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

External Specification Document

Saturn Stream System

Doc. #ST-98-031194

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-98-031194 **Date** _____

Document name External Specification Document: Saturn Stream System

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

History of Modifications

1994-02-21

Access Image Diagram	Modification
• StmArea	Addition
• StmSct	Addition
• StmKey	Change
• STM_SetCdbufFull	Addition
• STM_OpenResi	Addition
• STM_ConnectCdbuf	Addition
• STM_MoveCdbuf	Addition
• STM_StartTrans	Addition
• STM_SetTrFad	Addition
• STM_OpenFid	Change
• STM_OpenFrange	Change
• STM_SetKey	Change
• STM_GetInfo	Change
• STM_EraseCdbuf	Change
• STM_GetSctInfo	Change
• STM_GetErrStat	Change

- The stream read program modification in accordance with the STM_OpenFid change.
- The transmission function specification change and the corresponding change to the 5. (2) Transmission Function Example.
- Make it so that the word No. and not the byte No. are used for the data No.

Stream System Review and Other Items of Study

1. Function Access from Remaining CD Buffer Capacity
   Add the function STM_SetCdbufFull.

   This function allows the registered function to be accessed when the capacity remaining in the CD buffer is less than the set value.

2. Resident Stream Handling
   Add the functions STM_OpenResi and STM_SetTrFad.

   Normally, the data that is read into the CD buffer is transmitted to the host area when the play position reaches the transmission start FAD. Stream data that has been opened by STM_OpenResi, however, is resident in the CD buffer even after being transmitted to the host area. This stream is called a resident stream. When the transmission start FAD is reset (STM_SetTrFad) after residence, data is again transmitted when the play position reaches the set transmission start FAD.

   This function can handle the steam as a SIMM file or SCSI file in the same way as a file on a CD is normally handled. In this case, however, the stream is not resident on the CD buffer, so data is read from the SIMM or SCSI each time the stream is accessed.

   For details, refer to External Specifications, Section 4.4 Resident Stream.

3. Acquiring Actual Data Size After Data Transmission Has Begun
   For the transmission function, there is a high possibility that the actual data size acquisition functions (STM_SctToByte and STM_ByteToSct) will be used, so add the function STM_StartTrans. In addition, delete the transmission address from the transmission function argument and then make acquisition as the function value of STM_StartTrans. The actual data size acquisition function is to be used before STM_StartTrans is accessed.

4. Setting and Acquiring the Data No.
   For the stream system, the word No. and not the byte No. is to be used for the data number.

5. Function Arguments
   Changes were made for functions with many arguments so that performance does not degrade when a struct is made into an argument.

1.0 Summary

This document is the external specifications for a library that will allow streams (interleaved files, etc.) on CD to be efficiently read

1.1 Library Configuration

The library configuration for CD-related items is shown in Figure 1.1.

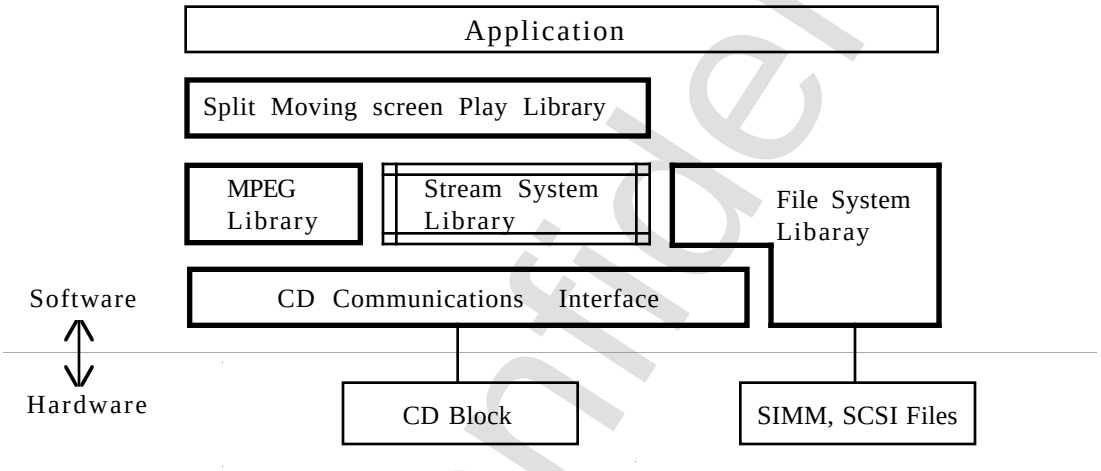


Figure 1.1 CD Related Library Configuration

1.2 Overview of Stream System Functions

(1) Supports CD-ROM XA Level File Access

- Supports access to mode 2 sector that used a sub header. (Access to mode 1 sector is also possible.)

(2) Supports a Variety of Transfer Methods

- The stream data on CD can be transferred to the main CPU area.
- Registering functions makes it possible to process data while reading stream data from the CD.
- The stream data read into the CD buffer can be freely manipulated.

(3) Supports Access By File ID

- File access is allowed via the file ID (sequence number in the directory) using the directory management function similar to the file system.

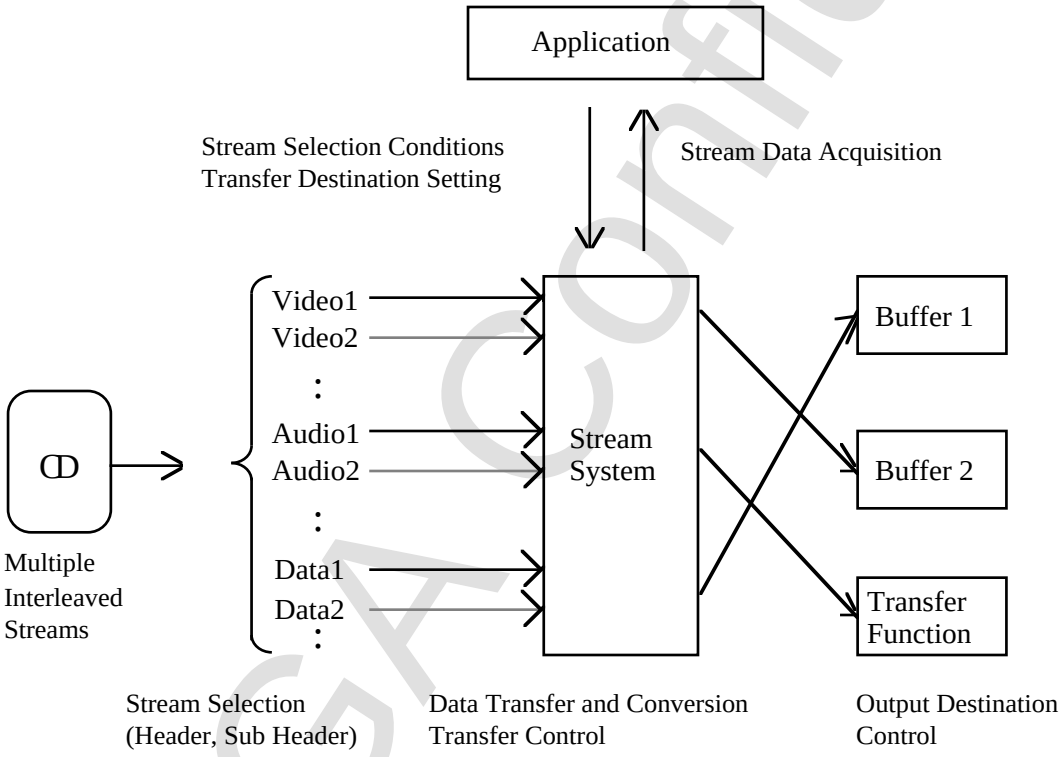
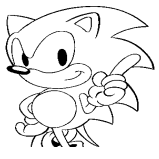


Figure 1.2 Overview Diagram of Stream System



2.0Definition of Terminology and Abbreviations

Table 2.1 Terminology Chart

Word	Meaning
Stream	The flow of logically connected data that has been classified via a sub header.
Stream Key	The key when a stream is read into a CD buffer. The stream key comes from the frame address range, file number (FN), channel number (CN), sub mode (SM), and coding information (CI). With respect to the sub mode (SM) and coding information (CI), the mask pattern comparison pattern can be specified. In this case the sector that is ($\left\{\begin{matrix}SM\\CI\end{matrix}\right\}$ & mask pattern == comparison pattern) is the sector that is read.
Stream Group	A collection of streams.
End Stream	The stream in a stream group that is the last stream to be played.
Loop Start Stream	The return stream after a stream group's end stream has been played back.
Transfer Start FAD	The pick up position for starting the transfer of data read into the CD buffer to the program buffer, etc.
Transfer Gate	The gate when data read into the CD buffer is transferred to the program buffer, etc. Closing this gate allows stream data to be accumulated in the CD buffer.

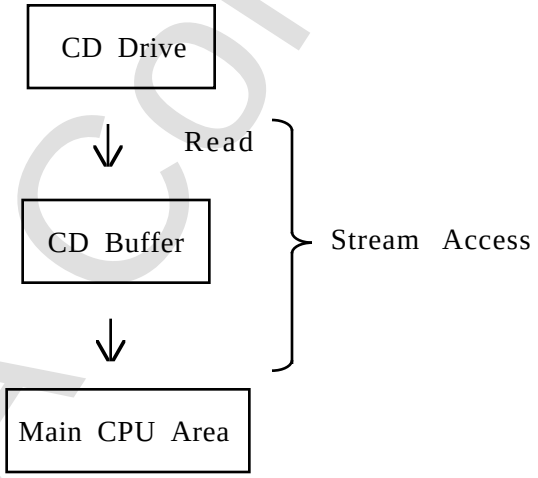
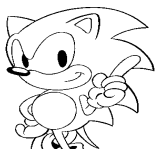


Figure 2.1 Stream System Data Flow

Table 2.2 Abbreviation Chart

Abbreviation	Meaning
GFS	general file system
STM	stream
StmGrpHn	stream group handle
StmHn	stream handle
TrMode	transfer mode
bn	buffer number
ci	coding information
cn	channel number
fad	frame address
fid	file ID
fn	file number
fname	file name
loopstm	loop stream
plyarea	play area
sinfo	sector information
sm	sub mode
sn	sector number
stype	sector type

- For other terminology, use the meanings given for the CD communications interface and the file system.
- For details regarding the directory, refer to File System (GFS) Directory.



3.0 Module Configuration

Following is the module configuration as seen from the application.

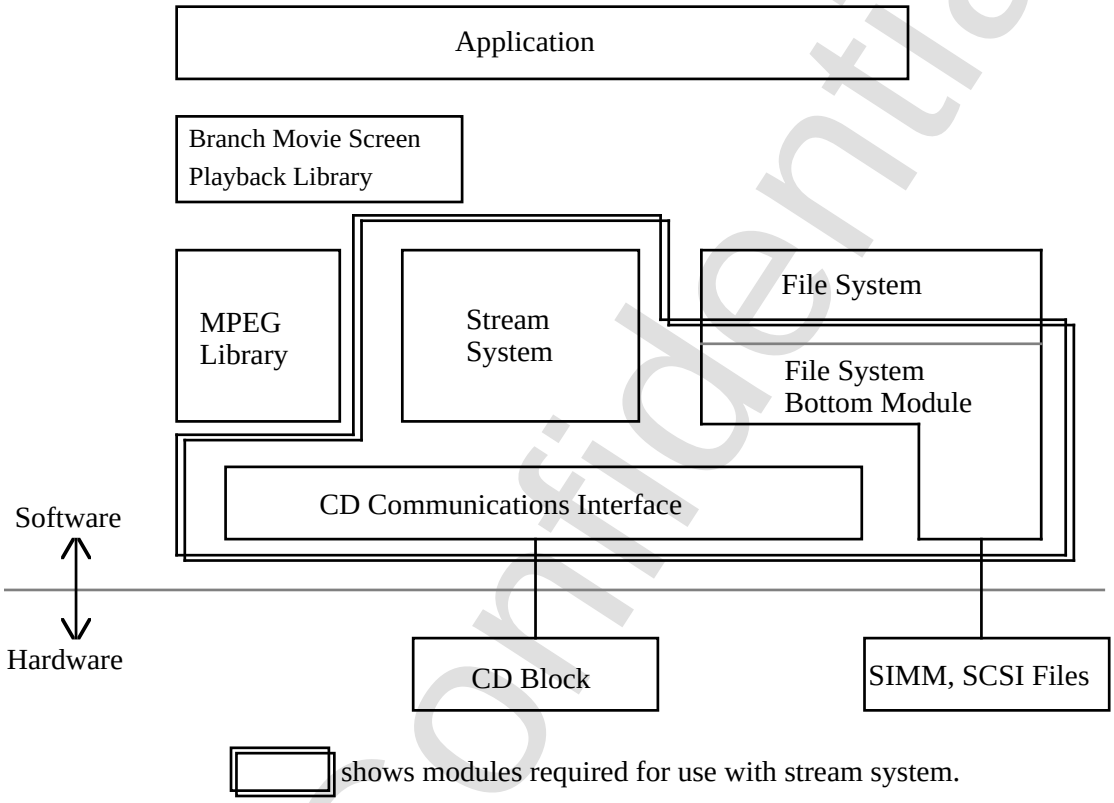


Figure 3.1 Module Configuration Diagram

The file system library and CD communications interface library are necessary to use the stream system library.

Each library uses the following global symbols. The application program must not use these symbols.

Table 3.1 Symbol Name for Each Library

Abbreviation	Meaning
Stream system	ST*_*
File system	GF*_*, GP*_*
CD communications interface	CD*_*

4.0 Overview of Stream Access

4.1 Streams and Stream Groups

(1) Stream

Sector groups with the same sub headers (FN, CN, SM, CI) and that undergo basically the same processing are called a stream. These sectors do not need to be physically contiguous.

(2) Stream Group

Interleaving and recording multiple streams as is done for audio and visual allows related streams to be synchronized and accessed at the same time. A collection of such related streams is called a stream group.

4.2 Stream Area

Stream areas are stipulated using the following methods.

(1) Opening the Stream via File

The file ID can be specified to open the stream (STM_OpenFid). In this case, the stream area is from the start frame address in the file to the end frame address.

The end frame address is actually calculated from the total number of sectors in the start frame address and file. In the case of interleaved files, the end frame address is calculated as having been recorded as a defined interleave factor (set interleave).

(2) Opening the Stream by Directly Specifying the Frame Address Area

The user can open the stream by directly specifying the frame address area (STM_OpenFrange). In this case, the specified area becomes the stream area.

The frame address area specifies the first frame address and physical sector number.

4.3 Stream Access Procedure

The following procedure is used to access streams.

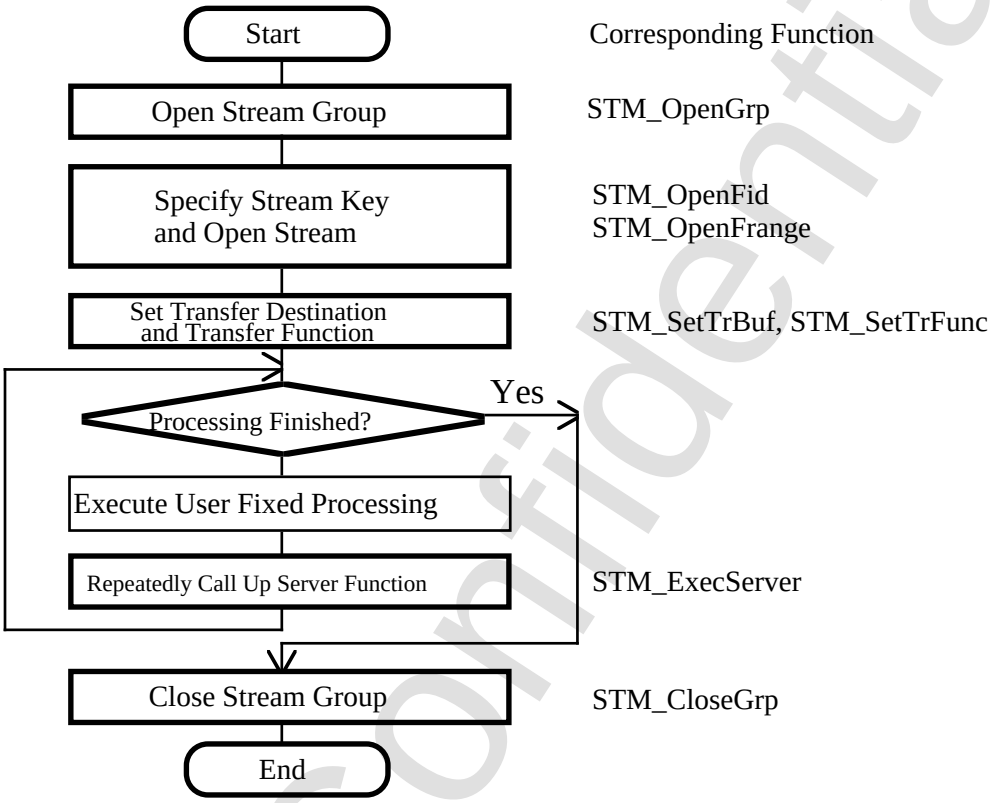


Figure 4.1 Stream Access Procedure

The stream access image is shown below.

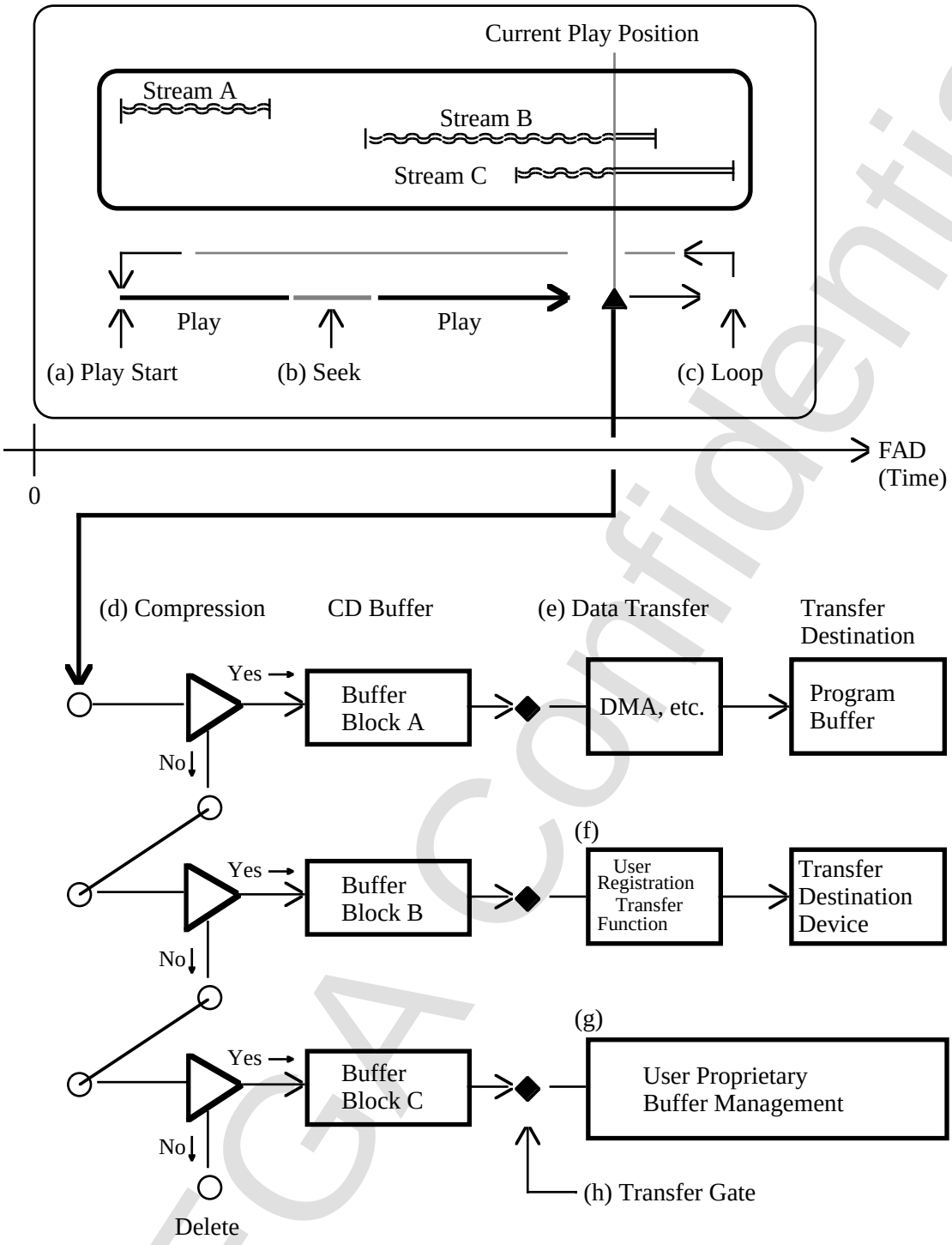


Figure 4.2 Overview of Stream Access

(a) Play Start

Of the streams in the stream group, the start FAD begins playing from stream A, which is the front most stream.

(b) Seek

When the playing stream has finished, the pickup moves to the start position of the next stream (stream B) and begins playing.

(c) Loop

When the last stream (stream C) has finished playing, the pickup moves to the start position of the loop start stream (the default is the front most stream) and begins playing.

(d) Filter and Buffer Block

Buffer blocks are allotted one-to-one with the streams.

Sector data that meets the stream key conditions is stored in a buffer block. Sector data that does not meet the key conditions is sent to the next filter.

(e) Data Transfer

When server functions are called up, the transfer start position and current play position that is set for each stream is compared, and if there is a stream that has reached the transfer start position, there will be an attemp to transfer data to the selected transfer area one round at a time.

If the transfer register or DMA come in use during the transfer, the server function will end at that time, and the next test transfer will start from the next stream.

(f) User Registration Transfer Function

Registering the transfer function allows data to be transferred while it is being processed, such as the decompression of compressed data.

(g) User Proprietary Buffer Management

When the transfer area and transfer function are not specified, the application program itself can manage the CD buffer data.

(h) Transfer Gate

Closing and opening the transfer gate temporarily stops the stream flow and then allows it to flow again. When the transfer gate is closed, the stream data is accumulated in the CD buffer.

4.4 Resident Stream

When relatively short streams need to be transferred repeatedly, they can be opened as resident streams (STM-OpenResi). Using resident streams allows data to be resident in the CD buffer without requiring the same data to be repeatedly read from the CD.

(1) Reading Resident Streams

Files specified in resident streams are only read into the CD buffer once. Data that is accessed repeatedly by returning to the loop start stream is not read into the CD buffer. Therefore, during the first stream access, play must be started from before the resident stream file area.

When the resident stream is opened during stream access and the play position has passed the resident stream area, care must be taken that the data is not read into the CD buffer.

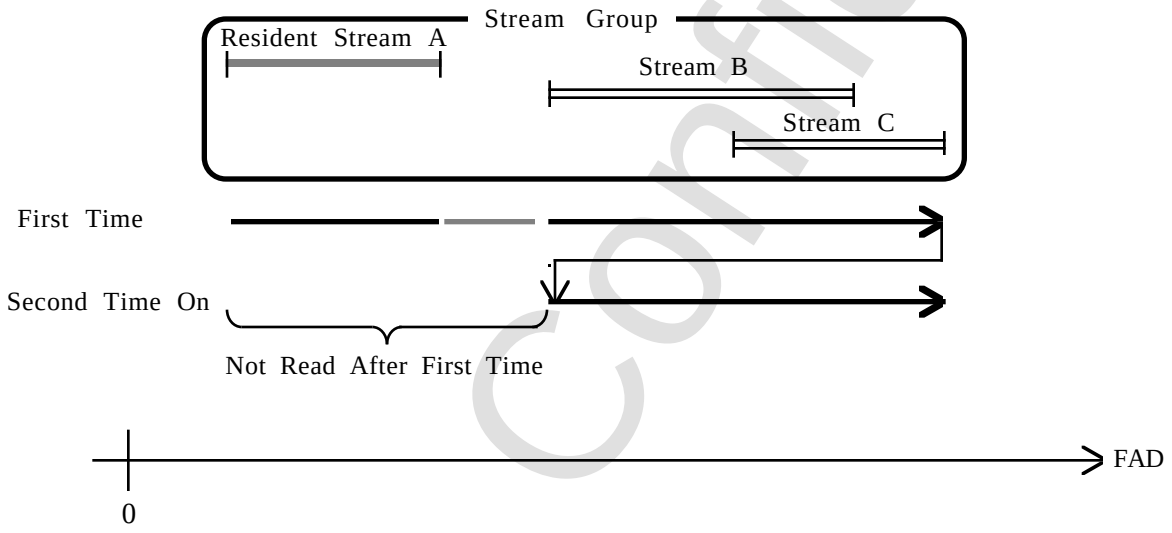
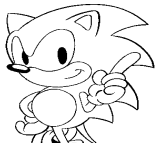


Figure 4.3 Resident Stream Read

(2) Resident Stream Transfer Start Frame Address

Once a resident stream has been read into the CD buffer, the transfer can be performed using optional timing. This timing is set using the transfer start frame address for the resident stream (STM_SetTrFad).

According to the normal stream access, the play position advances, and when it passes the set transfer start frame address, the stream data that is resident gets transferred.



Also, for resident streams, the stream data is not erased from the CD buffer even after being transferred to the host area.

(3) Using SIMM Files and SCSI Files (During Debugging)
It is possible to use SIMM files and SCSI files as resident streams only. When the specified file is a SIMM file or a SCSI file, the data can be read at each access without being resident in the CD buffer.

4.5 Precautions when Adding or Changing Settings During CD Play

In the stream system, the settings, such as the stream key, can be dynamically changed during CD play. However, after processing has been assigned to the CD block the change will be delayed until the setting contents become valid.

For this reason, the following functions must be initiated ten sectors or more before the target position.

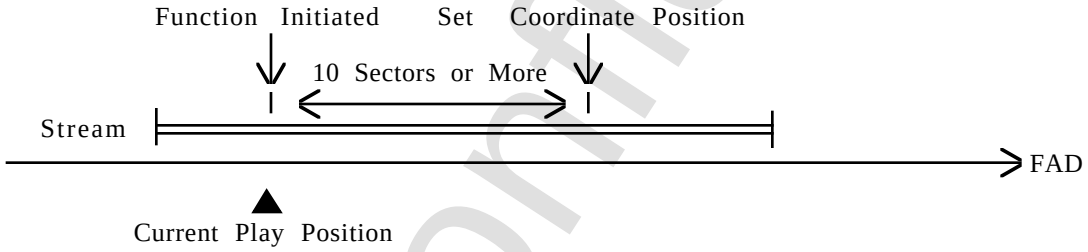


Figure 4.4 Function Initiation Timing

(1) Functions that Are Delayed Until the Settings Are Valid

Table 4.1 Functions with Corresponding Delays

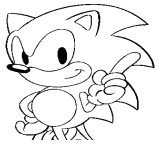
Stream opened by file ID	STM_OpenFid
Stream opened by play area	STM_OpenFrange
Resident stream open	STM_OpenResi
Stream close	STM_Close
Stream key setting	STM_SetKey
Filter and CD buffer block connection	STM_ConnectCdbuf

5.0 Stream Access Example

(1) Stream Read

Read the three streams A, B, and C into a_buf, b_buf, and c_buf respectively.

```
GfsFid          a_id, b_id, C_id;               /* file ID                    */
StmGrpHn        abc_grp;                        /* stream group handler       */
StmHn           a_stm, b_stm, c_stm;            /* stream handler             */
StmKey          key;                            /* stream key                 */
    Uint16      a_buf[ABUF_SIZE], b_buf[BUFSIZE], c_buf[C_BUFSIZE];
                                                /* transfer area              */

GFS_Init(...);                                  /* file system initialization */
STM_Init()                                      /* stream system initialization */


/* file ID acquisition                  */
a_id = GFS_NameTold(...);
                        .
                        .
                        .
abc_grp = STM_OpenGrp();                /* stream group open    */

/* stream key setting                                   /*
STM_Key_CN(&key) = STM_CN_NONE;                                 /* channel No. */
STM_KEY_CICMP(&key) = STM_KEY_CIVAL(&key) = STM_CI_NONE;  /* coding INF */


    /* stream open by file ID                           */
    STM_KEY,SMCMP(&key) =                       = STM_SM_VIDEO; /* video      */
    STM_KEY_SMAVAL(&key)                        stream
    a_stm = STM_OpenFid(abc_grp, a_id, &key);           /* stream A */
    STM_KEY_SMCMP(&key) =                       = STM_SM_AUDIO; /* open       */
    STM_KEY_SMVAL(&key)                         stream
    b_stm = STM_OpenFid(abc_grp, b_id, &key);           /* stream B */
    STM_KEY_SMCMP(&key) =                       = STM_SM_DATA; /* data stream  */
    STM_KEY_SMVAL(&key)
    c_stm = STM_OpenFid(abc_grp, c_id, &key);           /* stream C */

    /* transfer area setting   */
    STM_SetTrBuf(a_stm, a_buf, A_BUFSIZE);
    STM_SetTrBuf(b_stm, b_buf, B_BUFSIZE);
    STM_SetTrBuf(c_stm, c_buf, C_BUFSIZE);
    STM_SetExecGrp(abc_grp);                    /* actual group setting       */

    /* stream access */
    while (1) {
            if (STM_ExecServer() == STM_EXEC_COMPLETED) {
                break;                          /* stream access end          */
            }
            user();                             /* user processing            */
    }
    STM_CloseGrp(abc_grp);                      /* stream group close         */
```

(2) Transfer Function Setting

For program (1), make the following changes when using the function "decodeFunc" to transfer stream B's data while it is being decompressed.

(a) Change the underlined STM_SetTrBuf to STM_SetTrFunc.
STM_SetTrFunc(b_stm, decodeFunc, readBuf);

This change makes the decodeFunc operate every time the server function STM_ExecServer is called.

(b) The "decodeFunc" becomes as shown below:

Uint16readBuf[READBUF_SIZE];

Sint32 decodeFunc(void *obj, StmHn stm, Sint32 nsct)

```
Sint32          i;
Sint32          read_len;          /* Word number transferred by      */
                                   sub_func.
Sint32          nword:             /* Transfer word number            */
Uint16          *src;              /* Transfer address                */
Sint32          adlt;              /* Part of address changed every   */
                                   one-word transfer
                                   /*                (word            */
                                   unit)
Uint16          *buffer;           /* Transfer area                   */
nword = STM_SctToWord(stm, nsct)   /* Change from sector number to    */
                                   word number.
                                   /* Call STM_SctToWord before       */
                                   start transfer.

src = STM_StartTrans(stm, &adlt);  /* Start transfer.                 */

buffer = (Uint16 *)obj;            /* The STM_SetTrFunc's No.3        */
                                   argument crosses over to the obj.

for (i = 0; i < nword; i += read_len)  {
        buffer += sub_func(src, adlt, buffer, &read_len);
                                                /* Returns decompressed  */
                                                word count
        src += read_len * adlt;
}
return (nsct);                                  /* Return transfer       */
                                                sector count
}
```

(c) If data transfer has not ended when decodeFunc ends, (-1) must be returned.
(d) Transfer must be done in sector units.

6.0 Data Specifications

The stream system data specifications are listed below.

6.1 Data Table

The stream system data is shown in Table 6.1.

Table 6.1 Data Table

Data	Data Name	No.
Stream Access Status	StmAcStat	1.0
Transfer Gate Status	StmGate	2.0
Transfer Mode	StmTrMode	3.0
Fundamental Data		4.0
Library Handler	StmGrpHn, StmHn	5.0
Stream Key	StmKey	6.0
Stream Play Area	StmFrange	7.0
Sector Information	StmSct	8.0
Error Control	StmErr	9.0
Transfer Function	StmTrfunc	10.0
Call Function when CD Buffer Full	StmFullfunc	11.0
Error Function	StmErrfunc	12.0

6.2 Data Details

6.2.1 Stream Access Status

Title	Data	Data Name	No.
Data Specifications	Stream Access Status	StmAcStat	1.0

Table 6.2 Stream Access Status

Constant Name	Stream Access Status
STM_EXEC_COMPLETED	Access completed
STM_EXEC_PAUSE	Access pause
STM_EXEC_DOING	Accessing
STM_EXEC_WAIT	Transfer wait

When the stream cannot be accessed under the following conditions, the constant becomes STM_EXEC_WAIT.

Table 6.3 Conditions that Become a Transfer Wait

Transfer Gate	Condition
Opening	•When the transfer area is full. •When the empty area in the CD buffer disappears before transfer start FAD is reached.
Closing	•When the stream read has ended. •When the CD buffer is full.

6.2.2 Transfer Gate Status

Title	Data	Data Name	No.
Data Specifications	Transfer Gate Status	StmGate	2.0

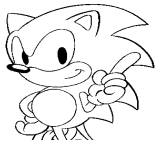


Table 6.4 Transfer Gate Status

Constant Name	Transfer Gate Status
STM_GATE_OPEN	Open status
STM_GATE_CLOSE	Closed status

- The default is STM_GATE_OPEN.

6.2.3 Transfer Mode

Title	Data	Data Name	No.
Data Specifications	Transfer Mode	StmTrMode	3.0

Table 6.5 Transfer Mode

Constant Name	Transfer Method	Load on CPU
STM_TR_SCU	SCU DMA	If the transfer destination is on B bus the CPU is working completely independently.
STM_TR_BDMA0	DMA burst channel 0	CPU is stopped.
STM_TR_BDMA1	DMA burst channel 1	CPU is stopped.
STM_TR_SDMA0	DMA cycle steal channel 0	Lower CPU processing capacity.
STM_TR_SDMA1	DMA cycle steal channel 1	Lower CPU processing capacity.
STM_TR_CPU	Software	The CPU is occupied but interruption processing is possible.

- The default is STM_TR_SCU.

6.2.4 Fundamental Data

Title	Data	Data Name	No.
Data Specifications	Fundamental Data		4.0

Table 6.6 Fundamental Data

Type Name	Explanation
Uint8	Uncoded 1-byte integer
Sint8	Coded 1-byte integer
Uint16	Uncoded 2-byte integer
Sint16	Coded 2-byte integer
Uint32	Uncoded 4-byte integer
Sint32	Coded 4-byte integer
Bool	Logic type. Takes the following values: False True

6.2.5 Library Handler

Title	Data	Data Name	No.
Data Specifications	Library Handler	StmGrpHn, StmHn	5.0

Table 6.7 Library Handler

Type Name	Explanation
StmGrpHn	Stream group handler
StmHn	Stream handler

6.2.6 Stream Key

Title Data Specifications	Data Stream Key (1/2)	Data Name StmKey	No. 6.0

(1) Data Definitions

```
typedef     struct {
            Sint16      fn;         /* File No.                          */
            Sint16      cn;         /* Channel No.                       */
            Sint16      smmsk;      /* Sum mode mask pattern             */
            Sint16      smval;      /* Sub mode comparison value         */
            Sint16      cimsk;      /* Coding information mask pattern   */
            Sint16      cival;      /* Coding information comparison value */
} StmKey;
```

(2) Access Macro

```
#define     STM_KEY_FN(stmkey)      ((stmkey)->fn)
#define     STM_KEY_CN(stmkey)      ((stmkey)->cn)
#define     STM_KEY_SMMSK(stmkey)   ((stmkey)->smmsk)
#define     STM_KEY_SMVAL(stmkey)   ((stmkey)->smval)
#define     STM_KEY_CIMSK(stmkey)   ((stmkey)->cimsk)
#define     STM-KEY-CIVAL(stmkey)   ((stmkey)->cival)
```

(3) Constant

(a) File No.

STM_FN_NONE No file No. is specified.

(b) Channel No.

STM_CN_NONE No channel No. is specified.

Title Data Specifications	Data Stream Key (2/2)	Data Name StmKey	No. 6.0

(c) Sub Mode

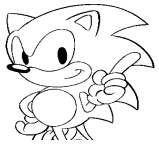
The sector that is sector sub mode & smmsk == smval is read.

Table 6.8 Constants for Sub Mode Specification

Constant Name	Type of Sector Read
STM_SM_AUDIO	Audio sector
STM_SM_VIDEO	Video sector
STM_SM_DATA	Data sector
STM_SM_NONE	No specification

(d) Coding Information

The sector that is coding information & cimsk == cival is read.



STM_CI_NONE Coding information is not specified.

6.2.7 Stream Play Area

Title	Data	Data Name	No.
Data Specification	Stream Play Area	StmFrange	7.0

(1) Data Definition

```
typedef     struct {
                Sitn32      sfad;       /* Play start FAD                    */
                Sitn32      fasnum;     /* Play sector No.                   */
} StmFrange:
```

(2) Access Macro

```
#define             STM_AREA_SFAD(plyarea)          ((plyarea)->sfad)
#define             STM_AREA_FASNUM(plyarea)        ((plyarea)->fasnum)
```

(3) Constant

Table 6.9 Constants for Play Area Specification

Constant Name	Sector Position
STM_FAD_CDTOP	Disk beginning FAD
STM_FAD_CDEND	Sector count when read to end of disk

6.2.8 Sector Information

Title	Data	Data Name	No.
Data Specifications	Sector Information	StmSct	8.0

(1) Data Definition

```
typedef     struct {
                Sint32      fad;        /* Frame address
                Sint32      fn;         /* File No.                          */
                Sint32      cn;         /* Channel No.                       */
                Uint8       sm;         /* Sub mode                          */
                Uint8       ci;         /* Coding information                */
} StmSct;
```

(2) Access Macro

```
#define             STM_SCT_FAD(sct)            ((sct)->fad)
#define             STM_SCT_FN(sct)             ((sct)->fn)
#define             STM_SCT_CN(sct)             ((sct)->cn)
#define             STM_SCT_SM(sct)             ((sct)->sm)
#define             STM_SCT_CI(sct)             ((sct)->ci)
```

6.2.9 Error Control

Title	Data	Data Name	No.
Data Specifications	Error Control	StmErr	9.0

(1) Data Definition

```
typedef     struct {
            Sint32          code;       /* Error code                          */
            Sint32          where;      /* Error occurrence location           */
            StmErrFunc      func        /* Call function when error occurs.    */
            Void            *obj;       /* Call function's first argument      */
} StmErr;
```

(2) Access Macro

```
#define     STM_ERR_CODE(err)       ((err)->code)
#define     STM_ERR_WHERE(err)      ((err)->where)
#define     STM_ERR_FUNC(err)       ((err)->func)
#define     STM_ERR_OBJ(err)        ((err)->obj)
```

6.2.10 Transfer Function

Title Data Specifications	Data Transfer Function	Data Name StmTrfunc	No. 10.0

[Format]	Sint32	(*StmTrFunc)(void *obj, StmHn stm, Sint32 nsct)
[Input]	obj	:Registration object
	stm	:Stream
	nsct	:Sector No.
[Output]	None	

6.2.11 Call Function when CD Buffer Is Full

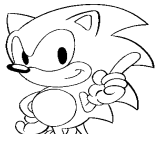
Title Data Specifications	Data Call Function when CD Buffer Is Full	Data Name StmFullfunc	No. 11.0

[Format]	void	(*StmFullFunc)(void *obj)
[Input]	obj	:Registration object
[Output]	None	

6.2.12 Error Function

Title Data Specifications	Data Error Function	Data Name StmErrfunc	No. 12.0

[Format]	void	(*StmErrFunc) (void *obj)
[Input]	obj	:Registration object
[Output]	None	



7.0 Function Table

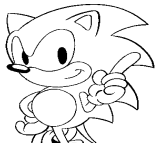
A list of the functions found in the stream system is given in Table 7.1.

Table 7.1 Function Table (1)

Function		Function Name	No.
Initialization			1.0
	Stream system initialization	STM_Init	1.1
Stream Group			2.0
	Stream group open	STM_OpenGrp	2.1
	Stream group close	STM_CloseGrp	2.2
	Stream count acquisition	STM_GetStmNum	2.3
	Stream handler acquisition	STM_GetStmHndl	2.4
	Call function registration when CD buffer is full	STM_SetCdbufFull	2.5
Stream			3.0
	Stream open using file ID	STM_OpenFid	3.1
	Stream open using play area	STM_OpenFrange	3.2
	Resident stream open	STM_OpenResi	3.3
	Stream close	STM_Close	3.4
	Stream key setting	STM_SetKey	3.5
	Stream information acquisition	STM_GetInfo	3.6
	Change from sector count to word count	STM_SctToWord	3.7
	Change from word count to sector count	STM_WordToSct	3.8
Transfer Setting			4.0
	Transfer area setting	STM_SetTrBuf	4.1
	Transfer function setting	STM_SetTrFunc	4.2
	Transfer start in transfer function	STM_StartTrans	4.3
	Transfer gate open and close	STM_SetTrGate	4.4
	Setting maximum transfer sector count	STM_SetTrPara	4.5
	Setting transfer start FAD	STM_SetTrFad	4.6
	Setting transfer mode	STM_SetTrMode	4.7
	Transfer area reset	STM_ResetTrBuf	4.8
Read Information Acquisition			5.0
	CD buffer block's sector count acquisition	STM_GetNumCdbuf	5.1
	Read sector information acquisition	STM_GetSctInfo	5.2
Transfer Information Acquisition			6.0
	Transfer area's data count acquisition	STM_GetLenTrBuf	6.1
	Transfer area full check	STM_IsTrBufFull	6.2
Stream Server Execution			7.0
	Server execution group specification	STM_SetExecGrp	7.1
	Server execution	STM_ExecServer	7.2
	Play position setting	STM_MovePickup	7.3
	Loop start stream specification	STM_SetLoop	7.4
	Execution status acquisition	STM_GetExecStat	7.5
	Stream access end check	STM_IsComplete	7.6
	Stream data transfer	STM_ExecTrans	7.7

Table 7.1 Function Table (2)

Function		Function Name	No.
CD Block Operation			8.0
	Filter and CD buffer block connection	STM_ConnectCdbuf	8.1
	CD buffer block data move	STM_MoveCdbuf	8.2
	CD buffer block data erase	STM_EraseCdbuf	8.3
Error Handling			9.0
	Registration of call function when error occurs	STM_SetErrFunc	9.1
	Error status acquisition	STM_GetErrStat	9.2



8.0 Function Details

8.1 Initialization

Title	Function	Function Name	No.
Function Specifications	Stream System Initialization	STM_Init	1.1

[Format] Bool STM_Init(void)
[Input] None
[Output] None
[Function Value] Initialization can normally be done: TRUE
Initialization cannot normally be done: FALSE

[Function]

Conduct initialization for using the stream system immediately after the program boot.

[Remarks]

(a) Must be performed immediately after GFS_Init.

8.2 Stream Group

Title	Function	Function Name	No.
Function Specifications	Stream Group Open	STM_OpenGrp	2.1

[Format] StmGrpHn STM_OpenGrp(void)
[Input] None
[Output] None
[Function Value] Stream group handler
NULL when the stream group could not be opened.

[Function]

Opens the stream group.

Title	Function	Function Name	No.
Function Specifications	Stream Group Close	STM_CloseGrp	2.2

[Format] void STM_CloseGrp(StmGrpHn grp)
[Input] grp :Stream group handler
[Output] None
[Function Value] None

[Function]

Closes the stream group.

Title	Function	Function Name	No.
Function Specifications	Stream No. Acquisition	STM_GetStmNum	2.3

[Format] Sint32 STM_GetStmNum(StmGrpHn grp)
[Input] grp :Stream group handler
[Output] None
[Function Value] Number of streams in the stream group.

[Function]

Acquires the number of streams in the specified stream group.

Title	Function	Function Name	No.
Function Specifications	Stream Handler Acquisition	STM_GetStmHndl	2.4

[Format	StmHn	STM_GetStmHndl(StmGrpHn grp, Sint32 nstm)
[Input]	grp	:Stream group handler
	nstm	:Play sequence No. (0 <= nstm < STM_GetStmNum)
[Output]	None	
[Function Value]	Specified play sequence stream handler. NULL when the pertinent stream does not exist.	

[Function]

Acquires the handler for the streams in the specified stream group.

[Remarks]

(a) When four streams are in the same stream group as shown below, the play sequence Nos. will be in the order of streams A, B, C, and D.

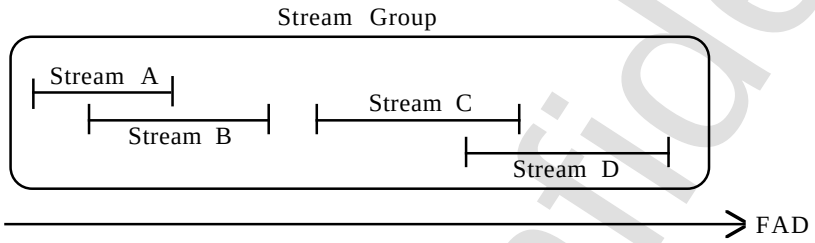


Figure 8.1 Stream Play Sequence No.

Title	Function	Function Name	No.
Function Specifications	Call Function Registration when CD Buffer Is Full	STM_SetCdbufFull	2.5

[Format]	void	STM_SetCdbufFull(StmGrpHn grp, Sint32 nsct, StmFullFunc func, void *obj)
[Input]	grp	:Stream group handler
	nsct	:Empty sector count
	func	:Call function (STM_FL_NULLFUNC when erased)
	obj	:Registered object
[Output]	None	
[Function Value]	None	

[Function]

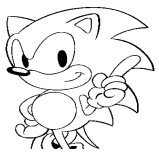
When the empty areas in the CD buffer fall below the specified value, the called functions are registered.

[Remarks]

(a) Registered functions have the following format.

```
void (*StmFullFunc)(void *obj);
```

(b) Registered objects are turned over to the first argument of the registered function.



8.3 Streams

Title	Function	Function Name	No.
Function Specifications	Stream Open Using File ID	STM_OpenFid	3.1

[Format]	StmHn	STM_OpenFid(StmGrpHn grp, Gfsfid, StmKey *key)
[Input]	grp	:Stream group handler
	fid	:File ID
	key	:Stream key
[Output]	None	
[Function Value]	Stream handler (NULL when cannot be opened)	

[Function]

Opens the stream using the file ID and registers the stream in the specified stream group.

[Remarks]

(a) The stream key file No. is not used.

Title	Function	Function Name	No.
Function Specifications	Stream Open Using Play Area	STM_OpenFrange	3.2

[Format]	StmHn	STM_OpenFrange(StmGrpHn grp, StmFrange *plyarea, StmKey *key)
[Input]	grp	:Stream group handler
	plyarea	:Play area
	key	:Stream key
[Output]	None	
[Function Value]	Stream handler (NULL when cannot be opened)	

[Function]

Opens the stream using the play area and registers the stream in the stream group.

Title	Function	Function Name	No.
Function Specification	Resident Stream Open	STM_OpenResi	3.3

[Format]	StmHn	STM_OpenResi(StmGrpHn grp, GfsFid, StmKey, *Key Sint32 Fad)
[Input]	grp	:Stream handler
	fid	:File ID
	key	:Stream key
	fad	:Transfer start FAD
[Output]	None	
[Function Value]	Stream handler (NULL when cannot be opened)	

[Function]

Opens resident streams.

Title	Function	Function Name	No.
Function Specifications	Stream Close	STM_Close	3.4

[Format]	void	STM_Close(StmHn stm)
[Input]	stm	:Stream handler
[Output]	None	
[Function Value]	None	

[Function]

Closes the specified stream.

Title Function Specifications	Function Stream Key Setting	Function Name STM_SetKey	No. 3.5

[Format]	void	STM_SetKey(StmHn stm, StmKey *stmkey)
[Input]	stm	:Stream handler
	stmkey	:Stream key
[Output]	None	
[Function Value]	None	

[Function]

Sets the stream key for the specified stream.

Title Function Specifications	Function Stream Information Acquisition	Function Name STM_GetInfo	No. 3.6

[Format]	StmGrpHn	STM_GetInfo(StmHn stm, GfsFid *fid, StmFrange *plyarea, Sint32 *bn, StmKey *stmkey)
[Input]	stm	:Stream handler
[Output]	fid	:File ID (When opened using the play area (-1))
	plyarea	:Play area
	bn	:Buffer block No.
	stmkey	:Stream key
[Function Value	Corresponding stream group	

[Function]

Acquires specified stream information.

Title Function Specifications	Function Change From Sector count to Sector Word	Function Name STM_SctToWord	No. 3.7

[Format]	Sint32	STM_SctToWord(StmHn stm, Sint32 nsct)
[Input]	stm	:Stream handler
	nsct	:Sector count
[Output]	None	
[Function Value]	The word Nos. corresponding to the specified Sector Nos.	

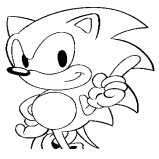
[Function]

Acquires the word Nos. from the specified sector areas starting from the beginning of the data that is written in the CD buffer block.

[Remarks]

(a) If a value larger than the read data word count is specified, the read sector count will return.

(b) Valid even if Forms 1 and 2 are mixed together.



Title	Function	Function Name	No.
Function Specifications	Change From Word No. to Sector No.	STM_WordToSct	3.8

[Format] Sint32 STM_WordToSct(StmHn stm, Sint32 nword)
[Input] stm :Stream handler
nword :Word count
[Output] None
[Function Value] Sector count corresponding to the specified word count

[Function]

Acquires the sector count from the specified word areas starting from the beginning of the data that is written in the CD buffer block.

[Remarks]

(a) If a value larger than the read data word count is specified, the read sector count will return.

(b) Valid even if Forms 1 and 2 are mixed together.

8.4 Transfer Setting

Title Function Specifications	Function Transfer Area Setting	Function Name STM_SetTrBuf	No. 4.1

[Format]	void	STM_SetTrBuf(StmHn stm, Uint16 *buffer, Sint32 bufsize)
[Input]	stm	:Stream handler
	buffer	:Transfer area
	bufsize	:Size of transfer area (word unit)
[Output]	None	
[Function Value]	None	

[Function]

Data transfer area is set in the specified stream.

[Remarks]

(a) The default transfer mode is DMA from SCU.

(b) When the transfer function is set, the transfer function has priority.

Title Function Specifications	Function Transfer Function Setting	Function Name STM_SetTrFunc	No. 4.2

[Format]	void	STM_SetTrFunc(StmHn stm, StmTrFunc func, void *obj)
[Input]	stm	:Stream handler
	func	:Transfer execution function
	obj	:Registration object
[Output]	None	
[Function Value]	None	

[Function]

Sets the transfer function in the specified stream. (STM_TR_NULLFUNC for erase)

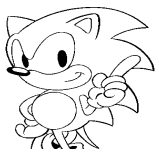
[Remarks]

(a) The format for the registered function is given below.

Sint32	(*StmTrFunc)(void *obj, StmHn stm, Sint32 nsct);
obj	:Registered object
stm	:Stream
nsct	:Sector count

(b) The transferred sector data is deleted from the CD buffer block.

(c) If data is being transferred at the time of a function end from DMA, etc., (-1) will be returned.



Title	Function	Function Name	No.
Function Specifications	Transfer Start In Transfer Function	STM_StartTrans	4.3

[Format]	Uint16	*STM_StartTrans(StmHn stm, Sint32 *adlt)
[Input]	stm	:Stream handler
[Output]	adlt	:The portion of the transfer address that changes with each word transfer.
[Function Value]	Transfer address	

[Function]

Begins transfer based on the transfer function.

[Remarks]

(a) When STM_SctToWord and STM_WordToSct are used for the transfer function, they should be called before this function is executed.

Title	Function	Function Name	No.
Function Specifications	Transfer Gate Open and Close	STM_SetTrGate	4.4

[Format]	void	STM_SetTrGate(StmHn stm, Sint32 gate)
[Input]	stm	:Stream handler
	gate	:Transfer gate status
[Output]	None	
[Function Value]	None	

[Function]

Opens and closes the specified stream's transfer gate.

Title	Function	Function Name	No.
Function Specifications	Setting Maximum Transfer Sector count	STM_SetTrPara	4.5

[Format]	void	STM_SetTrPara(StmHn stm, Sint32 tsct)
[Input]	stm	:Stream handler
	tsct	:Maximum transfer sector count (STM_TR_ALL for all read sectors.)
[Output]	None	
[Function Value]	None	

[Function]

Sets the maximum sector count that can be transferred at one time from the CD buffer block to the transfer area.

[Remarks]

(a) The data read into the CD buffer block is divided to a size less than this sector size and transferred.

(b) The default is one sector.

Title Function Specifications	Function Setting Transfer Start FAD	Function Name STM_SetTrFad	No. 4.6

[Format]	void	STM_SetTrFad(StmHn stm, Sint32 fad)
[Input]	stm	:Stream handler
	fad	:Transfer start FAD setting
[Output]	None	
[Function Value]	None	

[Function]

Sets the FAD that begins transfer of the data in the CD buffer block.

[Remarks]

(a) The default is the stream beginning FAD.

Title Function Specifications	Function Setting Transfer Mode	Function Name STM_SetTrMode	No. 4.7

[Format]	void	STM_SetTrMode(StmHn stm, Sint32 tmode)
[Input]	stm	:Stream handler
	tmode	:Transfer mode
[Output]	None	
[Function Value]	None	

[Function]

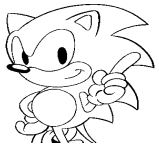
Sets the transfer method from the CD buffer block to the transfer area.

Title Function Specifications	Function Transfer Area Reset	Function Name STM_ResetTrBuf	No. 4.8

[Format]	void	STM_ResetTrBuf(StmHn stm)
[Input]	stm	:Stream handler
[Output]	None	
[Function Value]	None	

[Function]

Initializes the transfer destination pointer.



8.5 Read Information Acquisition

Title	Function	Function Name	No.
Function Specifications	CD Buffer Block's Sector count Acquisition	STM_GetNumCdbuf	5.1

[Format]	Sint32	STM_GetNumCdbuf(StmHn stm)
[Input]	stm	:Stream handler
[Output]	None	
[Function Value]	The CD buffer block sector count	

[Function]

Acquires the sector count that is read into the CD buffer block.

Title	Function	Function Name	No.
Function Specifications	Read Sector Information Acquisition	STM_GetSctInfo	5.2

[Format]	Bool	STM_GetSctInfo(StmHn stm, Sint32 sn, StmSct *sinfo)
[Input]	stm	:Stream handler
	sn	:Sector No. (The first sector is STM_CDBUF_TOP)
[Output]	sinfo	:Sector information
[Function Value]	TRUE	There is a specified sector.
	FALSE	There is no specified sector.

[Function]

Acquires the sector information that is read into the CD buffer block.

8.6 Transfer Information Acquisition

Title Function Specifications	Function Transfer Area's Data count Acquisition	Function Name STM_GetLenTrBuf	No. 6.1

[Format]	Sint32	STM_GetLenTrBuf(StmHn stm)
[Input]	stm	:Stream handler
[Output]	None	
[Function Value]	Data count (word unit)	

[Function]

Acquires the transfer area data count (word).

Title Function Specifications	Function Transfer Area Full Check	Function Name STM_IsTrBufFull	No. 6.2

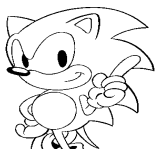
[Format]	Bool	STM_IsTrBufFull(StmHn stm)
[Input]	stm	:Stream handler
[Output]	None	
[Function Value]	TRUE	When the area size is reached.
	FALSE	When the area size is not reached.

[Function]

Checks whether the transfer area data count has reached the area size.

[Remarks]

(a) The transfer area can be initialized by STM_ResetTrBuf.



8.7 Stream Server Execution

Title	Function	Function Name	No.
Function Specifications	Server Execution Group Specification	STM_SetExecGrp	7.1

[Format]	void	STM_SetExecGrp(StmGrpHn grp)
[Input]	grp	:Stream group handler
[Output]	None	

[Function]

Specifies the stream group that is executed by the stream server.

[Remarks]

(a) When NULL is specified, the stream server is in stop status.

(b) When the stream group is reaccessed, the stop position is read.

Title	Function	Function Name	No.
Function Specifications	Server Execution	STM_ExecServer	7.2

[Format]	Sint32	STM_ExecServer(void)
[Input]	None	
[Output]	None	
[Function Value]	Stream access status	

[Function]

Executes the stream server.

Title	Function	Function Name	No.
Function Specifications	Play Position Setting	STM_MovePickup	7.3

[Format]	void	STM_MovePickup(StmHn stm, Sint32 ofs)
[Input]	stm	:Stream handler
	ofs	:Offset from the stream beginning (sector unit)
[Output]	None	
[Function Value]	None	

[Function]

Sets the play position of the stream group that contains the stream.

[Remarks]

(a) Move destination FAD = stream beginning FAD + offset.

(b) The pickup position is moved by STM_ExecServer.

Title	Function	Function Name	No.
Function Specifications	Loop Start Stream Setting	STM_SetLoop	7.4

[Format] void STM_SetLoop(StmGrpHn grp, StmHn loopstm)
[Input] grp :Stream group handler
loopstm :Loop start stream
[Output] None
[Function Value] None

[Function]

Specifies the stream group's loop start stream.

[Remarks]

(a) Does not loop when NULL is specified.

(b) When the loop start stream is closed, the beginning stream becomes the loop start stream.

Title	Function	Function Name	No.
Function Specifications	Execution Status Acquisition	STM_GetExecStat	7.5

[Format] SINT32 STM_GetExecStat(StmGrpHn grp, Sint32 *fad)
[Input] grp :Stream group handler
[Output] fad :FAD during play
[Function Value] Stream access status

[Function]

Acquires the execution status of the specified stream group.

Title	Function	Function Name	No.
Function Specification	Stream Access End Check	STM_IsComplete	7.6

[Format] Bool STM_IsComplete(StmHn stm)
[Input] stm :Stream handler
[Output] None
[Function Value] When stream access has ended TRUE
When stream access has not ended FALSE

[Function]

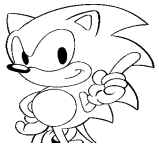
Checks whether access for the specified stream has ended.

[Remarks]

(a) The timing of stream access end is given below.

Table 8.1 Stream Access End Timing

Transfer Gate	Timing
Closed	When the read has ended
Open	When the transfer has ended



[Format]	Bool	STM_ExecTrans(StmHn stm)
[Input]	stm	:Stream handler
[Output]	None	
[Function Value]	TRUE	Transferred
	FALSE	Could not transfer

[Function]

Transfers the data of the specified stream in the CD buffer block.

[Remarks]

(a) Always FALSE when the transfer gate is closed.

(b) The set transfer mode and maximum transfer sector count are valid.

8.8 CD Block Operation

Title	Function	Function Name	No.
Function Specification	Filter and CD Buffer Block Connection	STM_ConnectCdBuf	8.1

[Format]	void	STM_ConnectCdbuf(StmHn keystm, StmHn bufstm)
[Input]	keystm	:Connection origin stream handler
	bufstm	:Connection destination stream handler (STM_CON_NULBUF when disconnected)
[Output]	None	
[Function Value]	None	

[Function]

Connects the filter in the CD block to the buffer block.

[Remarks]

(a) Reads from the filter allocated to the connection origin stream to the CD buffer block allotted to the connection destination stream.

(b) The same stream handler is specified when returning to the original setting.

(c) Shows connection for when the stream key is set to the OR condition.

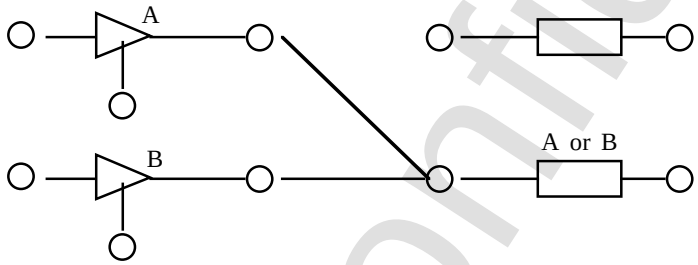


Figure 8.2 Connection When Stream Key Is Set to OR Condition

Title	Function	Function Name	No.
Function Specifications	CD Buffer Block Data Move	STM_MoveCdbuf	8.2

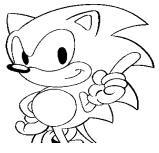
[Format]	void	STM_MoveCdbuf(StmHn src, Sint32 spos, Sint32 snum, StmHn dst)
[Input]	src	:Transfer source stream handler
	spos	:Sector position (the beginning sector is STM_CDBUF_TOP)
	snum	:Sector number (count) (STM_CDBUF_END when all the way to the end)
	dst	:Transfer destination stream handler
[Output]	None	
[Function Value]	None	

[Function]

Moves sector data from the buffer block in the CD block to filter.

[Remarks]

(a) Moves sector data from the CD buffer block allotted to the transfer origin stream to the filter allotted to the transfer destination stream.



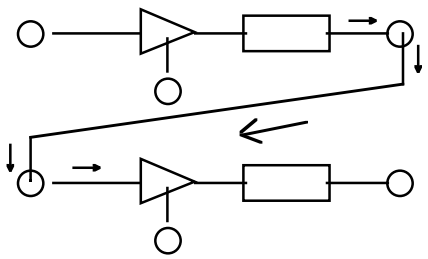


Figure 8.3 Connection During Sector Data Move

Title Function Specifications	Function CD Buffer Block Data Erase	Function Name STM_EraseCdbuf	No. 8.3

[Format]	void	STM_EraseCdbuf(StmHn stm, Sint32 spos, Sint32 snum)
[Input]	stm	:Stream handler
	spos	:Sector position (STM_CDBUF_TOP for beginning of sector)
	snum	:Sector number (count) (STM_CDBUF_END when all the way to the end)
[Output]	None	
[Function Value]	None	

[Function]

Erases sector data in the CD buffer block allotted to the stream.

8.9 Error Handling

Title	Function	Function Name	No.
Function Specifications	Registration of Call Function When Error Occurs	STM_SetErrFunc	9.1

[Format]	void	STM_SetErrFunc(StmErrFunc func, void *obj)
[Input]	func	:Call function
	obj	:Registered object
[Output]	None	
[Function Value]	None	

[Function]

Registers the function called when an error occurs.

[Remarks]

(a) The details regarding errors have not been decided yet.

(b) The registered function has the following format:

```
void (*StmErrFunc)(void *obj);
```

(c) The registered object is turned over to the registered function's first argument.

Title	Function	Function Name	No.
Function Specifications	Error Status Acquisition	STM_GetErrStat	9.2

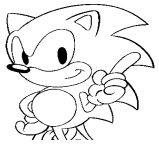
[Format]	StmErr *STM_GetErrStat(void)
[Input]	None
[Output]	None
[Function Value]	Error control structure

[Function]

Acquires error status.

[Remarks]

(a) The details regarding errors have not yet been set.



Precautionary Items Regarding Stream System Library Ver. 0.1

(a) The maximum number of stream groups that can be open at the same time is 12.

(b) The number of streams that can be opened at the same time is a maximum of 24, including the number of files currently open per file system

(c) The handling procedures for errors have not yet been set.

(d) To use the stream system, the file system and CD communication interface must be linked.