ROM header reference

The ROM header comes after the 68000 vectors and spans the `$100`-`$1FF` range. The header includes some information such as the name of the game, or when it was released, or whether it can save, etc. Sometimes emulators may also read parts of the header looking for some information (e.g. what to put on the titlebar or what peripherals to emulate by default).

ROM header format

Address	Size	Description
`$100`	16 bytes	System type
`$110`	16 bytes	Copyright and release date
`$120`	48 bytes	Game title (domestic)
`$150`	48 bytes	Game title (overseas)
`$180`	14 bytes	Serial number
`$18E`	2 bytes	ROM checksum
`$190`	16 bytes	Device support
`$1A0`	8 bytes	ROM address range
`$1A8`	8 bytes	RAM address range
`$1B0`	12 bytes	Extra memory
`$1BC`	12 bytes	Modem support
`$1C8`	40 bytes	(reserved, fill with spaces)
`$1F0`	3 bytes	Region support
`$1F3`	13 bytes	(reserved, fill with spaces)

System type

What system the game is running on, padded with spaces. For the Mega Drive, it usually would be "`SEGA MEGA DRIVE`" or "`SEGA GENESIS`". Some flashcarts enable extra functionality if a specific text is found here.

⚠ **This is the only part of the header a console pays attention to. If it doesn't find the initial "`SEGA`" text in this field, systems with TMSS will refuse to boot the game.**

All known system types so far:

List of system types

Value	Description
`"SEGA MEGA DRIVE"`	Mega Drive
`"SEGA GENESIS"`	Mega Drive
`"SEGA 32X"`	Mega Drive + 32X
`"SEGA EVERDRIVE"`	Mega Drive (Everdrive extensions)
`"SEGA SSF"`	Mega Drive (Mega Everdrive extensions)
`"SEGA MEGAWIFI"`	Mega Drive (Mega Wifi extensions)
`"SEGA PICO"`	Pico
`"SEGA TERA68K"`	Tera Drive (boot from 68000 side)
`"SEGA TERA286"`	Tera Drive (boot from x86 side)

Copyright and release date

A field with the format "`(C)XXXX YYYY.ZZZ`", where:

Parts of the copyright field

Part	Description
"XXXX"	Publisher
"YYYY"	Release year
"ZZZ"	Release month

The "XXXX" part is a four-character abbreviation that identifies the publisher. Sega used to assign these codes (leading to the T-series numbers) but of course they don't anymore, so pick a four letter abbreviation that identifies you and is unlikely to clash.

The release date is the four-digit year and a three-letter month abbreviation (just take the first three letters of the month's name and make them uppercase).

```
    dc.b  '(C)SEGA 1992.NOV'   ; Sonic 2's copyright!
```

Game title

What it says. Usually in all uppercase, pad the field with spaces.

There are two title fields (to account for the fact that games may change name during localization), if you just keep the name the same everywhere then use it in both fields.

```
    dc.b  'CHAMPION FINGER WRESTLING'  ; Domestic title
    dcb.b $150-*, ' '                  ; (padding)
    dc.b  'CHAMPION FINGER WRESTLING'  ; Overseas title
    dcb.b $180-*, ' '                  ; (padding)
```

This is the only field where text is *not* required to be ASCII: you're allowed to use Shift-JIS characters as well.

Serial number

A field with the format "`XX YYYYYYYY-ZZ`", where:

Parts of the serial number field

Part	Description
"XX"	Software type
"YYYYYYYY"	Serial number
"ZZ"	Revision

The software type indicates what kind of software is in the cartridge (usually "`GM`", for game). The full list of known types so far is:

List of software types

Code	Type
"GM"	Game
"AI"	Aid
"OS"	Boot ROM (TMSS)
"BR"	Boot ROM (Sega CD)

The serial number is an identifier unique to the game that used to be assigned by Sega. Try to pick something that's unlikely to clash (maybe something using the publisher code from the copyright field to make it even more unlikely) and stick to it.

The revision is a two-digit number indicating which release of the game is this. The initial release is "`00`", the first patch is "`01`", next one is "`02`", etc.

```
    dc.b  'GM 00001051-00'  ; Serial number
```

ROM checksum

A 16-bit value that holds a checksum. Some games check the ROM to see if the checksum matches, and some emulators may also do it and complain if the checksum is bad, but otherwise it does nothing.

You can compute the checksum by adding up every 16-bit word from address `$000200` up to the end of the ROM (keep only the lower bits).

Device support

A list of devices or other features the game supports. One letter per device, the unused space is filled in with spaces. It's usually "`J`" (3 button controller) or "`J6`" (3 and 6 button controllers).

Note that 6-button controllers only need to be added to the list if the game actually makes uses of the extra buttons. Do *not* add it to simply indicate that the game will work with those controllers.

```
    dc.b  'J6'          ; Device support
    dcb.b $1A0-*, ' '   ; (padding)
```

The full list is this:

List of devices

Letter	Description
"J"	3-button controller
"6"	6-button controller
"0"	Master System controller
"A"	Analog joystick
"4"	Multitap
"G"	Lightgun
"L"	Activator
"M"	Mouse
"B"	Trackball
"T"	Tablet
"V"	Paddle
"K"	Keyboard or keypad
"R"	RS-232
"P"	Printer
"C"	CD-ROM (Sega CD)
"F"	Floppy drive
"D"	Download?

ROM address range

Two 32-bit values, indicating the address range covered by the cartridge. The first value is always `$000000`, while the last value is usually the ROM size minus one (unless you're using a mapper [https://plutiedev.com/beyond-4mb], then use the last address the console will see).

```
    dc.l  $000000    ; ROM start address
    dc.l  RomSize-1  ; ROM end address
```

Some common ROM sizes you may see:

Last address for common ROM sizes

Last address	ROM size
$07FFFF	512KB
$0FFFFF	1MB
$1FFFFF	2MB
$3FFFFF	4MB

RAM address range

Two 32-bit values, indicating where RAM starts and ends. In the case of the Mega Drive, they should be always $FF0000 and $FFFFFF.

```
    dc.l  $FF0000  ; RAM start address
    dc.l  $FFFFFF  ; RAM end address
```

Extra memory

This field indicates if there's extra memory (e.g. for saving progress, or simply extra RAM to toy with). The contents of this field depends on what's in the cartridge.

If there's no extra memory

If the game doesn't have extra memory, fill it with spaces.

If it has SRAM or FRAM

If the game *does* have extra RAM, the field is as follows:

SRAM field format

Size	Description
2 bytes	Always "RA"
1 byte	RAM type
1 byte	Always $20
4 bytes	Start address
4 bytes	End address

The RAM type is used to indicate whether the data is saved when turning off the console and what kind of accesses (byte or word) are allowed:

List of SRAM types

Type	Saves?	Usage
$A0	No	16-bit
$B0	No	8-bit (even addresses)
$B8	No	8-bit (odd addresses)
$E0	Yes	16-bit
$F0	Yes	8-bit (even addresses)
$F8	Yes	8-bit (odd addresses)

The start and end addresses specify the first and last addresses where SRAM shows up.

```
    dc.b 'RA',$F8,$20     ; SRAM type
    dc.l $200001          ; SRAM start address
    dc.l $20FFFF          ; SRAM end address
```

If it has EEPROM

If the game has EEPROM instead, the field should look like this (not much room to change aside from the addresses, which should indicate on which range EEPROM is accessible):

```
    dc.b 'RA',$E8,$40     ; EEPROM type
    dc.l $200001          ; EEPROM start address
    dc.l $200001          ; EEPROM end address
```

Sadly, this doesn't include information about the way the EEPROM is wired, and different publishers opted for different ways to wire it.

Modem support

If the modem is not supported, pad with spaces.

If the modem *is* supported, it's a field with the format "`MOxxxxyy,zww`" (yes, comma), where:

Parts of the modem field

Part	Description
"xxxx"	Publisher
"yy"	Game number
"z"	Version
"ww"	Region and microphone

The publisher is the same one as in the copyright field.

The "`yy,z`" make a game number and version. Games with the same "number" are compatible i.e. can talk to each other, *even if they come from completely different publishers*. The "version" is used to distinguish between them ("`0`" for the original, then "`1`", "`2`", etc.). This value was assigned by Sega of Japan back then.

The "`ww`" part indicates region support and whether the game supports the microphone (yes, voice chat in the early '90s). The possible values are:

Possible support values in the modem field

Value	Japan	Overseas
00	Yes (no mic.)	No
10	Yes (with mic.)	No
20	No	Yes (no mic.)
30	No	Yes (with mic.)
40	Yes (no mic.)	Yes (no mic.)
50	Yes (with mic.)	Yes (with mic.)
60	Yes (no mic.)	Yes (with mic.)
70	Yes (with mic.)	Yes (no mic.)

"`80`" and "`90`" are also mentioned as possible values, but they aren't described so they may have been reserved.

Iwis says

Some games seemed to use a point instead of a comma (i.e. "`MOxxxxyy.zww`") so it's possible both were valid.

Region support

A list of all the regions the game supports (one letter per region). This field had two formats over the Mega Drive's lifespan: the original format (we'll call it "old style") is the most well known and is one letter per region. The later format (we'll call it "new style") uses a single character to store information about all regions.

We recommend sticking to the original "old style" format in homebrew.

Old style

The most well known format, it stores one letter per region. If not all the regions are used then pad the field with spaces. The regions can come in any order. To support all three regions just use `"JUE"`.

```
    dc.b    "JUE"    ; All regions
```

If e.g. only Japanese systems are supported (note the spaces):

```
    dc.b    "J  "    ; Japan-only
```

The regions are these:

List of regions

Letter	Region
"J"	Japan
"U"	Americas
"E"	Europe

If you're writing code to check a game's region, bear in mind that in a few cases the spaces are in the middle or beginning, not just at the end! Always check all three bytes.

New style

At some point Sega switched to a different region format. This format uses a single character (padding the rest with spaces). The character in question is a hexadecimal digit… in ASCII (yeah, really).

The digit in question is a 4-bit value where every bit represents a supported region:

Format of region bitfield (new style format)

Bit	Weight	Region	Coverage
Bit 0	+1	Domestic, 60Hz	Japan, South Korea, Taiwan
Bit 1	+2	Domestic, 50Hz	—
Bit 2	+4	Overseas, 60Hz	North America, Brazil
Bit 3	+8	Overseas, 50Hz	Europe, Hong Kong

All possible combinations are as follows:

List of region combinations (new style format)

Character	Japan	(invalid)	Americas	Europe
"0"	No	No	No	No
"1"	Yes	No	No	No
"2"	No	Yes	No	No
"3"	Yes	Yes	No	No
"4"	No	No	Yes	No
"5"	Yes	No	Yes	No
"6"	No	Yes	Yes	No

"7"	Yes	Yes	Yes	No
"8"	No	No	No	Yes
"9"	Yes	No	No	Yes
"A"	No	Yes	No	Yes
"B"	Yes	Yes	No	Yes
"C"	No	No	Yes	Yes
"D"	Yes	No	Yes	Yes
"E"	No	Yes	Yes	Yes
"F"	Yes	Yes	Yes	Yes

lwis says

Yes, this means that "E" is ambiguous, since in the old style it means Europe only, but in the new style it means everywhere but Japan. In practice you're unlikely to find it in the new style (since it includes the "invalid" region for no real reason), so you can assume it really means Europe.