

**TERRA
ONION**



MEGA SD DEV MANUAL REV.2

**USING CD ENHANCED
MEGADRIVE GAMES**

You can easily add CD audio to an existing megadrive game with the MegaSD command interface. This mode doesn't have access to the extra CD hardware.

This interface is accessed as an overlay at the 3F7F6h - 3FFFFh area. By default the overlay is disabled, and so, any access to that area goes to the game rom at these addresses and behave as normal.

In order to access the command interface, **you must enable the overlay by writing CD54h to address 3F7FAh**. It must be written as a single word write, writing the bytes separately won't work.

Once it's enabled, the overlay **shows different command and result ports**:

- **3F7FEh**: Command port, as with the overlay port, you must always use word writes. The high 8 bits are the command, the low 8 bits are a parameter
- **3F7FCh**: Result port. This port contains the response to the previous command (if any)
- **3F7FAh**: overlay port. Writing a value different than CD54h here closes the overlay, and rom appears there as usual
- **3F7F6h**: ID port, when the overlay is open, the value 4241h appears here
- **3F7F8h**: ID port, when the overlay is open, the value 5445h appears here. With the previous port, they form the word "RATE"
- **3F800h - 3FFFFh**: 2048 byte data area. This data is used to send and receive data and parameters to/from the ARM.

Commands:

- **10h**: Retrieve the MegaSD version and serial number.

Parameter: Not used

Response: Not used

Data:

3F800h : 'M'

3F801h : 'E';

3F802h : 'G';

3F803h : 'A';

3F804h : 'S';

3F805h : 'D';

3F806h : Version high byte (hh)
3F807h : Version low byte (ll)
3F808h : Build byte (rr) (Version is hh.ll Rrr)
3F809h : 00h;
3F80Ah : FFh;
3F80Bh : FFh;
3F80Ch - 3F800Fh : Serial number (32 bits)

- **11h:** Play specified CDDA track and stop when finished
Parameter: Track number 1 - 99 (01h - 63h)
Response: Not used
- **12h:** Play specified CDDA track and loop
Parameter: Track number 1 - 99 (01h - 63h)
Response: Not used
- **13h:** Pause CDDA/WAV play
Parameter: (Firmware up to 1.03): Not used
(Since firmware 1.04): Fade out for the specified number of audio sectors
(there are 75 sectors per second, so specify 75 for 1 second fade out)
Response: Not used
- **14h:** Resume CDDA/WAV play
Parameter: Not used
Response: Not used
- **15h:** Set CDDA/WAV Volume
Parameter: Volume 00h-FFh
Response: Not used
- **16h:** Get CDDA/WAV Status (Since firmware 1.04)
Parameter: Not Used
Response: 0: CDDA/WAV is not playing. 1: CDDA/WAV playing
- **17h:** Request CD sector read
Parameter: Not used
Response: Not used
Data: 3F800h-3F803h: LBA of data sector to read (including pregap,
so initial sector is usually 150 (96h))

- **18h:** Transfer last read sector
Parameter: Not used
Response: Not used
Data: 3F800h-3FFFFh: sector data
- **19h:** Request read of next cd sector
Parameter: Not used
Response: Not used
- **1Ah:** Play specified CDDA track, use loop offset set in Data (Since firmware 1.04)
Parameter: Track number 1 - 99 (01h - 63h)
Response: Not used
Data: 3F800h-3F803h : offset in sectors from the start of the track to apply when looping.
- **1Bh:** Play CDDA from specific sectors (Since firmware 1.04)
Parameter: 1: loop play, 0: no loop
Response: Not Used
Data:
 - 3F800h-3F803h:** Starting disc sector to begin playing from
 - 3F804h-3F807h:** End disc sector. When this sector number is arrived, the sound will either stop or restart at the Loop sector, based on the parameter value
 - 3F808h-3F80Bh:** Loop sector. To loop the whole data, set this value to the same than the starting sector.
- **1Ch:** Begin reading from a file (Since firmware 1.04)
Parameter: Not used
Response: 0: error opening file. 1: Open ok
Data: 3F800h-3F8xxh: full path file name. Zero terminated, Max 1023chars + terminator.

Notes:

Path separator is /.

Specify the file name starting with ./ to read it in the current directory, otherwise path is absolute from the root of the sdcard: ("data.bin" mean read data-bin from the root of the sd card, "./data.bin" means read data.bin from the directory where the rom file is located.

- **1Dh:** Read data from the current selected file (Since firmware 1.04)
Parameter: Not used
Response: 0: error reading data, 1: Read ok
Data:
 - 3F800h-3F803h:** Offset to start reading from. Set FFFFFFFF to continue reading from last read.

3F804h-3F807h: Number of bytes to read

3F808h-3F80Bh: ROM destination address.

3F808h = 00h: Transfer to ROM. Bytes 3F807h-3F80Bh specify a rom physical offset, so allowed values are 000000h to 7FFFFFFh. Use SSF2 extended mapper to access data stored beyond 400000h

Notes: When destination is ROM, this command does a DMA to the game ROM, taking ownership of the cartridge bus, and must be run and waited from code running in RAM with interrupts off. Don't run any ROM code (or interrupts or VDP DMA from ROM) while the data is transferred.

• **1Eh:** Read directory files

Parameter: Not used

Response: 0: error retrieving directories. 1: Directory read ok

Data:

3F800h-3F803h: ROM destination address.

3F804h-3F8xxh: full path of the directory to scan

Destination Data: The game list will be written at the specified physical rom offset.

Ensure there is enough capacity to hold up to 1000 64-byte file names

Offset 0-1: number of read files

Offset 2 + 64*x: File name and flags for file x

Sub offset 0-62: File Name + terminator

Sub offset 63: Flags: 00h: file: 01h: directory, FFh: go to previous directory.

As with the file open command, use ./ to specify paths relative to the directory the rom is located.

• **1Fh:** Play WAV file:

Parameter: 1: loop play, 0: no loop

Response: 0: error opening file or incorrect format. 1: Open ok and play started

Data:

3F800h-3F803h: offset in samples for the loop address. Specify 0 to loop the entire file

3F804h-3F8xxh: full path file name. Zero terminated, Max 1023 chars + terminator.

Notes: To stop wave play, just issue normal CDDA Pause/Resume commands. Only 44100Hz, 16-bit, stereo .wav files are supported.

As with the file open command, set ./ before the wav file to read it from the directory the rom is located.

- **20h:** Read a 2048 byte block from file to sector buffer

Parameter: Not Used

Response: Not Used

Data:

3F800h-3F803h: Offset to start reading from.

Note: Use the Transfer last read sector (command 18h) to copy data from the sector buffer to the 3F800h block.

- **21h:** Read next 2048 byte block from file to sector buffer

Parameter: Not Used

Response: Not Used

Note: Use the Transfer last read sector (command 18h) to copy data from the sector buffer to the 3F800h block.

Once a command is sent, **the ARM will be busy till finishing, no more commands can be sent.** To know when it has finished processing a command, just read the command port (3F7FEh). If high byte (command byte) is 00h, it has finished processing command, otherwise, it's still processing the command.

The overlay doesn't need to be open for the commands to continue running, so, you can send a command that takes a long time (like reading a sector to the sector buffer), close the overlay and continue the game as is, then open the overlay, and check if it has finished...

You can double-buffer the sector reading for faster speed and avoiding waiting times. To do so:

- **1** - First request to read the first sector (cmd 17h), by specifying its LBA (for example sector 150).
- **2** - Wait till the command has completed by reading the command port.
- **3** - Transfer the sector data to the data buffer (cmd 18h)
- **4** - Wait till the command has completed by reading the command port.
- **5** - Request reading of next cd sector (cmd 19h)
- **6** - Using the 68k, transfer the data from the first sector from address 3F800h to the desired RAM or VDP address (you can use VDP dma)
- **7** - Wait till the read next sector command has completed.
- **8** - Repeat from step 3 till all sectors have been transferred

Specifying loop points in .cue files (new in Firmware 1.04)

Starting with firmware 1.04, MegaSD MD+ games can specify loop and loop points in the .cue files, so it doesn't need updating the patched code to support a different track arrangement. This is done by using the REM comment keyword in the .cue file, that is generally ignored.

Those commands override the loop mode and offset specified in the code, so you can for example always use the "Play with loop" command, and use the .cue commands to control whether it should actually loop. Available commands are:

REM LOOP

Forces the track to play looped.

REM LOOP xxxxx

Forces the track to play looped and when looping, start xxxxx playing sectors from the start of the track.

REM NOLOOP

Forces the track to play without loop

Command are specified AFTER the TRACK xx declaration, so they affect the last specified TRACK number. For example:

```
FILE "1-StPetersburgIntro.wav" WAVE
  TRACK 01 AUDIO
    INDEX 01 00:00:00
    REM NOLOOP
```

```
FILE "2-StPetersburg1.wav" WAVE
  TRACK 02 AUDIO
    INDEX 01 00:00:00
    REM LOOP 132
```

```
FILE "3-StPetersburg2.wav" WAVE
  TRACK 03 AUDIO
    INDEX 01 00:00:00
    REM LOOP 36
```

Using ROM + MegaCD hardware

You can improve the previous method by adding access to the MegaCD hardware as if the original hardware was plugged.

Start with the same method, **adding a megadrive ROM and a CD image to the same folder**, and make a small modification to the ROM header. At offset 100h where usually "SEGA MEGA DRIVE" or "SEGA GENESIS" appears, it must be changed to "SEGA MEGASD" and filling with spaces till 10Fh. This will make MegaSD to load the ROM, the BIOS according to the CD security sector (or EU if none). In this mode, the CD hardware appears at offset 400000h, that is, BIOS is at 400000h, PRG RAM window is at 402000h, WORD RAM is at 600000h. CD ASIC registers are at A12000h as usual.

You can directly write your own SUB CPU program to PRG RAM and execute it. That's ok for using the PCM or GPU chips, but interfacing the CD hardware is complicated, so it's better to load the bios and initialize it, using what the Mega CD manuals call mode 1 boot.

There are several samples over the internet on how to do that. You can start at:
<http://www.sega-16.com/forum/showthread.php?19657-SEGA-CD-Mode-1>.

The game ROM is not able to use any mappers, so it's limited to 4MB. By default it's mapped as ROM, but you can make it read/write by writing 57h ('W') to address A130FFh, so you can use any of the rom space as RAM. For backup data read/save, you must use the CD backup RAM (8KB), either through bios, or directly writing to it. Keep in mind that if a backup ram file doesn't exist, MegaSD will format it in the cd bios format.

The command overlay used in the previous method is also available, but it's not recommended as will cause issues with the normal CD processing mode.

Using the MegaSD enhanced SSF2 mapper

MegaSD contains an enhanced SSF2 mapper that allows to use the entire 8MB ROM available area with bankswitching, and also being able to remap the first block, that is fixed in the SSF2 mapper (however, it must be changed with extreme care, as interrupt vectors are there). Also the entire area can be set to Read/Write, so you can use the banks as RAM if needed. When using the Read/Write mode, savestates are unreliable, as they will not save the entire rom area.

To enable the mapper, set SEGA SSF2 at ROM address 100h (instead of the normal SEGA MEGA DRIVE or SEGA GENESIS).

Bankswitching Regs:

A130F0: P-W-----

P : Protect bit. It must be set to 1 in order to W or A130F1h bank to apply
W : Allow writting to the ROM area (1: enable, 0: disable (default))

A130F1: ----RRRR

RRRR : 512KB bank to map to 00000h-7FFFFh area (max 0Fh).
Ensure interrupts are disabled when changing this bank.
A130F0h and F1 can be written in a single 16 bit write to A130F0h

A130F3: ----RRRR

RRRR : 512KB bank to map to 80000h-FFFFFFh area (max 0Fh)

A130F5: ----RRRR

RRRR : 512KB bank to map to 100000h-17FFFFh area (max 0Fh)

A130F7: ----RRRR

RRRR : 512KB bank to map to 180000h-1FFFFFFh area (max 0Fh)

A130F9: ----RRRR

RRRR : 512KB bank to map to 200000h-27FFFFh area (max 0Fh)

A130FB: ----RRRR

RRRR : 512KB bank to map to 280000h-2FFFFFFh area (max 0Fh)

A130FD: ----RRRR

RRRR : 512KB bank to map to 300000h-37FFFFh area (max 0Fh)

A130FF: S---RRRR

S : Special bank when 1

RRRR : 512KB bank to map to 380000h-3FFFFFFh area (max 0Fh)

The last bank (380000h) allows some special bank values when high bit (S bit) is set:

80h: Map 16KB of non volatile SRAM, mapped on the odd bytes. Writing to this RAM causes its data to be stored in the SD card. When the game is loaded, it will be restored from a file. Don't use it as general purpose RAM, as writes here trigger saving to the SD card.

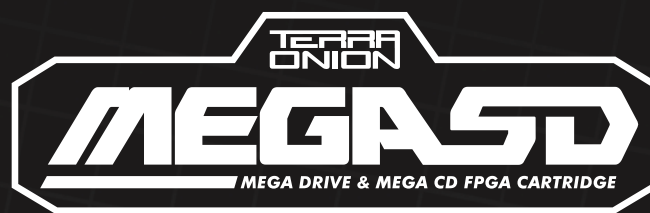
81h: Map the PCM chip into this bank space. It's the same Mega CD PCM chip. Refer to a MegaCD programming manual to know how to use this chip.

The PCM chip interface is 8 bit wide, and its memory map is as follows:

380001h - 381FFFh Registers, mapped only on the odd bytes

382001h - 383FFFh Memory window, mapped only on odd bytes

Note that in order to be able to write to this area, the W bit in the A130F0 port must be set first, otherwise writes are ignored by the fpga, and you won't be able to write to backup ram or to PCM chip.



Consumer info available at
<https://www.terraonion.com>

