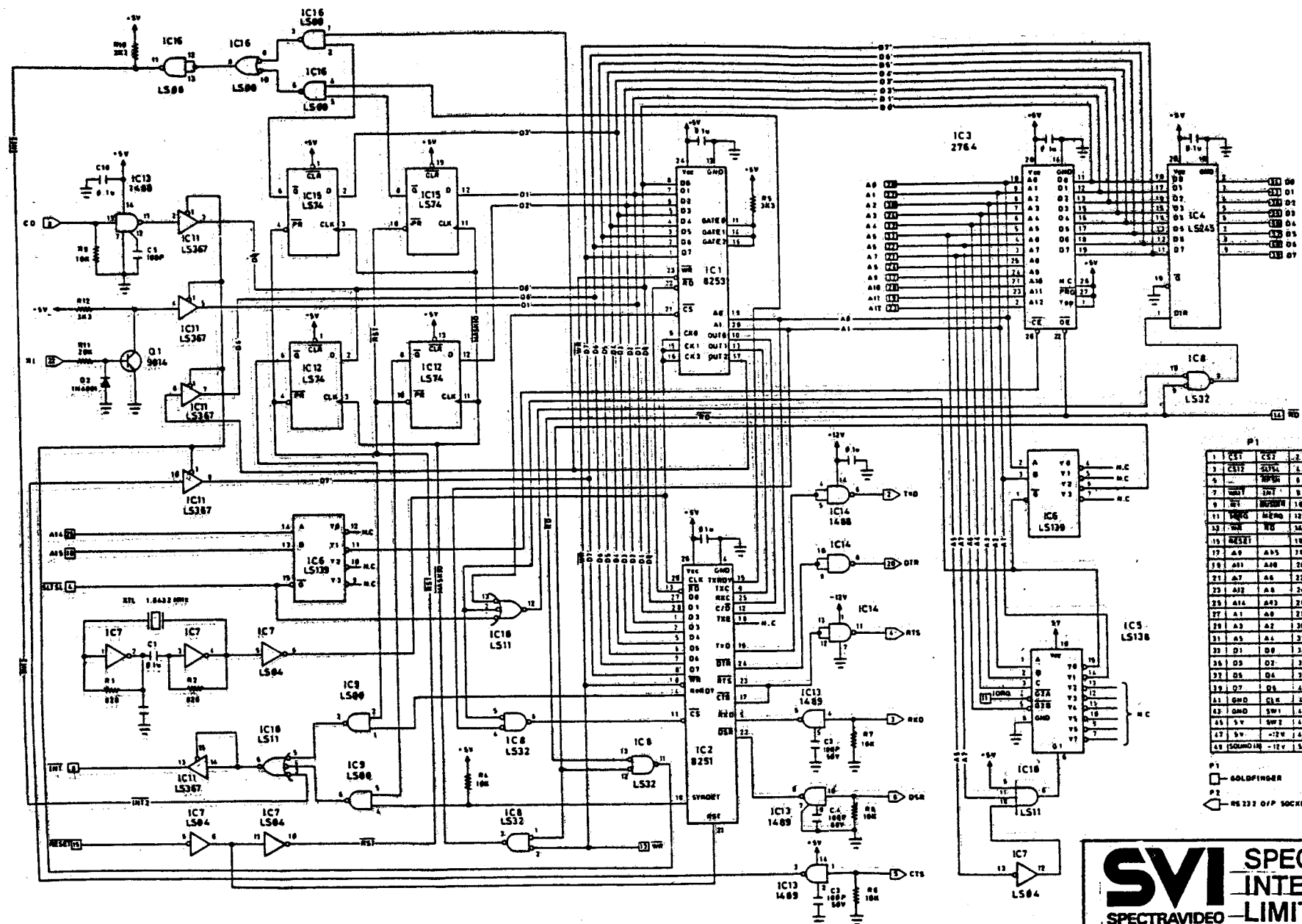
Baud Rate	:	50 - 19200 bps (15 different baud rate)
Counter Channel Usage	:	CH0 - receive baud rate clock  CH1 - transmit baud rate clock  CH2 - used by application (e.g. time out counter)
Power Consumption	:	+ 5V 250mA + 12V 25mA - 12V 28mA
Dimensions (mm)	:	170(L) x 123(W) x 32(D)

5.2 SPARE PART LISTGENERAL RECOMMENDED SPARE PART LIST FOR SVI-757 MSX  
RS-232C INTERFACE CARTRIDGE

<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
DD757101	5 1/4" BLANK DISK 48TPI	1.40
IC1488	IC 1488	0.40
IC1489	IC 1489	0.40
IC8251A	8251A USART	1.80
IC8253	8253A PROGRAMMABLE TIMER	1.80
ID0001	IN4148:SILICON DIODE	0.10
IL00LS	74LS00	0.20
IL04LS	74LS04	0.20
IL11LS	74 LS11	0.20
IL138LS	74LS138	0.30
IL139LS	74LS139	0.30
IL245LS	74LS245	0.50
IL32LS	74LS32	0.20
IL367LS	74LS367	0.30
IL74LS	74LS74	0.20
IM27643	2764 EPRON 300NS	3.40
IT0010	TRANSISTOR 9014C	0.10
KE3010025	ELECT CAP 10UF 25V	0.10
KR1100050	100PF 50V NPO CAP.	0.10
KR2001050	1000PF CERAMIC CAP 50V	0.10
KR2100016	0.1UF 16V CERAM CAP	0.10
MC3200184	1.8432 MHZ X'TAL	0.80
MP242	24 PINS IC SOCKET	0.10
MP252-1	RS232 SOCKET	1.50
MP282	28 PIN IC SOCKET	0.20
PC747101	EXPORT CARTON	0.10
PG757101	G/B FOR MSX RS232 CARD	0.30
PI318104(S)	GUARANTEE CARD (SVI)	0.10
PI757101	MSX RS232 CARD MANUAL	0.50
PI757102	RS232 SOFTWARE MANUAL	0.40
PP727101	POLYFOAM FOR MSX CARD	0.20
PP727102	POLYFOAM BOTTOM	0.20
PZ638002	MODEM PVC BAG	0.10
RF1823	820 OHM RSTR 1/4W 5%	0.10
RF2333	3K3 1/4W 5% RESISTOR	0.10
RF3103	10KOHM 1/4W +/-5%	0.10
RF3203	20KOHM 1/4W +/-5%	0.10
SP75701-01	MSX RS232 CARD PCB	3.00
WA003	QC LABEL	0.10
WA019	QC LABEL (WHITE)	0.10
WA088	FCC LABEL	0.10
WA120	SERIAL NO LABEL	0.10
WA126	CP/M SYSTEM DISK LABEL	0.10
WN727002	REAR COVER	0.30
WN757001	MSX RS232 CARD FRONT COV	0.30
XS050	TS/PH SCREW D3X8	0.10
XS054	MS/PH SCREW M3X0.5PX8	0.10
	TOTAL:	21.40

### 5.3 CIRCUIT DIAGRAM

- SVI-757 MSX RS232 Interface Cartridge (Schematic Diagram)
- SVI-757 MSX RS232 Interface Cartridge PCB Layout (Component Side)
- SVI-757 MSX RS232 Interface Cartridge PCB Layout (Solder Side)
- SVI-757 MSX RS232 Interface Cartridge Component Layout



P1

1	CE1	CE2	A
2	CE1	CE2	A
3	...	...	...
4	...	...	...
5	...	...	...
6	...	...	...
7	...	...	...
8	...	...	...
9	...	...	...
10	...	...	...
11	...	...	...
12	...	...	...
13	...	...	...
14	...	...	...
15	...	...	...
16	...	...	...
17	...	...	...
18	...	...	...
19	...	...	...
20	...	...	...
21	...	...	...
22	...	...	...
23	...	...	...
24	...	...	...
25	...	...	...
26	...	...	...
27	...	...	...
28	...	...	...
29	...	...	...
30	...	...	...
31	...	...	...
32	...	...	...
33	...	...	...
34	...	...	...
35	...	...	...
36	...	...	...
37	...	...	...
38	...	...	...
39	...	...	...
40	...	...	...
41	...	...	...
42	...	...	...
43	...	...	...
44	...	...	...
45	...	...	...
46	...	...	...
47	...	...	...
48	...	...	...

P1  
 □ GOLDPINGER  
 P2  
 ◁ RS 232 O/P SOCKET

**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

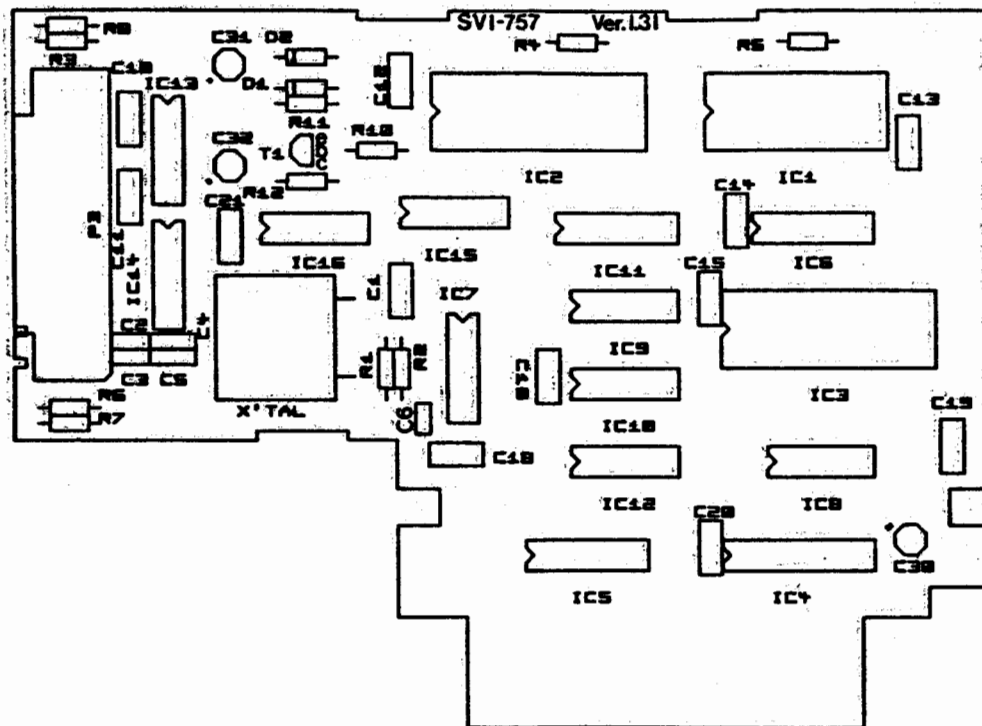
TITLE—SVI-757 MSX  
 RS232C INTERFACE CARTRIDGE  
 (SCHEMATIC DIA.)

VERIFIED BY KIMBA LAU  
 DATE -7-85

DOCUMENT NO STM-757-001

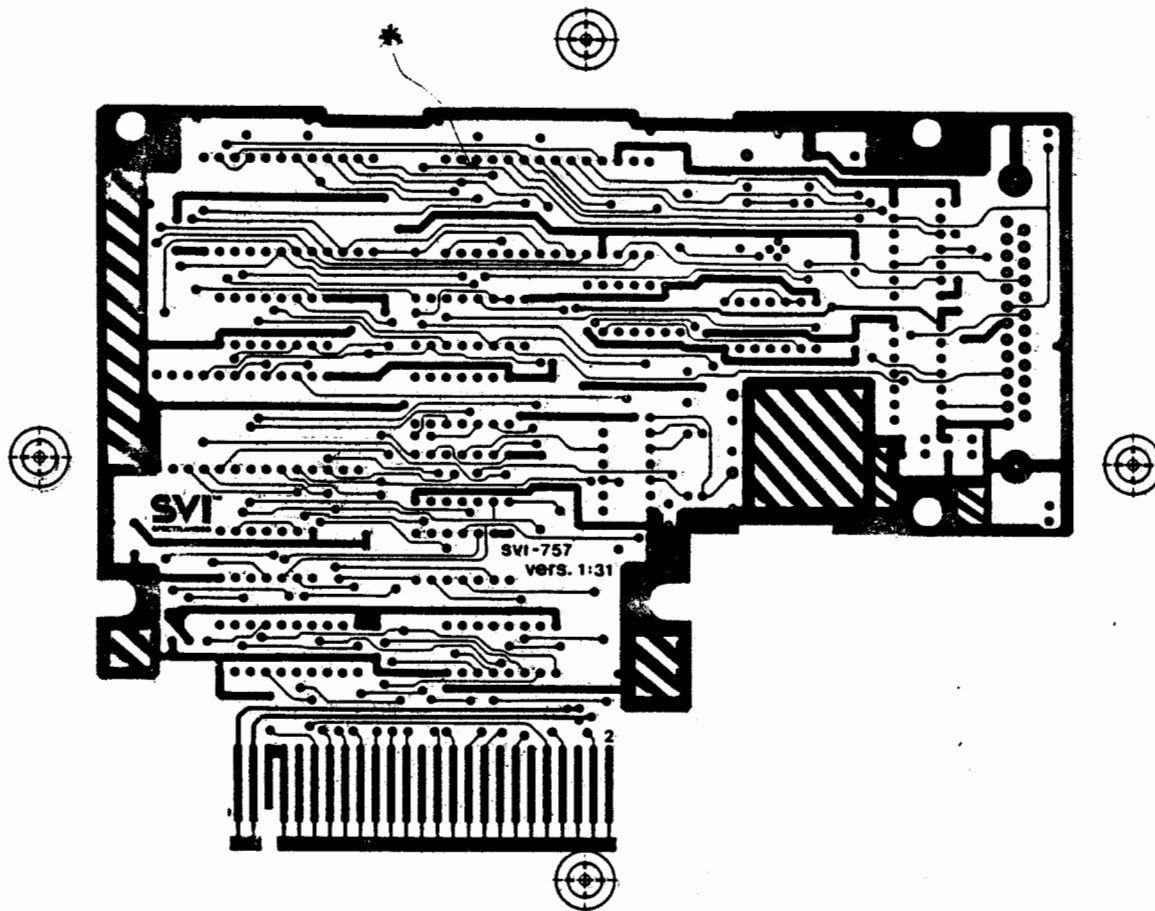
VERSION SHEET 1  
 NO 1.0 OF 4





CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI SPECTRAVIDEO INTERNATIONAL</b> <b>SPECTRAVIDEO LIMITED</b>	
<b>CONTROLLED ENGINEERING DOCUMENT</b> THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE — SVI-757 RS232C INTERFACE CARTRIDGE COMPONENT LAYOUT	VERIFIED BY: <u>KIMBA LAU</u> DATE: <u>18-7-1985</u>
DOCUMENT NO. <u>STM-757-004</u>	VERSION NO. <u>1</u> SHEET <u>4</u> OF <u>4</u>



- NOTE: (1) \* Cut trace at  
I.C. 2 Pin 17
- (2) Connect a jumper  
wire between IC2  
pin 17 to pin 23

CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
INTERNATIONAL  
SPECTRAVIDEO LIMITED

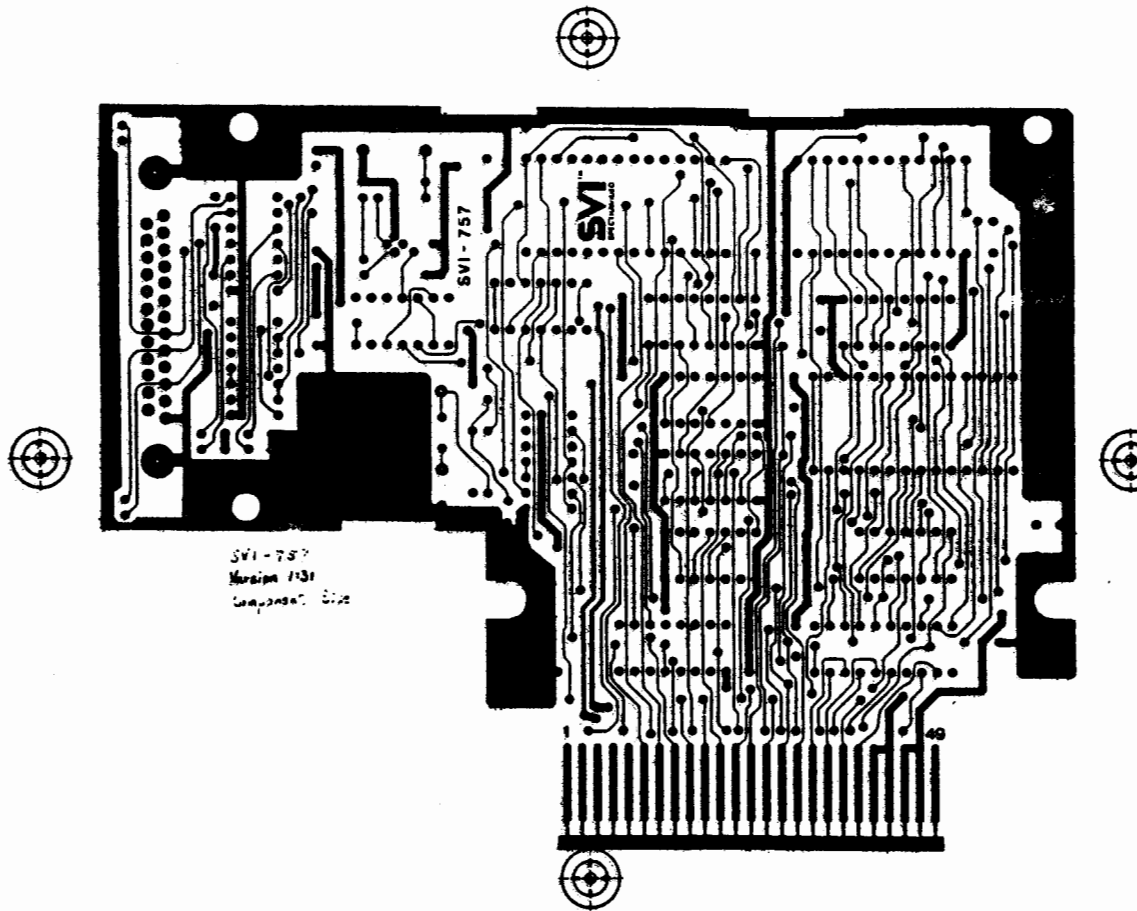
**CONTROLLED ENGINEERING DOCUMENT**  
THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE — SVI-757 RS232C  
INTERFACE CARTRIDGE  
PCB LAYOUT (SOLDER SIDE)

VERIFIED BY KIMBA LAU  
DATE 8-7-1985

DOCUMENT NO STM-757-002

VERSION SHEET 2  
NO 1.31 OF 4



SVI-757  
Revision 1/81  
Component Side

CONFIDENTIAL  
THIS DOCUMENT CONTAINS  
CONFIDENTIAL INFORMATION  
WHICH IS THE EXCLUSIVE  
PROPERTY OF SPECTRAVIDEO  
INTERNATIONAL LIMITED

**SVI** SPECTRAVIDEO  
INTERNATIONAL  
SPECTRAVIDEO LIMITED

**CONTROLLED ENGINEERING DOCUMENT**  
THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS  
THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE — SVI-757 RS232C INTERFACE CARTRIDGE PCB LAYOUT (COMPONENT SIDE)	VERIFIED BY <u>KIMBA LAU</u> DATE <u>-7-1981</u>
--	--

DOCUMENT NO. <u>CTM-757-003</u>	VERSION NO. <u>1.31</u>	SHEET <u>3</u> OF <u>4</u>
------------------------------------	----------------------------	-------------------------------

(6) SVI-767 MSX DATA CASSETTE

The SVI-767 is an inexpensive device that uses ordinary cassette tape to add storage and retrieval capability to the computer system. The built-in Automatic Level Control (ALC) ensures reliable and high quality recording. Tape counter, LED indicator and auto-stop are standard features.

Specifications for product improvement are subject to change without notice.

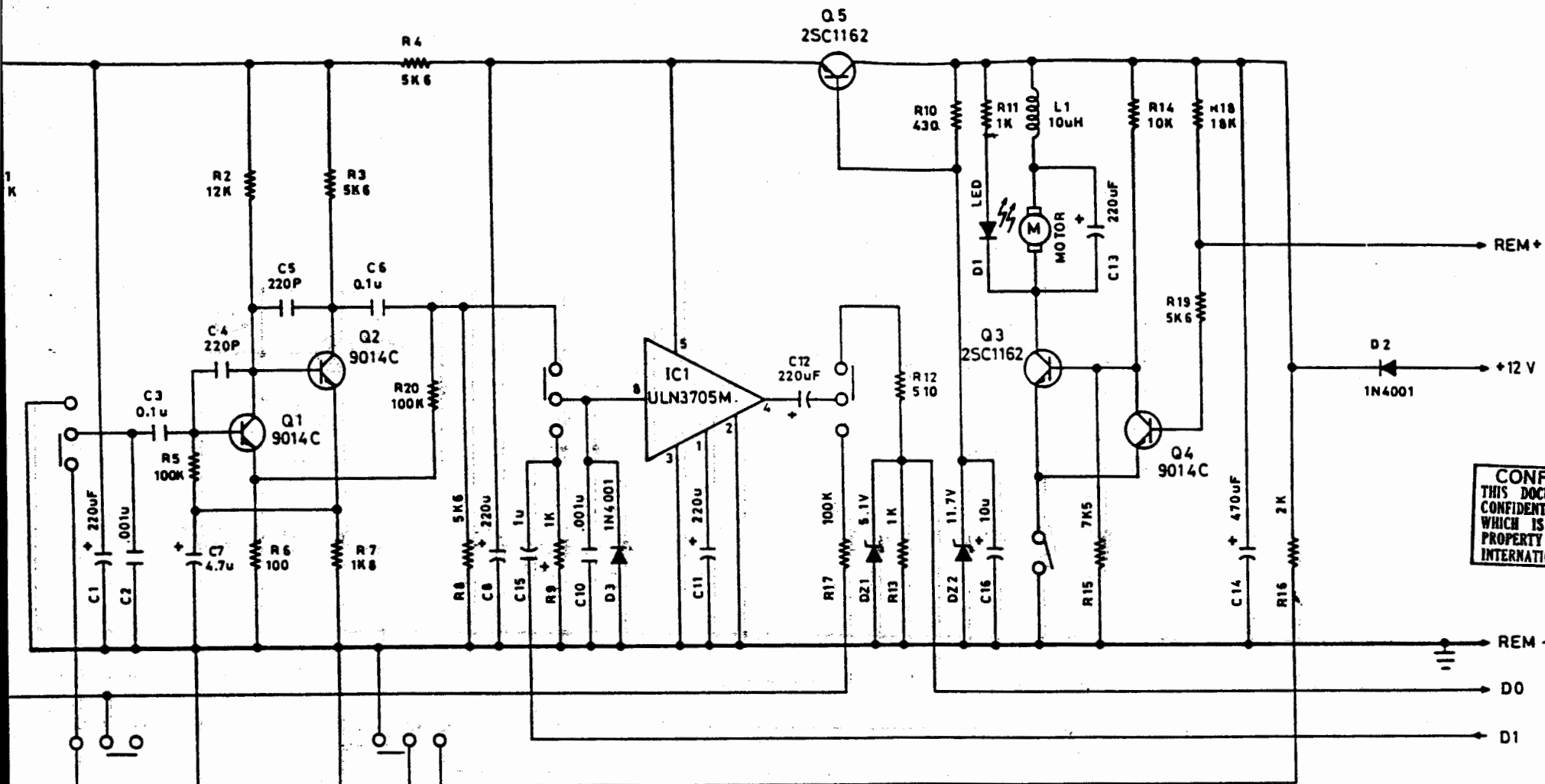
6.1 SPARE PART LISTGENERAL RECOMMENDED SPARE PART LIST FOR SVI-767 MSX  
DATA CASSETTE

<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
MM00001	MOISTURE ABSORBANT (5GM)	0.10
PC767101-01	OUTC 475X430X630MM F767	0.20
PG767101	G/B F MONO CASSETTE	0.50
PI31807	SVI WARRANTY	0.10
PI767101	INSTRUCTION MANUAL F767	0.30
PP70701	POLYFOAM F MSX DISK DRIVE	0.90
PZ01001	PLASTIC BAG 9.5X16.5 IN	0.10
WP76701	FCC LABEL	0.10
WP76703	S/NO LABEL 767	0.10
WP76703G	S/NO LABEL F G/B	0.10
DL010	LED 2 X 5 (MR62D)	0.10
MJ0378	D2X140MMS/SHID WIRE (BLK)	0.10
MJ0370	D2X140MMS/SHID WIRE GREY	0.10
MJ040	2-CORES WIRE 120MM	0.10
MJ044	2-CORES WIRE 150MM	0.10
MJ050	J WIRE DO.8X70MM YELLOW	0.10
MJ115	UL1007 #24 RED 180MM	0.10
MJ118	UL1007 #24 BLACK 180MM	0.10
MQ76701	DC POWER JACK	0.10
MS90303	PUSH SWITCH 6X2X1 L:7MM	0.20
TC0001	10 H CHOKE COIL	0.10
WN60107	CABLE TIE 70MM	0.10
WN86319	PLASTIC TUBE DAI 1 MM	0.10
WN90205	INLAY (LED)	0.10
WN90302	CABINET BOTTOM	0.70
WN90303	CASSETTE DOOR	0.20
WN90304	KNOB "REC"	0.10
WN90305	KNOB "REWIND"	0.10
WN90306	KNOB "F.FORWARD"	0.10
WN90307	KNOB "PLAY"	0.10
WN90308	KNOB "STOP"	0.10
WN90311	BELT SQ 0.8 X D38	0.10
WN90313S	PUSHING INNER DAI 6	0.10
WN90314	CABLE CLAMP	0.10
WN90401	CABINET TOP	0.60
WP76702	SVI-767 CASSETTE BACK L	0.10
WS90401	SHIELD PAPER 120X45MM	0.10
XC60104	EYELET RING (10014-6)	0.10
XC60513	I.T. WASHER M3	0.10
XC60517	M3X8 FLAT WASHER	0.10
XC76701	CASSETTE NAME PLATE F 767	0.10
XC90301	COUNTER BRACKET	0.10
XC90302	SPRING-PLATE DOOR	0.10
XC90303	REC PUSH-PLATE	0.10

GENERAL RECOMMENDED SPARE PART LIST FOR SVI-767 MSX  
DATA CASSETTE

<u>PART NO.</u>	<u>DESCRIPTION</u>	<u>FOB HONG KONG (US DOLLAR)</u>
XC90309	WASHER M3X10	0.10
XC90401	FERRITE RING	0.20
XS010	M 3 X 16 PA	0.10
XS032	SCREW D3X8PA	0.10
XS057	M 3 X PO.5 X 6 PM	0.10
XS059	M2.6X5.0 TTP	0.10
XS062	D3 X 10PT	0.10
XS065	M2X5 TTP SCREW	0.10
IC3705	ICULN 3705N SPRAGUE B	0.40
ID0012	DIODE IN4001	0.10
IT0011	TRANSISTOR 9014C	0.10
IT0013	TRANSISTOR 25C1162	0.20
IZ0006	4.9V ZENER 1/4W	0.10
IZ0007	ZENER 12V 1/2W	0.10
KE3001050	ELECT CAP 1UF 50V	0.10
KE3010025	ELECT CAP 10UF 25V	0.10
KE3220016	ELECT CAP 220UF 16V	0.10
KE3470016	ELECT CAP 470UF 16V	0.10
KM2001050	MYLAR CAP 0.001UF 50V	0.10
KR1220050	CERAMIC CAP 220PF	0.10
KR2100012	CERAMIC CAP 0.1UF 12V	0.10
KT2470016	TAN. CAP. 0.47UF 16V	0.10
RF1102	RESIST 100 OHM 1/4W +/-5%	0.10
RF1472	RESIST.470 OHM 1/4W +/-5%	0.10
RF1512	RESIST. 510 OHM 1/4W +/-5%	0.10
RF2102	RESISTOR 1K 1/4W +/-5%	0.10
RF2182	RESIST 1.8K OHM 1/4W +/-5%	0.10
RF2202	RESIST 2K 1/4W +/-5%	0.10
RF2562	RESIST 5.6K OHM 1/4W +/-5%	0.10
RF2752	RESIST 7.5K 1/4W +/-5%	0.10
RF3102	RESISTOR 10K 1/4W +/-5%	0.10
RF3122	RESIST.12K OHM 1/4W +/-5%	0.10
RF3182	RESISTOR 18K	0.10
RF3272	RESIST 27K 1/4W +/-5%	0.10
RF4102	RESIST 100K OHM 1/4W 5%	0.10
SP767-03	SVI-767 PCB VER 1.2	0.20
YD209	MSX767 CASSETTE I/O CABL	0.90
YS90301	COUNTER MA-M2-13K-6	0.60
YS90303	DECK MECH.1000H MONOHEAD	6.00
*YS76701	DC12V 200MA ADPT220V VDE	2.30
	TOTAL:	21.20

\* MAY HAVE DIFFERENT VERSIONS FOR DIFFERENT COUNTRIES

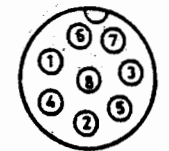


**CONFIDENTIAL**  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

HEAD

E. HEAD

- 4 - D1
- 5 - D0
- 6 - REM -
- 7 - REM -
- 1, 2, 3, 8 - GND

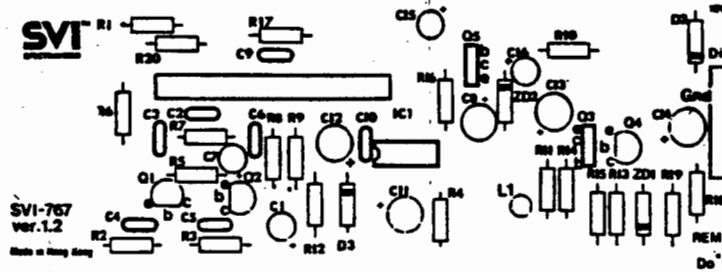
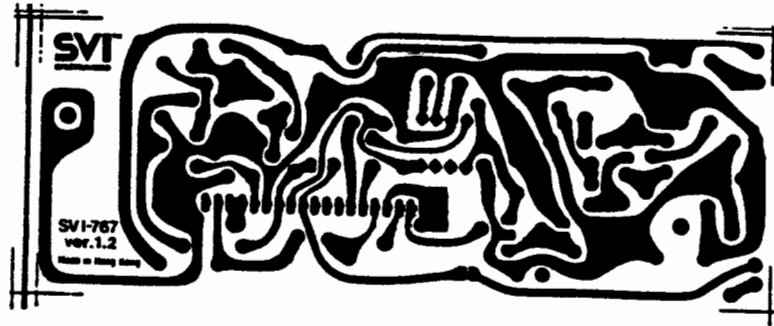


CASSETTE PLUG

**SVI SPECTRAVIDEO INTERNATIONAL LIMITED**

**CONTROLLED ENGINEERING DOCUMENT**  
 THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.

TITLE SVI-767 MSX DATA CASSETTE (SCHEMATIC DIA)	VERIFIED BY KIMBA LAU
	DATE 1-7-85
DOCUMENT NO. 767-001	VERSION NO. 1.2
	SHEET 1 OF 1



CONFIDENTIAL  
 THIS DOCUMENT CONTAINS  
 CONFIDENTIAL INFORMATION  
 WHICH IS THE EXCLUSIVE  
 PROPERTY OF SPECTRAVIDEO  
 INTERNATIONAL LIMITED

<b>SVI</b> SPECTRAVIDEO INTERNATIONAL SPECTRAVIDEO LIMITED	
CONTROLLED ENGINEERING DOCUMENT THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION WHICH IS THE EXCLUSIVE PROPERTY OF SPECTRAVIDEO INTERNATIONAL LTD.	
TITLE— SVI-767 MSX DATA CASSETTE PCB & COMPONENT LAYOUT	VERIFIED BY— KIMBA LAU DATE— 2A-7-85
DOCUMENT NO. STM-767-002	VERSION SHEET NO. 1.2 OF 1