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

SEGA Saturn Dual CPU
User's Guide

Doc. # ST-202-R1-120994

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-202-R1-120994 **Date** _____

Document name SEGA Saturn Dual CPU User's Guide

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

Update History

Date	Version	Update History
August 5, 1994	1.0	
September 28, 1994	1.1	Correction of typographical errors in address values for 2.2 EVA board settings
December 9, 1994	1.2	Corrected translated error on page 5. Address 28000000H under section 6.1 was changed to 21800000H.

Contents

1.0 Dual CPU Configuration in Sega Saturn

1.1 Complete Shared Mode (Memory Sharing)

In the Sega Saturn, the slave CPU shares all external devices with the master CPU. The programs executed by the slave CPU and the reset vector obtained during reset are exactly the same as those of the master CPU. However, identification of the master and slave CPUs is performed in the boot ROM because each CPU must be initialized separately.

When the slave and master CPUs compete for external access, one of the CPUs is forced to wait for access and execution speed decreases.

1.2 Communication by FRT Input Capture Signal

The Sega Saturn is designed so that the FRT Input Capture signal of the SH2 built-in module can be used to implement communication between the master and slave CPUs. Specifically, to input the FRT Input Capture signal to the slave CPU, write 16-bit data to address 21000000H; to input the FRT Input Capture signal to the master CPU, write 16-bit data to address 21800000H.

2.0 Settings for Debugging Environment

2.1 E7000 ICE Settings

To use the E7000 ICE in the slave CPU, slave mode must be specified with the ICE MODE command.

Setting Procedure

:mode;c [Return]
E7000 MODE(MD5-0)=xx? 2E [Return]
MODE SET (C:CONFIGURATION/U:USER/M:MASTER-SLAVE)=X? C [Return]
CONFIGURATION WRITE OK?(Y/N)? Y [Return]

2.2 EVA Board Settings

To use the EVA board in the slave CPU, the following settings are required:

1. Release the slave CPU reset in the master CPU emulator.

Release Procedure

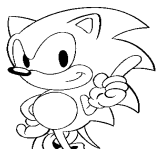
:m 2010001f:b [Return]
2010001f xx ? 02 [Return]
20100020 xx ? . [Return]

2. The EVA board in the slave CPU is started.
3. Execute em ct=d on the EVA board in the slave CPU.

When using the EVA board in the slave CPU, note that the EVA board cannot be operated until the slave CPU reset is released.

3.0 Clock Change

When the clock is changed, the SMPC switches the slave CPU to reset state. The slave CPU must therefore be restarted by the SMPC command (SSH_ON = 02H) after the clock is changed.



4.0 Programming

4.1 Program Sharing

Because the slave and master CPUs share all external devices, either CPU can execute an execution file from any location. (If the cache is used as internal RAM, programs placed in internal RAM are not shared.)

4.2 Stack Variables and Static and Global Variables

Re-entrant functions can be executed concurrently for the slave and master CPUs because a stack area is allocated separately for each CPU (boot ROM setting). However, if both CPUs execute programs that rewrite static or global variables, special attention must be paid to the execution sequence and to concurrent execution in each CPU.

4.3 Slave CPU Initialization

In slave CPU activation by the master CPU, the boot ROM rewrites the execution entry (SYS_SETSINT(164H, EntryFunc);) after read initialization. The master CPU then uses the SMPC command (SSH_ON = 02H) to release the slave reset. (EntryFunc is the slave entry function).

4.4 Initialization (Vector, Stack) by the Boot ROM

The boot ROM initializes the slave CPU as follows:

1. Vector VBR is set to address 6000400H.
2. Stack SP is set to address 6001000H.
3. Of the interrupts, only the FRT Input Capture interrupt is enabled (64H, level 1).

5.0 Communication from Master CPU to Slave CPU

Bidirectional communication is necessary for coordinating operation between the master and slave CPUs. For communication from the master CPU to the slave CPU, the Sega Saturn provides the methods described in the following sections.

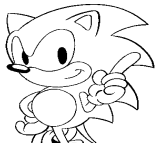
5.1 FRT Input Capture Interrupt

During boot ROM initialization, the FRT Input Capture interrupt is set for the slave CPU. To set interrupt processing from the master CPU, use the application initialization routine to register the interrupt processing routine to interrupt vector 64H of the slave CPU. (When the interrupt processing routine is registered from the master PCU, the interrupt vector is 164H.)

To issue this interrupt from the master CPU, write 16-bit data to address 21000000H. The slave CPU then starts the interrupt processing that was previously registered.

5.2 Polling of the FRT Input Capture Flag

When the SH CPU accepts the FRT Input Capture signal, it sets a flag to an internal register in the FRT unit. The SH CPU then monitors change in the flag during processing wait state (or synchronous wait state). This method is especially effective when there are many synchronous or processing waits because it has a shorter reception time than the interrupt processing method. When using this method, be sure to disable the FRT Input Capture interrupt in the application initialization routine. (Set 01H to TIER.)



6.0 Communication from Slave CPU to Master CPU

In addition to providing communication methods from the master CPU to the slave CPU, the Sega Saturn also provides all of the same communication methods for communication from the slave CPU to the master CPU.

6.1 FRT Input Capture Interrupt

During boot ROM initialization, the FRT Input Capture interrupt is set for the master CPU. To set interrupt processing from the slave CPU, set up the application initialization routine to register the interrupt processing routine to interrupt vector 64H of the master CPU.

To issue this interrupt from the slave CPU, write 16-bit data to address 21800000H. The slave CPU then starts the interrupt processing that was previously registered.

6.2 Polling of the FRT Input Capture Flag

When the SH CPU accepts the FRT Input Capture signal, it sets a flag to an internal register in the FRT unit. The SH CPU then monitors change in the flag during processing wait state (or synchronous wait state). This method is especially effective when there are many synchronous or processing waits because it has a shorter reception time than the interrupt processing method. When using this method, be sure to disable the FRT Input Capture interrupt in the application initialization routine.

7.0 Data Transfer Between Master and Slave CPUs

The cache unit of the SH CPU does not support a bus snoop function. Therefore when data is transferred between the master and slave CPUs, the CPU reading the data must either perform a cache-through read or read the data after the cache of the target area is invalidated.

7.1 Cache-through Read

In cache-through read, the CPU reads data starting from the address obtained when 20000000H is added to the target address.

7.2 Cache InValidate

To invalidate the cache, the CPU writes 0 by 16-bit access to the address obtained when 40000000H is added to the target address. This operation invalidates a 16-byte area that includes the target address.

8.0 Programming Examples Using the Slave CPU

The following sections shows program examples that use the slave CPU (when the FRT Input Capture Flag is polled.)

8.1 Slave CPU Initialization

The following examples show a program that initializes the slave CPU and an execution entry function of the slave CPU.

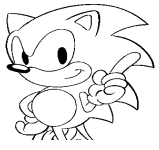
```
void  /* Initialize/start slave CPU (from master CPU) */
   InitSlaveCPU(void)
{
      volatile Uint 16 i;

   /* Set Slave SH reset status in Slave SH */
      while((*SMPC_SF & 0x01) == 0x01);
      *SMPC_SF = 1;                 /* — SMPC StatusFlag SET */
      *SMPC_COM = SMPC_SSHOFF;      /* — Slave SH OFF SET */
      while((*SMPC_SF & 0x01 == 0x01);
      for(i = 0; i < 10; i++);      /* slave RESET assert length */
      SETSINT(0x94,(void *)&SlaveCPUmain);     /* Set Entry Function */
   /* Release reset status of Slave SH */
      *SMPC_SF = 1;                 /* — SMPC Status Flag SET */
      *SMPC_COM = SMPC_SSHON;       /* — Slave SH ON SET */
      while((*SMPC_SF & 0x01) == 0x01);
}

void  /* Slave CPU entry function (FRT Input Capture Polling) */
   SlaveCPUmain(void)
{
      /* Wait until SlaveCommand is set */
      /* then call function for SlaveCommand */
      set_imask(0xf);
      *IPRA = 0x0000;   /* Disable IPRA interrupt */
      *IPRB = 0x0000;   /* Disable IPRB interrupt */
      *TIER = 0x01;     /* Disable TIER FRT Input Capture interrupt */
      while(1){
      /* Use “FRT InputCaptureFlag” Polling for wait command from Master */
         if((*FTCSR & 0x80) == 0x80){
            *FTCSR = 0x00;          /* FTCSR clear */
      /* Execute function requested from master CPU */
         (*(void(*)(void))*(void**)((Uint32)&SlaveCommand+0x20000000))();
         SlaveCommand = (void*)0;   /* RESET request code */
      }
   }
}
```

8.2 Function Execution Request from Master CPU to Slave CPU

```
extern void SlaveFunction(void);
SlaveCommand = SlaveFunction;
*(Uint16 *)0x21000000 = 0xffff;      /* FRT Input Capture for Slave */
```

Virtual CD Tool (September 27, 1994 Version)

This document provides details about the virtual CD tool 9/27/94.

Note that to use the virtual CD, virtual internal CD Board ROM version 3.1 must be replaced with version 3.2. Also note that the script syntax changed when the VCDPRE.EXE was upgraded from version 1.xx to version 2.xx.

1. Virtual CD interface board EPROM (version 3.2)
   This version corrects a fault that prevented CD-DA data on the virtual CD from being played in multi-player mode.
2. VCDEMU.EXE (version 1.70)
   This version implements multi-index and scan play. Scan play is valid only in real-time emulation mode.

   The current version does not support file interleaving during direct DOS file access.

   Note: To operate VCDEMU.EXE version 1.70, you must upgrade the ROM on the virtual CD board to version 3.2.

3. VCDPRE.EXE, VCDBUILD.EXE (version 2.13)
   Modifications from version 2.11
   - The directory record is always located at the beginning of a sector when the number of files is large and the directory file extends over multiple sectors.
   - Processing is interrupted if the same file name exists.

   [**Restriction**]
   The total number of files that are currently operating is only a little more than 500. Although the specification allows up to 1300 files to be executed as options when XMS is used, this specification has not been announced due to a bug that was found.

   Furthermore, an urgent request to support 2000 or more files has been made.

   Note: In version 2.xx, the syntax of the script file is different from the syntax used in version 1.xx. Users of version 1.xx must change the script file. For details, refer to the syntax in item 7, EXSAMPLE.SCR.

4. VCDUTL.EXE (version 1.00)
5. VCDMKTOC.EXE (version 1.20)
6. SWAP.EXE (version 1.00)
   SWAP.EXE is an endian conversion program for CD-DA data.
7. EXSAMPLE.SCR
   EXSAMPLE.SCR is an example of new script file for VCDPRE version 2.xx. Refer to this example during script modification.