Sega Master System Information

Cartridge port layout

Back				Front
+5V	1		2	Wr
Mreq	3		4	ROM/RAM select
M8-B	5		6	A14
A13	7		8	A8
A9	9		10	A11
M0-7	11		12	A10
CrtOe	13		14	D7
D6	15		16	D5
D4	17		18	D3
GND	19		20	GND
GND	21		22	D2
D1	23		24	D0
A0	25		26	A1
A2	27		28	A3
A4	29		30	A5
A6	31		32	A7
A12	33		34	??
+5V	35		36	A15
M1	37		38	IoReq
Refresh	39		40	Halt
Wait	41		42	Int
JyDs	43		44	BusReq
BusAck	45		46	Reset
Clk	47		48	JRead
MC-F	49		50	NMI

Shaded pins don’t appear to be used for ROM cartridges and are mainly signals connected to the processor. Pins 3, 4, 46 and 47 are used on some cartridges, normally ones with extra RAM or battery backed RAM. Pins 3 and 47 are used by Codemasters cartridges as they don’t appear to use the regular Sega roms. Instead they have a PAL16R4 to generate the addresses A14, A15, A16 and A17 for a regular 2Mbit ROM chip. Pin 4 appears to select either ROM (0) or RAM (1), probably controlled by register 0xFFFC bit 3

Pre-decoded memory areas.

Pin	Type	Address range
M0-7	ROM (32k page)	0000 - 7FFF
M8-B	ROM/RAM (16k page)	8000 - BFFF
MC-F	RAM (16k)	C000 - FFFF

These don’t normally get connected in cartridges, some games have them

Cartridge Sizes

Part Number	ROM Size (k bytes)	No. of Pages
4xxx	32	2
5xxx	128	8
6xxx	Combo (256?)	16
7xxx	256	16
8xxx	3D (256?)	16
9xxx	512	32
x5xx	Battery Backed Ram	

Some non Sega cartridges have the code bytes swapped (eg Terminator) others don't have one at all.

Cartridge internal info

The last 16 bytes of the 1st 32k block of code holds some cartridge information, addresses 0x7FF0 to 0x7FFF. The first 8 bytes contain the text "TMR SEGA". The next 8 bytes are coded as follows (as far as I can work out!)

Address	Data
7FF8	Unknown Possibly a date code??
7FF9	Unknown 2nd part of date code??
7FFA	Checksum Lo
7FFB	Checksum Hi
7FFC	Part No. Lo
7FFD	Part No. Hi
7FFE	Unknown (configuration options??)
7FFF	Unknown (configuration options??)

In some cartridges the some fields are used, others they are FF. The only fields for sure are the part numbers, they match the number on the box and are used to identify the size of the cartridge ROM. The first two bytes usually contain FF, FF, although sometimes they have what appears to be a date, usually the year, eg 1987, 1990 etc.

The checksum calculation is the sum of all bytes except the TMR SEGA and following 8 bytes. Only low 2 bytes are used. The 512k ROMS may use a different method as they always seem to fail the checksum calculations.

Examples of cartridge info

Address	Mickey Mouse Castle of Illusion	Gauntlet	Fantasy Zone	Double Dragon	Sonic The Hedgehog
7FF8	FF	19	54	FF	59
7FF9	FF	90	41	FF	59
7FFA	F3	24	EB	C6	1B
7FFB	2A	72	82	5C	A5
7FFC	53	06	52	12	76
7FFD	70	50	50	70	70
7FFE	0E	20	02	00	03
7FFF	04	4F	4F	4F	03

Address	Zool	Xenon 2	World Grand Prix	Teddy Boy	Pacmania
7FF8	20	FF	FF	20	FF
7FF9	20	FF	FF	20	FF
7FFA	59	A3	39	75	6D
7FFB	0E	8B	09	89	B0
7FFC	75	38	80	03	10
7FFD	70	70	50	40	50
7FFE	3D	21	00	00	20
7FFF	32	40	4F	4C	4F

How to read cartridges.

The first 32k (2 pages) of a cartridge is easy, set address, set CrtOE low, read data bus, set CrtOE high. If reading from a particular area, it is also a good idea to set the appropriate area select bit (M0-7 for 1st 32k, M8-B for next 16k etc).

From the cartridge info fields the number of pages can be determined, subtract 2 and this is how many further pages must be read (1st 2 should have already been read).

There are a number of paging registers but the one to use is at address 0xffff for 128k cartridges, reading a 16k block starting from address 0x8000. All other cartridge types write page number to 0xfffe and read 16k starting from 0x4000. I don’t know why the 128k cartridges are different. Paging registeres are accessed by setting address, setting data bus, clear CrtOE and WR, set CrtOE and WR. page register should then have been written to, read from the appropriate memory area to acces the page. I must determine the exact sequencing (check data books etc).

Register	**Page Location**	**Page Size**
0xFFFD	0x0400	15k
0xFFFE	0x4000	16k
0xFFFF	0x8000	16k

Sega Master System Cartridge Reader

This information refers to the MKII reader that connects to the Centronics port, although the basic operation applies to the MK I 8255 I/O card version.

The interface consists of a number of sections, namely, address generation, control signal latching and interface control. The sections are described below. The centronics port being used must be capable of bi-directional data transfers, this is achieved by setting bit 5 of the control register high, this sets the output buffer to a high impedance state allowing the data pins to be driven externally

Address Generation.

Since the cartridge reader basically reads sequential addresses, the address generation is achieved by means of a pair of 12 bit counters, IC1 and IC2 on the schematic. IC1 is connected to the address pins A0 to A11, IC2 is connected to address pins A12 to A15 to generate the full 64k address range. The address is set by means of two inputs, a clear input that resets the address to zero and an increment input that sets the next address on the address pins.

Reading Data.

Once the address has been set the data direction must be set to read, the OE pin is set low and the data is read on data register.

Interface Control.

Control signals and extra cartridge signals are done through a latch (IC3) that is set with the appropriate bit map. Only the Reset address and pre-decoded address area signals are currently used.

Andrew Lindsay
3/03/1997

PC Parallel Port Mini-FAQ

By Kris Heidenstrom (kheidenstrom@actrix.gen.nz), revision 2, 941019

1. INTRO

This is a four page mini-FAQ with the information essential for programming the PC parallel port. Many subjects are not covered in detail. Comments and suggestions to kheidenstrom@actrix.gen.nz.

A parallel port links software to the real world. To software, the parallel port is three 8-bit registers occupying three consecutive addresses in the I/O space. To hardware, the port is a female 25-pin D-sub connector, carrying twelve latched outputs from the computer, accepting five inputs into the computer, with eight ground lines.

The normal function of the port is to transfer data to a printer through eight data pins, using the remaining signals as flow control and miscellaneous controls and indications.

The original port was implemented with TTL/LS logic. Modern ports are implemented in an ASIC or a combined serial/parallel port chip, but are backward compatible. Some modern ports are bidirectional.

2. BIOS LPT PORT TABLE

A parallel port is identifed by its I/O base address, and also by its LPT port number. The BIOS power-on self-test checks specific I/O addresses for the presence of a parallel port, and builds a table of I/O addresses in the low memory BIOS data area, starting at address 0040:0008 (or 0000:0408).

This table contains up to four 16-bit words. Each entry is the I/O base address of a parallel port. The first word is the I/O base address of LPT1, the second is LPT2, etc. If less than 4 ports were found, the remaining entries in the table are zero. DOS, and the BIOS printer functions (accessed via int 17h), use this table to translate an LPT port number to a physical port at a certain address.

The addresses are checked in a specific order, and entries are put into the table as they are found, so there will never be gaps in the table, and a particular I/O address does not necessarily always equate to the same specific LPT port number, although there are conventions.

2.1 ADDRESSING CONVENTIONS

The video card’s parallel port is normally at 3BCh. This address is checked first by the BIOS, so if a port exists there, it will be LPT1. The BIOS then checks at 378h, then at 278h. AFAIK there is no standard address for a fourth port, and BIOSes only look for three.

3. DIRECT HARDWARE ACCESS

The port consists of three 8-bit registers at adjacent addresses in the I/O space. The registers are defined relative to the I/O base address, and are at IOBase+0, IOBase+1, and IOBase+2. Always use 8-bit accesses on these registers.

3.1 DATA REGISTER

The data register is at IOBase+0. It may be read and written (using the IN and OUT instructions, or inportb() and outportb() or inp() and outp()). Writing a byte to this register causes the byte value to appear on pins 2 through 9 of the D-sub connector (unless the port is bidirectional and is set to input mode). The value will remain latched until you write another value to the data register. Reading this register yields the state of those pins.

```
        7 6 5 4 3 2 1 0          * . . . . . . .  D7 (pin 9), 1=High, 0=Low      . * . . . . . .  D6
(pin 8), 1=High, 0=Low       . . * . . . . .  D5 (pin 7), 1=High, 0=Low      . . . * . . . .  D4
(pin 6), 1=High, 0=Low       . . . . * . . .  D3 (pin 5), 1=High, 0=Low      . . . . . * . .  D2
(pin 4), 1=High, 0=Low       . . . . . . * .  D1 (pin 3), 1=High, 0=Low      . . . . . . . *  D0
(pin 2), 1=High, 0=Low
```

3.2 STATUS REGISTER

The status register is at IOBase+1. It is read-only (writes will be ignored). Reading the port yields the state of the five status input pins on the parallel port connector at the time of the read access:

7 6 5 4 3 2 1 0 * . . . . Busy . . (pin 11), high=0, low=1 (inverted) . * . . . . . . Ack . . (pin 10), high=1, low=0 (true) . . * . . . . . No paper (pin 12), high=1, low=0 (true) . . . * . . . . Selected (pin 13), high=1, low=0 (true) . . . . * . . . Error. . (pin 15), high=1, low=0 (true). . . . . * * * Undefined

3.3 CONTROL REGISTER

The control register is at IOBase+2. It is read/write:

7 6 5 4 3 2 1 0 * * . . . . . . Unused (undefined on read, ignored on write) . . * . . . . . Bidirectional control, see below . . . * . . . . Interrupt control, 1=enable, 0=disable . . . . * . . . Select . . (pin 17), 1=low, 0=high (inverted) . . . . . * . . Initialize (pin 16), 1=high, 0=low (true) . . . . . . * . Auto Feed (pin 14), 1=low, 0=high (inverted) . . . . * Strobe . . (pin 1), 1=low, 0=high (inverted)

3.3.1 BIDIRECTIONAL CONTROL BIT

The bidirectional control bit is only supported on true bidirectional ports - on other ports, it behaves like bits 7 and 6. On a proper bidirectional port, setting this bit to '1' causes the outputs of the buffer that drives pins 2 through 9 of the 25-pin connector to go into a high-impedance state, so that data can be _input_ on those pins.
In this state, values written to the data register will be stored in the latch chip, but not asserted on the connector, and reading the data register will yield the states of the pins, which may be driven by an external device without stressing or damaging the port driver.
Also note that on some machines, another port must be set correctly to enable the bidirectional features, in addition to this bit.

3.3.2 INTERRUPT ENABLE BIT

DOS and BIOS do not use the interrupt facility of the parallel port during printing. OS/2 does, or did, use and rely on the parallel port interrupt. For experimenters, it is useful as a general purpose externally triggerable interrupt input.
The interrupt control bit controls a tristate buffer that drives the IRQ line. Setting the bit to '1' enables the buffer, and an IRQ will be triggered on each falling edge (high to low transition) of the Ack signal on pin 10 of the 25-pin connector. Disabling the interrupt allows other devices to use the IRQ line.
The actual IRQ number is either hardwired (by convention, the port at 3BCh uses IRQ7), or is jumper-selectable (IRQ5 is a common choice). Sound cards, in particular, tend to use IRQ7 for their own purposes.
To use the IRQ you must also enable the interrupt via the interrupt mask register in the interrupt controller, at I/O address 21h.

3.3.3 PRINTER CONTROL BITS

The bottom four bits are latched and presented on the parallel port connector, much like the data register. Three of them are inverted, so writing a '1' will output a low voltage on the port pin for them.
These four outputs are open collector outputs with pullup resistors, so an external device can force them low without stressing the driver in the PC, and they can even be used as inputs.
To use them as inputs, write 0100 binary to the bottom four bits of the control register. This sets the outputs all high, so they are pulled high by the pullup resistors (typically 4700 ohms). An external device can then pull them low, and you can read the pin states by reading the control register. Remember to allow for the inversion on three of the pins.
If you are using this technique, the control register is not strictly 'read/write', because you may not read what you write (or wrote).

4 SAMPLE CODE

These sample code fragments are totally untested.

4.1 RETURN I/O BASE ADDRESS OF NOMINATED LPT PORT

This function returns the I/O base address of the nominated LPT port.The input value must be 1 to 4. If the return value is zero, the pecified port does not exist.

```
#include <dos.h>unsigned int get_lptport_iobase(unsigned int
lptport_num) {   return *((unsigned int far *)MK_FP(0x40, 6) + 2 *
lptport_num);   }
```

4.2 TOGGLE DATA BITS IN A LOOP

This function just toggles the eight data bits on LPT1 as fast as it can, forever. Not very useful. Change outportb() and inportb() to outp() and inp() for Microsoft C, I think.

```
#include <dos.h>void toggle_parallel_data_bits(void) {   unsigned int
lpt_ioaddr;   if ((lpt_ioaddr = get_lptport_iobase(1)) != 0) {
outportb(ioaddr, 0x55);      for (;;) outportb(ioaddr, inportb(ioaddr) ^
0xFF);      }   return;   }
```

5. FILE TRANSFER PROGRAM CABLES

The parallel-to-parallel cable is used by DRDOS's INTERLNK program.a pparently Laplink and FastLynx cables are the same. The pin-to-pin connection between two male 25-pin D-sub connectors is: 2-15, 3-13, 4-12, 5-10, 6-11, and the reverse: 15-2, 13-3, 12-4, 10-5, and 11-6, and ground: 25-25. This requires eleven wires. If you have spare wires, link some extra grounds together. Pins 18 to 25 inclusive are grounds. A very long cable may be unreliable, limit it to 10 metres.

6. DISCLAIMER

In no event shall the author be liable for any damages whatsoever for any loss relating to this document. Use it at your own risk!

End of the PC Parallel Port Mini-FAQ.-- Kris Heidenstrom
kheidenstrom@actrix.gen.nz Wellington, New Zealand