

Unpublished work Copyright 2002-2004 Charles MacDonald

Table of Contents

- 1.) Introduction
- 2.) Overview
- 3.) Memory map
- 4.) Port map
- 5.) Keyboard and I/O
- 6.) TMS9929A information
- 7.) Miscellaneous
- 8.) Assistance Needed
- 9.) Credits and acknowledgements
- 10.) Disclaimer

What's New:

[07/29/04]

- Updated memory refresh information.
- Added chip revision information.
- Added power supply and A/V cable notes.

[03/04/04]

- Added cassette custom chip pinout
- Fixed error labeling PPI pins PC3 and PC4.
- Finished MITEC-2 pinout
- Confirmed unused pins in several connectors
- Added information on Z80 / SN76489A clock generation
- Changed /EXCS to be /EXM1 to fit naming conventions

[03/01/04]

- Added details on DRAM interface. Some guesswork, but mostly accurate.
- Added almost complete MITEC-2 custom chip pinout.
- Finished cartridge connector pin assignments
- Added keyboard ribbon cable pin assignments
- Added serial printer port pin assignments
- Fixed joystick pin assignments
- Expanded coverage of cart types
- Fixed several errors

[06/05/02]

- Added notes on PPI control register and work RAM
- Fixed initial values for PPI port B bits 7-5
- Added information on VDP register #1 bit 7

[05/31/02]

- Updated keyboard information
- Added interrupt information

1.) Introduction

The SC-3000H is a personal computer made by Sega. For information about Sega's 8-bit computers and game systems that came before the Mark III, check out the documentation for MEKA:

<http://www.smspower.org/meka/>

To briefly sum things up, here's a list of the different systems:

SG-1000	No keyboard
SG-1000 II	No keyboard, but supports the SK-1100 external keyboard
SC-3000	Built-in keyboard with soft rubber keys
SC-3000H	Built-in keyboard with hard plastic keys

Expansion hardware

An addition for the SC-3000 was the SF-7000, which adds 64K of RAM, 8K of ROM (for a disk operating system), Centronics parallel port, and an

RS-232C serial port. Marc Le Dourain has a webpage about it at:

<http://www.multimania.com/mavati>

2.) Overview

The SC-3000H has the following components:

Manufacturer	Part	Description
Fujitsu	MB82128-15	2Kx8 SRAM (Z80 work RAM)
Fujitsu	MB74LS145	1 of 10 decoder (keyboard row select)
Mitsubishi	MN74HC04	Hex inverter (timing related)
Motorola	MCM4517P15 (x8)	16Kx1 Page Mode DRAM (TMS9929A VRAM)
NEC	uPD9255AC-2	Programmable peripheral interface
SEGA	MITEC-2	Custom chip (Z80 address decoding)
Sharp	LH0080A	CPU (Z80-A compatible)
Texas Instruments	TMS SN76489AN	Programmable sound generator
Texas Instruments	TMS 9929ANL	Video display controller (PAL)

Note that both the SN76489A and TMS9929A are the 'A' versions, as opposed to the original chips which have no letter suffix.

The TMS9929 has no Graphics II mode, this was added in the TMS9929A.
The SN76489 noise generator works differently compared to the SN76489A.

The main clock runs at 10.738635 MHz, which is used directly by the TMS9929A. The hex inverter, along with some analog components, is used to divide the main clock by 3 to generate the Z80 and SN76489A clock of 3.579545 MHz.

It also has several connectors on the back and side panels:

- Two joystick connectors (male DB9)
- Audio / video output (female 5-pin DIN)
- Printer interface (female 7-pin DIN)
- Cassette in and out jacks (2 female mono miniplug jacks)
- 44-pin cartridge connector

Pin assignments

=====

Joystick connectors

=====

Pin diagram of male DB9 plug on back of console:

1 2 3 4 5
6 7 8 9

- 1 : Joystick up
- 2 : Joystick down
- 3 : Joystick left
- 4 : Joystick right
- 5 : Unused (not connected to anything)
- 6 : Trigger left
- 7 : Unused (not connected to anything)
- 8 : Common
- 9 : Trigger right

Pin 8 is the common side of all switches in the joystick and is connected to output line 7 of the 74LS145. It is LOW when the seventh keyboard row is selected (enabling the joystick ports for input) and HIGH otherwise (joystick ports disabled).

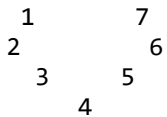
This is different from the SMS and Genesis/MegaDrive which use pin 8 as ground and pin 5 as +5V. The original type of Master System joypads are compatible with the SC-3000H, but I would not advise using any special pads (autofire or programmable) or any Genesis/MegaDrive pads.

=====

Printer connector

=====

Pin diagram of female 7-pin DIN plug on back of console:



- 1 : Unused (not connected to anything)
- 2 : PPI PC5 (DATA output)
- 3 : PPI PC7 (/FEED output)
- 4 : PPI PB6 (BUSY input)
- 5 : PPI PC6 (/RESET output)
- 6 : PPI PB5 (FAULT input)
- 7 : GND

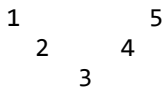
The pin names are from the SF-7000 documentation, not my own.

=====

Audio / Video connector

=====

Pin diagram of female 5-pin DIN plug on back of console:



- 1: Composite video
- 2: Ground
- 3: Ground
- 4: Ground
- 5: Ground
- 6: Monoaural audio

=====

Internal keyboard ribbon cable headers

=====

Pin numbering goes from left (1) to right (11) for both connectors.

CN7

- 1 : 74LS145 output line 0
- 2 : 74LS145 output line 1
- 3 : 74LS145 output line 2
- 4 : 74LS145 output line 3
- 5 : 74LS145 output line 4
- 6 : 74LS145 output line 5
- 7 : 74LS145 output line 6
- 8 : PPI PA0
- 9 : PPI PA1
- 10 : PPI PA2
- 11 : PPI PA3

CN8

- 1 : PPI PA4
- 2 : PPI PA5
- 3 : PPI PA6
- 4 : PPI PA7
- 5 : PPI PB0
- 6 : PPI PB1
- 7 : PPI PB2
- 8 : PPI PB3
- 9 : Ground

10 : Not used (see notes)
11 : From RESET key to MITEC-2 pin 3 (NMI-IN)

Pins 1-7 of CN7 are outputs to select rows in the keyboard's key matrix.

Pins 8-11 of CN7 and 1-8 of CN8 are inputs with pull-up resistors to read data from the selected keyboard row.

Pin 10 is not used by the keyboard. It is connected to the power switch and could have been intended to control the power LED, which instead has it's own direct connection to the switch.

Pin 11 of CN8 is an output to the NMI-IN pin of the MITEC-2 chip.

=====

Cartridge connector pin assignments

=====

Solder side

A01 : Z80 A0
A02 : Z80 A1
A03 : Z80 A2
A04 : Z80 A3
A05 : Z80 A4
A06 : Z80 A5
A07 : Z80 A6
A08 : Z80 A7
A09 : Z80 A8
A10 : Z80 A9
A11 : Z80 A10
A12 : Z80 A11
A13 : Z80 A12
A14 : Z80 A13
A15 : Z80 D0
A16 : Z80 D1
A17 : Z80 D2
A18 : Z80 D3
A19 : Z80 D4
A20 : Z80 D5
A21 : Z80 D6
A22 : Z80 D7

Components side

B01 : +5V
B02 : +5V
B03 : Work RAM /CS
B04 : /EXM1
B05 : MEMORY /RD
B06 : MEMORY /WR
B07 : I/O PORT /RD
B08 : I/O PORT /WR
B09 : Unused (not connected to anything)
B10 : Z80 /MREQ
B11 : /CONT
B12 : /RAS0
B13 : /CAS0
B14 : CA7
B15 : /RAS1
B16 : /CAS1
B17 : /RCSEL
B18 : Z80 A14
B19 : Z80 A15
B20 : Unused (not connected to anything)
B21 : GND
B22 : GND

The work RAM /CS signal is connected through a resistor to pin B03 of the

cartridge connector and the /CS pin of work RAM. If pin B03 is left unconnected then work RAM is enabled for \$C000-\$FFFF, if tied to +5V then work RAM is disabled, freeing up \$C000-\$FFFF for on-cart hardware to use.

Pin B04 appears to be a chip select signal for memory accesses to the \$8000-\$BFFF range.

Pins B05 and B06 are /MREQ and /RD,/WR combined for memory read/write strobes.

Pins B07 and B08 are /IORQ and /RD,/WR combined for I/O read/write strobes.

Pin B11 is connected to PPI PB4 through a pull-up resistor. The SF-7000 documentation labels this pin /CONT, though it's just another input bit.

Pins B12 to B17 are used for controlling DRAM, managed by the MITEC-2 chip.

3.) Memory map and cartridge types

The memory map is mostly defined by the cartridge. As mentioned earlier the MITEC-2 chip does provide a /CS signal for the \$8000-\$BFFF and \$C000-\$FFFF range, but it's up to the cartridge how to map ROM, RAM, etc. to memory.

For any memory area that is unused, such as the work RAM area when it is disabled, reads return the high byte of the address bus (e.g. reading \$C1AB would return \$C1).

I only have a Safari Hunting and Basic Level III A cartridge, so some of the following information may not apply to other cartridge types.

Safari Hunting

This cartridge has two sockets, one for a 16K ROM and another for a 2K RAM. The Safari Hunting game only uses the ROM and leaves the RAM socket empty. Here are the pin assignments:

16K ROM (28-pin DIP)

+---\ /---+			
VCC	01	28	VCC
A12	02	27	VCC
A7	03	26	A13
A6	04	25	A8
A5	05	24	A9
A4	06	23	A11
A3	07	22	/OE (From Z80 A15)
A2	08	21	A10
A1	09	20	/CS (from Z80 /MREQ)
A0	10	19	D7
D0	11	18	D6
D1	12	17	D5
D2	13	16	D4
GND	14	15	D3
+-----+			

2Kx8 SRAM (24-pin DIP)

+---\ /---+			
A7	01	24	VCC
A6	02	23	A8
A5	03	22	A9
A4	04	21	/WR (From pin B06, memory /WR)
A3	05	20	/OE (From pin B05, memory /RD)
A2	06	19	A10
A1	07	18	/CS (From pin B04, \$8000-\$BFFF /EXM1)
A0	08	17	D7
D0	09	16	D6
D1	10	15	D5
D2	11	14	D4

```

GND |12    13| D3
+-----+

```

This gives the following memory map:

```

$0000-$3FFF : ROM
$4000-$7FFF : ROM (mirror)
$8000-$BFFF : RAM (mirrored repeatedly every 2K)
$C000-$FFFF : Work RAM (mirrored repeatedly every 2K)

```

Basic Level III A (PAL)

This cartridge has a 32K ROM and four 16Kx4 DRAM sockets. Two of them have TMS4416 16Kx4 DRAMs, the other two are empty. This would allow up to 32K total, but only 16K is present.

32K ROM (28-pin DIP)

```

+-----+
VCC |01    28| VCC
A12 |02    27| A14
A7  |03    26| A13
A6  |04    25| A8
A5  |05    24| A9
A4  |06    23| A11
A3  |07    22| /OE (From pin B05, memory /RD)
A2  |08    21| A10
A1  |09    20| /CS (From 74LS32)
A0  |10    19| D7
D0  |11    18| D6
D1  |12    17| D5
D2  |13    16| D4
GND |14    15| D3
+-----+

```

One gate of the 74LS32 is used to enable the ROM only during memory reads when Z80 A15 is low, giving the following memory map:

```

$0000-$7FFF : ROM
$8000-$BFFF : DRAM (first 16K)
$C000-$FFFF : Work RAM (mirrored repeatedly every 2K)
- or -
$C000-$FFFF : DRAM (latter 16K if work RAM disabled and missing DRAM chips added)

```

DRAM interface

The MITEC-2 chip provides several output signals for managing DRAM:

MITEC pin	Cartridge pin	Signal name	Description
19	B14	CA7	DRAM column, bit 7
20	B17	/RCSEL	Row/column select (0= row, 1= column)
21	B12	/RAS0	Row address strobe for DRAM
22	B15	/RAS1	Row address strobe for DRAM
23	B13	/CAS0	Column address strobe for DRAM
24	B16	/CAS1	Column address strobe for DRAM

The Basic Level III A cartridge uses the above pins like so:

The two TMS4416 DRAM chips mapped to \$8000-\$BFFF use /RAS0 and /CAS0. Their /CAS signal is /RCSEL and /CAS0 logically-OR'd together.

The two unused chips mapped to \$C000-\$FFFF use /RAS1 and /CAS1. Their /CAS signal is /RCSEL and /CAS1 logically-OR'd together.

CA7 is common to all four chips.

Two multiplexers are used to break up the Z80 address bus into the row

and column words, formatted as follows:

TMS4416	Row (/RCSEL=0)	Column (/RCSEL=1)
DRAM A0	GND	Z80 A0
DRAM A1	Z80 A8	Z80 A1
DRAM A2	Z80 A9	Z80 A2
DRAM A3	Z80 A10	Z80 A3
DRAM A4	Z80 A11	Z80 A4
DRAM A5	Z80 A12	Z80 A5
DRAM A6	Z80 A13	Z80 A6
DRAM A7	GND	CA7

Bits 0 and 7 of the row for 16Kx4 DRAMs are unused to allow expansion. 64Kx4 DRAMs use these bits for address bits 14 and 15, respectively. Perhaps the SF-7000 uses this to implement it's 64K RAM.

Z80 A7 is not used for bit 7 of the column. Instead CA7 is used instead. Z80 A7 is used by A7 of the 32K ROM only.

Cartridge information from other sources:

Terebi Oekaki

\$0000-\$1FFF : ROM
\$6000 : Graphics tablet axis to read
\$8000 : Pen pressure sense / axis data conversion busy flag
\$A000 : Axis data

This cartridge connects to a pressure sensitive graphics tablet. A plastic stylus is used, which can have it's position tracked when pressed against the tablet.

SF-7000

\$0000-\$3FFF : ROM (8K, presumably mirrored twice)
\$0000-\$FFFF : RAM (as 48K from \$4000 or 64K from \$0000)

On power-up, the first 16K is for ROM and the remaining 48K is for RAM. The ROM can be switched out through the SF-7000's PPI, allowing all of the RAM to be used.

4.) Z80 port map

The VDP, PSG, and PPI are enabled according to the following port addresses:

Range	PPI	VDP	PSG	Data returned
-----	---	---	---	-----
\$00-1F	Y	Y	Y	PPI+VDP
\$20-3F	N	Y	Y	VDP
\$40-5F	Y	N	Y	PPI
\$60-7F	N	N	Y	Instruction referenced by R
\$80-9F	Y	Y	N	PPI+VDP
\$A0-BF	N	Y	N	VDP
\$C0-DF	Y	N	N	PPI
\$E0-FF	N	N	N	Instruction referenced by R

For each location, data written goes to all devices that are enabled.

For addresses where both the PPI and VDP are enabled (regardless of the PSG) the data returned is from the PPI but has a few bits corrupted, most likely due to the VDP trying to place information on the data bus at the same time.

The officially documented ports for each device are the ones where only one of them is enabled. The PSG has no readable registers, so even when it's enabled, there is no change to the data returned.

The SF-7000 maps its own hardware at ports \$E0-\$FF. I haven't looked into SF-7000 emulation much, so I can't comment on how these ports are used.

Memory refresh behavior

The Z80 has a function for providing DRAM refresh. During any opcode fetch cycle (either a prefix byte or the opcode itself) the following events occur during states T3 and T4:

- /RFSH goes low
- /M1 goes high
- A15-A8 output I register contents
- A7-A0 output R register contents
- /MREQ goes low for the latter half of the T3 cycle and the first half of the T4 cycle

So it looks like a regular memory access from the address indicated by the IR register pair, except /RFSH is used instead of /RD.

Typically, most SC-3000 cartridges are designed where ROM /OE = /MREQ and ROM /CS = A15. In this case they will respond to a refresh cycle within addresses \$0000-\$7FFF and output data to the bus.

The MITEC-2 chip does not enable work RAM during a refresh cycle, so the data returned during a refresh cycle to \$C000-\$FFFF is the last value left on the data bus from a previous operation, unless there is some external on-cart hardware using this memory range for its own purpose. (e.g. SF-7000) This also implies work RAM would be disabled by having cartridge connector pin B03 (work RAM /CS) tied to +5V.

This behavior manifests itself when reading I/O ports \$60-\$7F and \$E0-\$FF. No device will drive the data bus when these ports are read, so the value returned is the byte read during the refresh cycle. Here is a step-by-step description of what occurs:

Assume ROM data at offset \$2A00 is \$AB, \$CD, \$EF
Sample program is:

```
xor a
ld b, a
ld c, $E0
ld a, $2A
ld i, a
ld a, $01
ld r, a ; Set refresh address in IR pair. ($2A01)
in a, (c) ; Read from port in BC pair. ($00E0)
```

We are interested in the timing for the 'in a, (c)' instruction. In this case, it has three machine cycles of four T states each. The opcode for the encoding of this particular instruction is \$ED, \$78.

[Opcode fetch cycle]

T1 : Address bus outputs PC
T2 : Read opcode \$ED
T3 : Refresh cycle start (IR = \$2A01, read \$CD)
T4 : Refresh cycle end

[Opcode fetch cycle]

T1 : Address bus outputs PC
T2 : Read opcode \$78
T3 : Refresh cycle start (IR = \$2A02, read \$EF)
T4 : Refresh cycle end

[I/O cycle]

T1 : Address bus outputs BC (\$00E0)
T2 : /IORQ and /RD go low
TW : Forced wait cycle (lengthened if /WAIT low)
T3 : Data bus sampled, /IORQ and /WR go high

In this example, we assume there is ROM that will respond to the memory access for the reasons listed earlier. (ROM /OE = /MREQ, ROM /CS = A15)
The value read back from the IN instruction is \$EF, that was left over from the previous refresh cycle.

If there was no device to drive the data bus, the last value on the data bus is \$78, and that is the value that is returned during the IN instruction read.

An exception is the Basic Level III A cartridge; it's ROM is disabled during a refresh cycle, but the lower 3 bits of the data bus seem to fluctuate, so you get values like \$7B, \$7F, instead of \$78. Maybe all of the additional hardware in the cartridge that is connected to the data bus affects the bus capacitance somewhat, so it doesn't 'hold' the previous value as well from the T2 state of the 2nd machine cycle all the way to T3 of the 3rd machine cycle.

5.) Keyboard and I/O

The keyboard, gamepads, and cassette/prINTER interface are handled through an 8255 PPI. This chip has three 24 I/O pins which are arranged into 3 8-bit I/O ports called A, B, and C. It also has a control register which defines if the ports are outputs or inputs, amongst other things.

Port A (input)

D7 : Keyboard/gamepad input data
D6 : Keyboard/gamepad input data
D5 : Keyboard/gamepad input data
D4 : Keyboard/gamepad input data
D3 : Keyboard/gamepad input data
D2 : Keyboard/gamepad input data
D1 : Keyboard/gamepad input data
D0 : Keyboard/gamepad input data

Pins PA7-PA0 have pull-up resistors and are active low inputs.

Port B (input)

D7 : From CASSETTE-IN miniplug jack
D6 : BUSY input from printer port
D5 : FAULT input from printer port
D4 : /CONT input from B11 on cartridge connector
D3 : Keyboard/gamepad input data
D2 : Keyboard/gamepad input data
D1 : Keyboard/gamepad input data
D0 : Keyboard/gamepad input data

Pins PB6-PB0 have pull-up resistors and are active-low inputs, PB7 is an active-high input.

With no external devices attached, bits 7,6,5 return 0,1,1 respectively.

Port C (output)

D7 : To printer port pin 3 (/FEED output)
D6 : To printer port pin 5 (/RESET output)
D5 : To printer port pin 2 (DATA output)
D4 : Unused (not connected to anything)
D3 : To CASSETTE-OUT miniplug jack
D2 : To 74LS145 to select 1 of 8 keyboard rows (bit 2)
D1 : To 74LS145 to select 1 of 8 keyboard rows (bit 1)
D0 : To 74LS145 to select 1 of 8 keyboard rows (bit 0)

The descriptions of the PPI ports were taken from Marc Le Dourain's SF-7000 page, which I believe were in turn taken from a SF-7000 manual. So I can't really elaborate much more on what each bit does, beyond what I've checked myself.

The SC-3000H has a 64-key keyboard, and two gamepad ports. One of the keys is called "RESET" and generates a NMI on the Z80 when pressed.

Bits 7-0 of port A and 3-0 of port B are used to return data from the keyboard and gamepads. This data is broken down into 8 groups, with seven for the keyboard rows and one for the two gamepads.

The keyboard rows are assigned to the following keys, where each bit returns one for a pressed key and zero if the key has not been pressed.

Columns												
PPI Port A								PPI Port B				
Rows	D0	D1	D2	D3	D4	D5	D6	D7	D0	D1	D2	D3
-	-----											
0	'1'	'Q'	'A'	'Z'	ED	','	'K'	'I'	'8'	---	---	---
1	'2'	'W'	'S'	'X'	SPC	'.'	'L'	'O'	'9'	---	---	---
2	'3'	'E'	'D'	'C'	HC	'/'	','	'P'	'0'	---	---	---
3	'4'	'R'	'F'	'V'	ID	PI	':'	'@'	'-'	---	---	---
4	'5'	'T'	'G'	'B'	---	DA	']'	'['	'^'	---	---	---
5	'6'	'Y'	'H'	'N'	---	LA	CR	---	YEN	---	---	FNC
6	'7'	'U'	'J'	'M'	---	RA	UA	---	BRK	GRP	CTL	SHF
7	1U	1D	1L	1R	1TL	1TR	2U	2D	2L	2R	2TL	2TR

ED = "ENG DIER'S"
 SPC = (Spacebar)
 HC = "HOME CLR"
 ID = "INS DEL"
 PI = (PI symbol)
 DA = (Down arrow on keypad)
 LA = (Left arrow on keypad)
 RA = (Right arrow on keypad)
 CR = "CR" (Enter)
 UA = (Up arrow on keypad)
 YEN = (Yen symbol)
 BRK = "BREAK"
 GRP = "GRAPH"
 CTL = "CTRL"
 FNC = "FUNC"
 SHF = "SHIFT"
 1U = Joystick #1 up
 1D = Joystick #1 down
 1L = Joystick #1 left
 1R = Joystick #1 right
 1TL = Joystick #1 left trigger
 1TR = Joystick #1 right trigger
 2U = Joystick #2 up
 2D = Joystick #2 down
 2L = Joystick #2 left
 2R = Joystick #2 right
 2TL = Joystick #2 left trigger
 2TR = Joystick #2 right trigger
 --- = Key is unused, always returns 1

The keys are arranged in an 8x12 matrix as shown above. Some combinations of key presses will cause other keys to appear to be pressed. There are two rules which define this behavior:

If two or more keys on any keyboard row are pressed, pressing any key on another row that occupies the same columns will cause all other keys in the same columns to appear to be pressed.

For example, if keys 1, Q, A, Z are pressed, pressing 2 will also make keys W, S, and X appear to be pressed.

If two or more keys on any keyboard column are pressed, pressing any key on another column that occupies the same rows will cause all other keys in the same rows to appear to be pressed.

For example, if keys 1, 2, 3, 4 are pressed, pressing Q will also make keys W, E, and R appear to be pressed.

The two joystick ports are affected by this behavior too.

Most software writes \$92 to the PPI control register and \$07 to PPI port C, which configures ports A and B as inputs and all bits of port C as outputs, as well as selecting row 7 of the keyboard matrix to access the gamepads.

When port C is configured as an output, reading it returns the last value written to port C. The PPI control register cannot be read, and always returns \$FF.

6.) TMS9929A information

I'll include more details later on. For now, just a few notes:

- When writing to a VDP register, bits 6-3 of the byte written are ignored by the VDP.
- Data written to the data port is also copied to the VRAM read buffer.
- Bits 7-3 of register #0 and bit 2 of register #1 have no use.
- Bit 0 of register #0 does not turn off the screen when set, but rather distorts the synchronization of the display. This bit is intended to enable external video input which is unused in the SC-3000H.
- Bit 7 of register #1 affects how the VDP generates addresses when accessing VRAM. Here's a table illustrating the differences:

VDP address (Column)	VRAM address	
	4K mode	8/16K mode
AD0	VA0	VA0
AD1	VA1	VA1
AD2	VA2	VA2
AD3	VA3	VA3
AD4	VA4	VA4
AD5	VA5	VA5
AD6	VA12	VA6
AD7	Not used	Not used
(Row)		
AD0	VA6	VA7
AD1	VA7	VA8
AD2	VA8	VA9
AD3	VA9	VA10
AD4	VA10	VA11
AD5	VA11	VA12
AD6	VA13	VA13
AD7	Not used	Not used

ADx - TMS9928 8-bit VRAM address/data bus

VAx - 14-bit VRAM address that the VDP wants to access

How the address is formed has to do with the physical layout of memory cells in a DRAM chip. A 4Kx1 chip has 64x64 cells, a 8Kx1 or 16Kx1 chip has 128x64 or 128x128 cells. Because the DRAM address bus is multiplexed, this means 6 bits are used for 4K DRAMs and 7 bits are used for 8K or 16K DRAMs.

In 4K mode the 6 bits of the row and column are output first, with the remaining high-order bits mapped to AD6. In 8/16K mode the 7 bits of the row and column are output normally. This also means that even in 4K mode, all 16K of VRAM can be accessed. The only difference is in what addresses are used to store data.

7.) Miscellaneous

In my experience a US SMS 2 power supply and standard Genesis / MegaDrive

A/V cable (composite video + mono audio) will work with a SC-3000H.

It seems that when an interrupt occurs, the value on the Z80's data bus is random data. It could be the same as reading unused ports, where the Z80's R register indexes ROM to select the value read.

This means that interrupt mode 0 and 2 cannot be used reliably. In the latter case it may be possible to set the I register to a location in RAM and fill up all 257 bytes from that point with the same byte, say \$08, so the Z80 will jump to \$0808 regardless of what data it read as the low byte for the vector table. I think a similar technique is used in some Spectrum computer software, though I have not tested this myself.

The SN76489A is not reset when the SC-3000H is powered on. If you cycle the power to reset the machine, then the PSG plays the same sounds that were last written to it, and from a cold boot the PSG emits a medium volume tone.

Custom chip pin assignments

Cassette interface control (11-pin SIP)

1 : From CASSETTE-IN miniplug jack
2 : Input from PPI PC3 (Tape data out)
3 : To CASSETTE-OUT miniplug jack
4 : Ground
5 : Output to PPI PB7 (Tape data in)
6 : Unused (not connected to anything)
7 : Unused (not connected to anything)
8 : Unused (not connected to anything)
9 : Unknown (connected to a transistor and some other analog components)
10 : Unused (not connected to anything)
11 : +5V

This part is a small epoxy-covered PCB with a row of 11 pins on the right side. It digitizes the incoming tape data to be read through PPI PB7 and converts the output of PPI PC3 to analog data to be recorded onto the tape.

SEGA MITEC 2 (28-pin DIP)

```

      +----V----+
Z80 /NMI |01 o x 28| VCC
Z80 /MREQ |02 i o 27| TMS9929A /CSW
NMI-IN  |03 i o 26| TMS9929A /CSR
Z80 /RD  |04 i o 25| SN76489A /CE
Z80 /WR  |05 i o 24| /CAS1
Z80 /IORQ |06 i o 23| /CAS0
Z80 /RFSH |07 i o 22| /RAS1
MEMORY /RD |08 o o 21| /RAS0
MEMORY /WR |09 o o 20| /RCSEL
I/O PORT /RD |10 o o 19| CA7
I/O PORT /WR |11 o i 18| Z80 A7
Z80 A14 |12 i i 17| Z80 A6
Z80 A15 |13 i o 16| /EXM1
GND |14 x o 15| Work RAM /CS
      +-----+
```

NMI-IN is from the keyboard RESET key. I don't know if it's active high or low.

The memory /RD, /WR strobes are used by the 2Kx8 work RAM and cartridge connector.

The I/O port /RD, /WR strobes are used by the 8255 PPI and cartridge connector.

The TMS9929A has no chip select input, so it has it's own read/write strobes.

The SN76489A /WE and /READY pins are tied together to Z80 /WAIT. I think

the MITEC-2 enables the PSG (via /CE) when the Z80 has written to the PSG port and is driving the data bus; then PSG /READY signal will delay the Z80 while the PSG is processing the data being input from the bus.

Things to check:

- Find source of data read during memory read to unused address spaces.
- Find source of data read during interrupt cycle in IM 0 and IM 2.
- Determine why TMS9929 VRAM content resets to \$FF over a period of time, only for tiles in the pattern generator that are not accessed. (currently being displayed)

8.) Assistance Needed

- Is there any information about how the cassette and printer interfaces work? Does any software support either one (Basic maybe), or are there any cassette images available?
- I'm looking for datasheets for the MCM4517P15, TMS4027, TMS4108, TMS4116, TMS4416, MB81416 DRAMs, as well as schematics for any hardware that uses a TMS9918 or related video chip with only 4K of video RAM.
- I'd be interested in knowing what kind of hardware is inside a Terebi Oekeki or Rozetta no Shouzou cartridge (chip part numbers, etc.)
- Could anyone tell me what parts are in a SF-7000? If it implements it's main memory with two 64Kx4 DRAMs, would somebody be willing to trace out the connections to the cartridge port, for comparison with the way DRAMs are managed in the Basic Level III A cartridge?

9.) Credits and acknowledgements

- Omar Cornut for MEKA, the cartridge and A/V pinouts, Terebi Oekaki information, and a lot of advice. :)
- Marc Le Dourain for his great SF-7000 webpage. This is where I got information about the 8255 PPI and keyboard.
- Chris MacDonald for support and program testing.
- Sean Young for the TMS9918 documentation.

10.) Disclaimer

If you use any information from this document, please credit me (Charles MacDonald) and optionally provide a link to my webpage (<http://cgfm2.emuviews.com/>) so interested parties can access it.

The credit text should be present in the accompanying documentation of whatever project which used the information, or even in the program itself (e.g. an about box)

Regarding distribution, you cannot put this document on another website, nor link directly to it.

Unpublished work Copyright 2002-2004 Charles MacDonald