General Notice

When using this document, keep the following in mind:

1. This document is confidential. By accepting this document you acknowledge that you are bound by the terms set forth in the non-disclosure and confidentiality agreement signed separately and /in the possession of SEGA. If you have not signed such a non-disclosure agreement, please contact SEGA immediately and return this document to SEGA.

2. This document may include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new versions of the document. SEGA may make improvements and/or changes in the product(s) and/or the program(s) described in this document at any time.

3. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without SEGA'S written permission. Request for copies of this document and for technical information about SEGA products must be made to your authorized SEGA Technical Services representative.

4. No license is granted by implication or otherwise under any patents, copyrights, trademarks, or other intellectual property rights of SEGA Enterprises, Ltd., SEGA of America, Inc., or any third party.

5. Software, circuitry, and other examples described herein are meant merely to indicate the characteristics and performance of SEGA's products. SEGA assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples describe herein.

6. It is possible that this document may contain reference to, or information about, SEGA products (development hardware/software) or services that are not provided in countries other than Japan. Such references/information must not be construed to mean that SEGA intends to provide such SEGA products or services in countries other than Japan. Any reference of a SEGA licensed product/program in this document is not intended to state or simply that you can use only SEGA's licensed products/programs. Any functionally equivalent hardware/software can be used instead.

7. SEGA will not be held responsible for any damage to the user that may result from accidents or any other reasons during operation of the user's equipment, or programs according to this document.

NOTE: A reader's comment/correction form is provided with this document. Please address comments to :

SEGA of America, Inc., Developer Technical Support (att. Evelyn Merritt)
150 Shoreline Drive, Redwood City, CA 94065

SEGA may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

SEGA OF AMERICA, INC.
Consumer Products Division

SATURN/32X
Graphics References
ver. 2.0

Doc. # ST-124-R1-091394

READER CORRECTION/COMMENT SHEET

Keep us updated!

If you should come across any incorrect or outdated information while reading through the attached document, or come up with any questions or comments, please let us know so that we can make the required changes in subsequent revisions. Simply fill out all information below and return this form to the Developer Technical Support Manager at the address below. Please make more copies of this form if more space is needed. Thank you.

General Information:

Your Name _____ **Phone** _____

Document number ST-124-R1-091394 **Date** _____

Document name Saturn/32X Graphics References ver. 2.0

Corrections:

Chpt.	pg. #	Correction

Questions/comments: _____

Where to send your corrections:

Fax: (415) 802-3963
Attn: Manager,
Developer Technical Support

Mail: SEGA OF AMERICA
Attn: Manager,
Developer Technical Support
275 Shoreline Dr. Ste 500
Redwood City, CA 94065

REFERENCES

In translating/creating this document, certain technical words and/or phrases were interpreted with the assistance of the technical literature listed below.

1. *KenKyusha New Japanese-English Dictionary*
   1974 Edition

2. *Nelson's Japanese-English Character Dictionary*
   2nd revised version

3. *Microsoft Computer Dictionary*

4. *Japanese-English Computer Terms Dictionary*
   Nichigai Associates
   4th version

Graphics Data Book

CONTENTS

1.0 SEGA2D Format 2
- 1.1 File Layout 2
- 1.2 Header 2
- 1.3 Map Data 2
- 1.4 Page Data 3
- 1.5 Character Generator Data 6
- 1.6 Palette Data 6
- 1.7 Attribute Data 6

2.0 DGT Format 7
- 2.1 File Layout 7
- 2.2 Header 7
- 2.3 Directory 7
- 2.4 CLUT 8
- 2.5 Bit Map Data 8

3.0 RGB Format 9
- 3.1 File Layout 9
- 3.2 Header 9
- 3.3 Bit Map Data 9

4.0 SEGA3D Format 10
- 4.1 File Guidelines 10
- 4.2 File Layout 11
- 4.3 Header 12
- 4.4 Model 12
- 4.5 Material 13
- 4.6 Slip 14
- 4.7 Object 14
- 4.8 Model Tree Configuration 15
- 4.9 Sample Data 18

5.0 SX2D Format 20
- 5.1 File Layout 20
- 5.2 Header 20
- 5.3 Map Data 21
- 5.4 Character Data 21
- 5.5 Palette Data 22
- 5.6 Attribute Data 22

6.0 DGT2 Format 23
- 6.1 File Layout 23
- 6.2 Image Size 23
- 6.3 CLUT 23
- 6.4 Bit Map Data 24

1.0 SEGA2D Format

The SEGA2D format is the SEGA standard scroll data format for Saturn.

1.1 File Layout

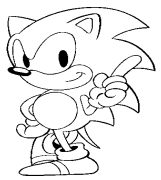
Offset	Size	Function
$000	$100	Header
can be changed	can be changed	Map Data
can be changed	can be changed	Page Data
can be changed	can be changed	CG Data
can be changed	can be changed	Palette Data
can be changed	can be changed	Attribute Data

1.2 Header

Offset	Size	Explanation
$00	$10	Identifier="SEGA_32BIT2DSCR"+$1A ("_" is space =$20
$10	Long	Map data offset address
$14	Long	Map data size (byte)
$20	Long	Page data offset address
$24	Long	Page data size (byte)
$30	Long	CG data offset address
$34	Long	CG data size (byte)
$40	Long	Palette data offset address
$44	Long	Palette data size (byte)
$50	Long	Attribute data offset address
$54	Long	Attribute data size (byte)
$58	$A8	Empty address, all $00

1.3 Map Data

Offset	Size	Function
$00	Word	Horizontal page number (page size)
$02	Word	Vertical page number (page size)
$04	$0C	Dummy data, all $FF
$10	Word x Page Number	Page number



1.4 Page Data

Offset	Size	Function
$00	Long	Character control data
$04	Word	Pattern name auxiliary data
$06	$OA	Dummy data, all $FF
$10	Word or Long	Pattern name data

Character Control Data
MSB

		Reserved							No. of Character Colors			Reserved			
–	–	0	0	0	0	0	0	–	CN2	CN1	CN0	0	0	0	SZ

LSB

	Reserved				Reserved					Reserved				Reserved	
–	0	0	0	–	0	0	0	–	–	0	0	–	–	0	0

- CN: No. of character colors 000=16 colors, 001=256 colors
- SZ: Character size 0=1x1 cell, 1=2x2 cells

Pattern Name Auxiliary Data
MSB

		Unused				Pattern Name Auxiliary Data									
PNB	NSM	–	–	–	–	?	?	?	?	?	?	?	?	?	?

- PNB: Pattern name data size 0=1 word, 1=2 words
- NSM: Character No. auxiliary mode 0=10 bits, 1=12 bits
- When the pattern name data size is 2 words, the character No. auxiliary mode and pattern name auxiliary data are invalid.
- Please refer to the details of pattern name auxiliary data presented hereafter.

When the Pattern Name Data Size=2 Words

Pattern Name Data

MSB

Flip		Special Function		Unused					Palette No.						
HF	VF	Pr	CC	–	–	–	–	–	6	5	4	3	2	1	0

LSB

	Character No.														
–	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

- HF: Left/right flip flag
- VF: Up/down flip flag
- Pr: Special priority bit
- CC: Special color calculation bit
- The unused bit (--) is ignored.

When the Pattern Name Data Size=1 Word

There are four possibilities when the character size (1x1 cell/2x2 cells) and character No. auxiliary mode (10 bits/12 bits) are combined.

- **Character Size=1x1 Cell and Character No. Auxiliary Mode=10 bits**

Pattern Name Data

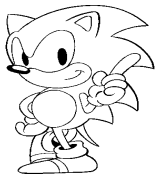
MSB LSB

Palette No.				Flip		Character No.									
3	2	1	0	HF	VF	9	8	7	6	5	4	3	2	1	0

Pattern Name Auxiliary Data

MSB LSB

Unused						Special Function		Palette No.			Character No.				
–	–	–	–	–	–	Pr	CC	6	5	4	14	13	12	11	10



Character Size=1x1 Cell and Character No. Auxiliary Mode=12 bits

Pattern Name Data

MSB LSB

Palette No.				Character No.											
3	2	1	0	11	10	9	8	7	6	5	4	3	2	1	0

Pattern Name Auxiliary Data

MSB LSB

Unused						Special Function		Palette No.			Character No.				
–	–	–	–	–	–	Pr	CC	6	5	4	14	13	12	–	–

- **Character Size=2x2 Cells and Character No. Auxiliary Mode=10 bits**

Pattern Name Data

MSB LSB

Palette No.				Flip		Character No.									
3	2	1	0	HF	VF	11	10	9	8	7	6	5	4	3	2

Pattern Name Auxiliary Data

MSB LSB

Unused						Special Function		Palette No.			Character No.				
–	–	–	–	–	–	Pr	CC	6	5	4	14	13	12	1	0

- **Character Size=2x2 Cells and Character No. Auxiliary Mode=12 bits**

Pattern Name Data

MSB LSB

Palette No.				Character No.											
3	2	1	0	13	12	11	10	9	8	7	6	5	4	3	2

Pattern Name Auxiliary Data

MSB LSB

Unused						Special Function		Palette No.			Character No.				
–	–	–	–	–	–	Pr	CC	6	5	4	14	–	–	1	0

1.5 Character Generator Data

Offset	Size	Function
$00	$10	Dummy data, all $FF
$10	Can be changed	CG data

1.6 Palette Data

Offset	Size	Function
$00	Word	First palette No.
$02	$0E	Dummy data, all $FF
$10	Word x Palette No.	Color code (five bits each for R, G, B)

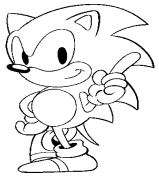
Color Code

MSB LSB

	B					G					R				
–	4	3	2	1	0	4	3	2	1	0	4	3	2	1	0

1.7 Attribute Data

Offset	Size	Function
$00	Word x Character No.	Attribute Data



2.0 DGT Format

The DGT format is the SEGA standard graphic data format for the index color mode.

2.1 File Layout

Offset	Size	Function
$000	$100	Header
$100	$20	Directory
$120	Can be changed	CLUT (Color Look Up Table)
Undefined	Can be changed	Bit map data

2.2 Header

Offset	Size	Explanation
$02	Word	Header size=$100
$07	Byte	Directory entry No.=$01
$08	Long	File size excluding header
$10	$10	Identifier="DIGITIZER_3_Ver2" ("_" is space=$20)
$20	$E0	Dummy data, all $00

- All empty addresses are $00.

2.3 Directory

Offset	Size	Explanation
$00	Long	Directory size=$20+CLUT size (can be changed)
$04	Word	Horizontal direction display position (normally $0000)
$06	Word	Vertical direction display position (normally $0000)
$08	Word	Horizontal direction image size
$0A	Word	Vertical direction image size
$10	$10	Directory name (file name excluding name extension)

- The "Directory Name" empty address is $20.
- The empty addresses excluding "Directory Name" are all $00.

2.4 CLUT

Offset	Size	Function
$00	Long x Palette No.	CLUT data

- One palette is configured as "Long x 16 colors."
- If even one color is used, one palette's worth data must be reserved.

CLUT Data
MSB

	Palette No.														
–	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

LSB

	Color Code														
?	B0	G0	R0	B4	B3	B2	B1	G4	G3	G2	G1	R4	R3	R2	R1

- The color code with respect to the palette numbers that are not used for bit map data even if there is a CLUT becomes $FFFF. This way, palette numbers not used from CLUT can be identified.

2.5 Bit Map Data

Offset	Size	Function
$00~	Word x Image Size	Palette No. for each pixel

3.0 RGB Format

The RGB format is the SEGA standard graphic data format for the RGB color mode.

3.1 File Layout

Offset	Size	Function
$000	$100	Header
$100	Can be changed	Bit map data

3.2 Header

Offset	Size	Explanation
$00	$10	Identifier="SEGA_32BITGRAPH"+$1A ("_" is space =$20
$10	Long	Dummy data=$FFFF FFFF
$14	Word	Horizontal direction display position (normally $0000)
$16	Word	Vertical direction display position (normally $0000)
$18	Word	Horizontal direction image size
$1A	Word	Vertical direction image size
$1C	Long	Empty address=$0000 0000
$20	$E0	Empty address=$00 x size

3.3 Bit Map Data

Offset	Size	Function
$00~	Byte	R color value (8 bit)
$01~	Byte	G color value (8 bit)
$02~	Byte	B color value (8 bit)

4.0 SEGA3D Format

The SEGA3D Format is the SEGA standard SATURN 3D data format (text).

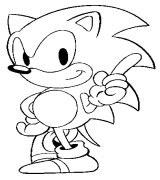
4.1 File Conventions

There are provisions for SEGA3D format files for the following information.

Reserved Code	Description
{ }	Data control construction
( )	Data collection
,	Data limit
;	Comment from now to the end of the line
" "	File name, comment character series (notations depend on process system)

Data Notation	Description
123	Decimal
0x123	Hexadecimal
abcde	Character series (20H~7FH)

Reserved Word	Description
model	Control construction
vertices	Control construction
polygons	Control construction
slips	Control construction
object	Control construction
material	Control construction
type	Control data
texture	Control data
color	Control data
gour_offset	Control data
gour_color	Control data
direction	Control data
comment	Control data



4.2 File Layout

A SEGA3D file consists of the following parts. With the exception of the header, the various parts can be freely laid out. In addition, when necessary, more than one unit of a part can exist or a part can be deleted.

Header
Model
Material
Slip
Object

Example of Only Modelling Data

Header
Model

Example of Material Type

Header
Model
Material
Material
Slip

Example of Multiple Model Types

Header
Model
Model
Object

Example of Complex Model Types

Header
Model
Model
Material
Material
Material
Slip
Slip
Object

4.3 Header

```
SEGA 3D
```

The header is a fixed character series used as an identifier, and must always exist at the beginning of a file.

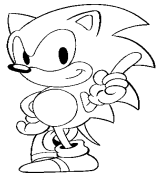
4.4 Model

```
model           modelname
{
        ;meter
        vertices n ;number of vertices
        {
                (x0,y0,z0),(nx0,ny0,nz0)                ;0
                (x1,y1,z1),(nx1,ny1,nz1)                ;1
                . . . . . .
                . . . . . .
        }
polygons n ;number of polygons
        {
                (v0,v1,v2,v3),  (Nx,Ny,Nz),(Zpos)               ;0
                . . . . . .
                . . . . . .
        }
}
```

- One vertices construction and polygons construction co-exist in one model.
- Vertices are indexed by the number of vertices (n); the index numbers (0~) are the vertex numbers.
- Polygons are indexed by the number of surfaces; the index numbers (0~) are the surface numbers.

Data	Description	Initial Value
modelname	Model name	File name
(x_n,y_n,z_n)	Each vertex coordinate	
(nx_n,ny_n,nz_n)	Each vertex's normal vector (-1.0<value<1.0)	(0,0,0)
$(v_?,v_?,v_?,v_?)$	Vertex numbers constituting the surface	
(Nx ,Ny,Nz)	Surface's normal vector	(0,0,0)
(Zpos)	Zsort standard [0=AVE: center, 1=MIN:nearest, 2=MAX:farthest]	(AVE)

- The number of vectors constituting a surface are set at four. Therefore, objects with five or more angles cannot be displayed as one polygon. In addition, triangles are expressed as (v0,v1,v2,v3) where v2 and v3 are the same coordinate.



- The vertex numbers constituting a surface are numbered in the clockwise direction.

4.5 Material

```
material        materialname
{
        type                    (TYPE0|TPE1|...)
        texture                 "texturefilename"
        color                   ColorCode
        gour_offset             offset
        gour_color              (R1,G1,B1),(R2,G2,B2)...
        direction               0,1,2,3
        comment                 "strings"
```

Data items are configured when necessary.

Data	Description	Initial Value
materialname	Material name	File name
$TYPE_n$	Material Type [0x00=NULL: None 0x01=FLAT: Flat shading 0x02=TEXT: Texture map 0x04=GOUR: Gouraud shading 0x08=MESH: Mesh 0x10=TRAN: Half transparency 0x20=SHAD: Half translucency 0x80=WIRE: Wire frame] Multiple specifications are connected by ":"	NULL
'*textureFileName*'	Texture map file name	None
ColorCode	Non-texture color code [RGB each 5 bits]	0x7fff
offset	Offset data that changes the color code's translucency [0x00~0x1f].	0x10
(R1,G1,B1)	Translucency of the four vertices that display gouraud shading.	0x10
0,1,2,3	Handles each texture map vertex.	0,1,2,3
"*strings*"	Comment character series written in the material. However, indentation in the middle of a sentence or existance of characters such as " " or ";" are not allowed.	None

4.6 Slip

```
slips           modelname
{
        materialname0
        materialname1
        .....
}
```

A slip is configured when material exists in the model surface. A material name then exists for each surface and corresponds to the surface number.

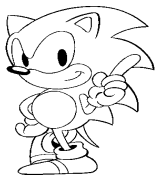
Data	Description	Initial Value
modelname	Corresponding model name	File name
$materialname_n$	Corresponds to the materialname	None

4.7 Object

```
object          objectname
{
        modelname0,(xs,ys,zs),(xd,yd,zd),  (xt,yt,zt),(xi,yi,zi)
        {
        }
        modelname1,(xs,ys,zs),(xd,yd,zd),  (xt,yt,zt),(xi,yi,zi)
        {
                modelname2,(xs,ys,zs),(xd,yd,zd),(xt,yt,zt),(xi,yi,zi)
                        {
                        modelname4,(xs,ys,zs),(xd,yd,zd),(xt,yt,zt),(xi,yi,zi)
                        {
                        }
                }
                modelname3,(xs,ys,zs),(xd,yd,zd),(xt,yt,zt),(xi,yi,zi)
                {
                }
        }
        .....
        .....
}
```

An object is configured when multiple models exist.

Data	Description	Initial Value
objectname	Object name	File name
modelname	Corresponding model name	
(xs,ys,zs)	Scale (ratio)	(1.0,1.0,1.0)
(xd,yd,zd)	Rotation [Angle: -180~+179]	(0,0,0)
(xt,yt,zt)	Translation [coordinates]	(0,0,0)
(xi,yi,zi)	Flip processing [0=OFF: None, 1=ON: Yes]	(0,0,0)



4.8 Model Tree Construction

When a flip processing flag appears for the first model, that flag is ignored and flip processing is only done for subsequent models that appear.

```
object   objectname
{
         modelname0,(1.0,1.0,1.0),(0,0,0),(0,0,0),(1,0,0) Flip processing invalid
         {
         }
         modelname1,(1.0,1.0,1.0),(0,0,0),(0,0,0),(1,1,1) Flip processing invalid
         {
         }
         modelname1,(1.0,1.0,1.0),(0,0,0),(0,0,0),(1,0,0) Flip processing valid
         {
         }
}
```

If data with the same model name exists, the newer model inherits all of the data that is under the older model of the same name.

```
object   objectname
{
         modelname0,(xs,ys,zs)...
         {
         }
         modelname1,(xs,ys,zs)...
         {
                  modelname2,(xs,ys,zs)...
                  modelname3,(xs,ys,zs)...
         }
         modelname1,(xs,ys,zs)...
         {
         }
}
```

Model inheritance image when an object like the one above is read.

```
         modelname0
         modelname1
                  modelname2
                  modelname3
         modelname1
                  modelname2  } The models under modelname1 are automatically inherited.
                  modelname3  }
```

The flip processing flag is not inherited, even though models are inherited. To create an object, like a mirror plane, flip processing must be specified for all inherited models.

```
object    objectname
{
          modelname1,(1.0,1.0,1.0),(0,0,0),(0,0,0),(0,0,0)
          {
                    modelname2,(1.0,1.0,1.0),(0,0,0),(0,0,0),(0,0,0)
                    modelname3,(1.0,1.0,1.0),(0,0,0),(0,0,0),(0,0,0)
          }
          modelname1,(1.0,1.0,1.0),(0,0,0),(0,0,0),(1,0,1) Flip processing: Yes
          {
          }
}
modelname1 inheritance image when an object like that above is read.
          modelname1
                    modelname2
                    modelname3
          modelname1,(1.0,1.0,1.0),(0,0,0),(0,0,0),(1,0,0)
                    modelname2,(1.0,1.0,1.0),(0,0,0),(0,0,0),(0,0,0)
                    modelname3,(1.0,1.0,1.0),(0,0,0),(0,0,0),(0,0,0)

The modelname1 flip processing data is not transferred to modelname2 and modelname3.
```

Models cannot use the same model name in a tree construction.

```
object    objectname
{
          modelname0,(xs,ys,zs)...
          modelname1,(xs,ys,zs)...
          {
                    modelname2,(xs,ys,zs)...
                              modelname1,(xs,ys,zs)... Creates and error when read.
                    modelname3,(xs,ys,zs)...
          }
}
```


If lower hierarchy models exist under data with the same model name, the lower models will be ignored.

```
object    objectname
{
          modelname1,(xs,ys,zs)...
          {
                    modelname2,(xs,ys,zs)...
                    modelname3,(xs,ys,zs)...
          }
          modelname1,(xs,ys,zs)...
          {
                    modelname4,(xs,ys,zs)...
          }
}
```

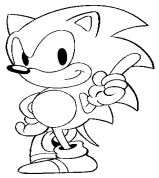
Model name 1 inheritance image when an object like the one above is read.

```
          modelname1
                    modelname2
                    modelname3
          modelname1
                    modelname2
                    modelname3
```

In this case, modelname4 is ignored.

4.9 Sample Data

```
1: SEGA 3D
2: ;SEGA 3D Format Sample Data
3: model CUBE
4: {
5:          vertices 8
6:          {
7:                    (1.000, 5.000, 1.000),(-0.577,  0.577, -0.577)      ;0
8:                    (1.000, 5.000, 5.000),(-0.577,  0.577,  0.577)      ;1
9:                    (1.000, 1.000, 5.000),(-0.577, -0.577,  0.577)      ;2
10:                   (1.000, 1.000, 1.000),(-0.577, -0.577, -0.577)      ;3
11:                   (5.000, 5.000, 5.000),( 0.577,  0.577,  0.577)      ;4
12:                   (5.000, 1.000, 5.000),( 0.577, -0.577,  0.577)      ;5
13:                   (5.000, 5.000, 1.000),( 0.577,  0.577, -0.577)      ;6
14:                   (5.000, 1.000, 1.000),( 0.577, -0.577, -0.577)      ;7
15:         }
16:         polygons 6
17:         {
18:                   (3, 2, 1, 0),(-1.000,  0.000,  0.000),(AVE)  ;0
19:                   (2, 5, 4, 1),( 0.000,  0.000,  1.000),(AVE)  ;1
20:                   (5, 7, 6, 4),( 1.000,  0.000,  0.000),(AVE)  ;2
21:                   (7, 3, 0, 6),( 0.000,  0.000, -1.000),(AVE)  ;3
22:                   (7, 5, 2, 3),( 0.000, -1.000,  0.000),(AVE)  ;4
23:                   (0, 1, 4, 6),( 0.000,  1.000,  0.000),(AVE)  ;5
24:         }
25: }
26: material mate00
27: {
28:         type          (FLAT)
29:         Color         0x7fec
30:         gour_offset   0x4210
31:         gour_color    (0x10,0x10,0x10),(0x10,0x10,0x10),
                          (0x10,0x10,0x10),(0x10,0x10,0x10)
32:         direction     0,1,2,3
33: }
34: material mate01
35: {
36:         type          (FLAT)
37:         Color         0x374e
38:         gour_offset   0x4210
39:         gour_color    (0x10,0x10,0x10),(0x10,0x10,0x10),
                          (0x10,0x10,0x10),(0x10,0x10,0x10)
40:         direction     0,1,2,3
41: }
42: slips  CUBE
43: {
44:         mate00
45:         mate00
46:         mate00
47:         mate01
48:         mate00
49:         mate00
```



```
50: }
51: object entire object
52: {
53:         CUBE,  (1.000, 1.000, 1.000),(0.000, 0.000, 0.000)
                   (0.000, 0.000, 0.000),(0.000, 0.000, 0.000)
54:         {
55:         }
56: }
```

5.0 SX2D Format

The SX2D format is the SEGA standard for the 32X scroll data format. The Macintosh file type is "SX2D."

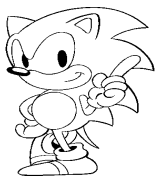
5.1 File Layout

Offset	Size	Function
$000	$100	Header
can be changed	can be changed	Map Data
can be changed	can be changed	Character Data
can be changed	can be changed	Palette Data
can be changed	can be changed	Attribute Data

5.2 Header

Offset	Size	Explanation
$00	Long	Identifier = "Sega_Super32X_2D" ("–" space = $20)
$10	Long	Map Data Offset Address
$14	Long	Map Data Size (bytes)
$20	Long	Character Data Offset Address
$24	Long	Character Data Size (bytes)
$30	Long	Palette Data Offset Address
$34	Long	Palette Data Size (bytes)
$40	Long	Attribute Data Offset Address
$44	Long	Attribute Data Size (bytes)
$48	$b8	Empty Address, all $00

- The data size equals 0 when there is no data.



5.3 Map Data

Offset	Size	Function
$00	Word	Horizontal size (pattern name units)
$02	Word	Vertical size (pattern name units)
$04	Word x number of pattern names	Pattern name data

Pattern Name Data

MSB LSB

Flip		Special Function		Character Number											
HF	VF	Pr	–	11	10	9	8	7	6	5	4	3	2	1	0

- HF: left-right flip flag, VF: up-down flip flag
- Pr: Special priority bit
- Unused bit (--) is ignored.

5.4 Character Data

Offset	Size	Function
$00	Word	Character mode = "PP"/"DC"
$02	Can be changed	Character data

- Character Mode : "PP" = Packed Pixel Mode (256 colors)
  "DC" = Direct Color Mode (32768 colors)

Character Data

Bit map data of 16 dot X 16 dot units.
In the packed pixel mode (character color number 256 colors), the bit map is configured by the palette code.

Character data per one dot.

MSB LSB

Palette Code							
7	6	5	4	3	2	1	0

- In the direct color mode, the bit map is configured by the color code.

Character data per one dot.

MSB LSB

Color Number														
*	B4	B3	B2	B1	B0	G4	G3	G2	G1	G0	R4	R3	R2	R1	R0

- * = Priority bit

5.5 Palette Data

Offset	Size	Function
$00	Word X 256	Color Code (R, G, B, each 5 bits)

- Data does not exist when in the direct color mode.

Color Code

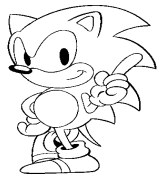
MSB

	B					G					R				
*	4	3	2	1	0	4	3	2	1	0	4	3	2	1	0

- * = Priority bit

5.6 Attribute Data

Offset	Size	Function
$00	Word X Number of Characters	Attribute Data



6.0 DGT2 Format

The DGT2 format is the new SEGA standard graphic data format (handles 32,768 colors). The Macintosh file type is "DGT2."

6.1 File Layout

Offset	Size	Function
$00	Word	Identifier = "PP" / "DC" / "RL"
$02	$04	Image Size
$06	Can be changed	CLUT (exists when needed)
Can be changed	Can be changed	Bit Map Data

Identifier
"PP" = Packed Pixel Mode
"DC" = Direct Color Mode
"RL" = Run Length Mode

6.2 Image Size

Offset	Size	Explanation
$00	Word	H image size
$02	Word	V image size

6.3 CLUT

Offset	Size	Function
$00	Word X 256	CLUT Data

- = Exists in the "PP" Mode and "RL" Mode.

MSB LSB

CLUT Data															
*	B4	B3	B2	B1	B0	G4	G3	G2	G1	G0	R4	R3	R2	R1	R0

- * = Any setting is okay

6.4 Bit Map Data

Offset	Size	Function
$00 ~	Can be changed	Bit map data

"PP" Mode

MSB LSB

CLUT Index Number							
7	6	5	4	3	2	1	0

"DC" Mode

MSB LSB

Color Number														
*	B4	B3	B2	B1	B0	G4	G3	G2	G1	G0	R4	R3	R2	R1	R0

- * = Any setting is okay

"RL" Mode

MSB LSB

Continuous Length								CLUT Index Number							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0