

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 Confidential

SCU

User's Manual

Third version

Doc. # ST-97-R5-072694

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-97-R5-072694 Date _____

Document name SCU User's Manual

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

Version History

Version 1: April 7, 1994

- New draft

Version 2: May 31, 1994

- Revisions according to April 28, 1994 meeting

Version 3: July 15, 1994

- Revisions requested on June 30 and July 11, 1994

SEGA Confidential

Introduction

This manual explains functions of the system controller and how they are used. The system controller transfers data rapidly and smoothly by means of the bus controls.

Explanation of Terms

The following terms are used in this manual.

- SCU** System Control Unit. The SCU contains the CPU I/F, A-Bus IF, B-BUS I/F, and smoothly effects data transfers between several processors connected through their respective I/F and bus. It also internally houses the DMA controller, interrupt controller, and DSP, and makes possible rapid DMA control, interrupt control, and processing of operations.
- Main CPU** Uses a RISC type CPU SH2 that controls the overall system. SH2 contains 32-bit internal and external buses.
- VDP1** Video Display Processor 1. Functions include character and line painting, color indication, Gouraud Shading color operations, screen output coordinate indication, and frame buffer display control.
- VDP2** Video Display Processor 2. Functions include scrolling the screen up/down/left/right, rotating the screen, determining priority order of multiple screens, and a priority function that controls the image process of color operations and color offset.
- SCSP** Acronym for Saturn Custom Sound Processor. This is a sound source LSI for multi-functional games that combines a PCM sound source and sound used for the DSP.

- SMPC** System Manager and Peripheral Control. Has the functions of managing system resets, control of interfacing with output devices (control pads, mouse, etc.), time display by a real time clock, and battery backup.
- Data** A bit is the smallest unit for expressing 1 or 0. 8 bits is a byte. 16 bits (or 2 bytes) is a word. 32 bits (or 4 bytes) is a 9 long word.
- A_Bus** Bus that connects external devices such as a ROM cassette or CD.
- B_Bus** Bus that connects VDP1, VDP2, and SCSP.

Manual Notations

This manual contains the following notations.

Binary	Represented by “B” at the end as in 100 _B . However, “B” may be omitted for 1 bit.
Hexadecimal	Represented by _H at the end as in 00 _H and FF _H .
Unit	1 KByte is 1,024 bytes. 1 Mbit is 1,048,576 bits.
MSB, LSB	The configuration of byte and word shows at the left the high order bit (MSB, most significant bit), and at the right the low order bit (LSB, least significant bit).
Undefined Bit	A bit not defined by an instruction word is represented by “—”
(R)	Represents read only data.
(W)	Represents write only data.
(R/W)	Represents data that can be read and written.
++	Shows increments. For example, when the CT0 register is incremented, it is shown as CT0++.
x=2-0	This indicates that 3 types exist, 2,1, and 0. For example, D _x R26-0[x=2-0] in the read address in section 3.2 “DMA Control Register” means that D2R26-0, D1R26-0, and D0R26-0 exist. Similarly, D2R26-0 indicates that D2R26 ~ D2R0 exist.

CONTENTS

INTRODUCTION

Explanation of Terms	(i)
Manual Notations	(iii)

List of Figures	(vii)
------------------------------	-------

List of Tables	(x)
-----------------------------	-----

CHAPTER 1 OVERVIEW 1

1.1 SCU Overview 2

System Diagram	2
----------------------	---

Block Diagram	3
---------------------	---

1.2 SCU Mapping 4

Operation of Cache Hit	5
------------------------------	---

1.3 SCU Register Map 7

Level 2-0DMA Set Register	8
---------------------------------	---

DMA Forced-Stop Register	8
--------------------------------	---

DMA Status Register	9
---------------------------	---

DSP Program Control Port	9
--------------------------------	---

DSP Program RAM Data Port	10
---------------------------------	----

DSP Data RAM Address Port	10
---------------------------------	----

DSP Data RAM Data Port	10
------------------------------	----

Timer 0 Compare Register	11
--------------------------------	----

Timer 1 Set Data Register	11
---------------------------------	----

Timer 1 Mode Register	11
-----------------------------	----

Interrupt Mask Register	12
-------------------------------	----

Interrupt Status Register	12
---------------------------------	----

A-Bus Interrupt Acknowledge Register	12
--	----

A-Bus Set Register	13
--------------------------	----

A-Bus Refresh Register	13
------------------------------	----

SCU SDRAM Select Register	14
---------------------------------	----

SCU Version Register	14
----------------------------	----

CHAPTER 2 OPERATION	15
2.1 DMA Transfer	16
Basic Operation of DMA	16
DMA Mode.....	18
Example of a Specific Use.....	21
2.2 Interrupt Control	27
Blanking Interrupt	29
Timer Interrupt	30
DSP-End Interrupt	33
Sound-Request Interrupt	33
SMPC Interrupt.....	33
PAD Interrupt	33
DMA End Interrupt	33
DMA-Illegal Interrupt.....	33
Sprite Draw End Interrupt	33
2.3 DSP	34
DSP Control from the Main CPU	34
CHAPTER 3 REGISTERS	39
3.1 Register List	40
3.2 DMA Control Registers	41
Level 2-0 DMA Set Register	41
DMA Mode, Address Update, Start Factor Select Register	46
DMA Force-Stop Register	47
DMA Status Register	47
3.3 DSP Control Ports	51
DSP Program Control Port	51
DSP Program RAM Data Port	53
DSP Data RAM Address Port	53
DSP Data RAM Data Port	54
3.4 Timer Registers	55
Timer 0 Compare Register	55
Timer 1 Set Data Register	55
Timer 1 Mode Register	56

3.5 Interrupt Control Registers	57
Interrupt Mask Register	57
Interrupt Status Register	58
3.6 A-Bus Control Registers	61
A-Bus Interrupt Acknowledge Register	61
A-Bus Set Register	62
A-Bus Refresh Register	72
3.7 SCU Control Registers	73
SCU SDRAM Select Register	73
SCU Version Register	73
CHAPTER 4 DSP CONTROL	75
4.1 DSP Internal BLOCK MAP	76
4.2 List of Commands	80
4.3 Operand Execution Methods	85
Jump Command Execution	85
Loop Command Execution	86
DMA Command Execution	87
End Command Execution	88
4.4 Special Process Execution	89
Loading a Program by the DMA Command	89
Repeating One Command	89
Executing a Subroutine Program	90
4.5 More About Commands	91
Operation Commands	91
Load Immediate Command	120
DMA Command	132
Jump Commands	141
Loop Bottom Commands	153
END Command	156

List of Figures

(Chapter 1 Overview)

Figure 1.1 Diagram of System	2
Figure 1.2 Block Diagram	3
Figure 1.3 SCU Mapping (Cache_address).....	4
Figure 1.4 Explanation of Cache Hit Operation	5
Figure 1.5 SCU Mapping (Cache_through_address).....	6
Figure 1.6 SCU Register Map	7
Figure 1.7 Level 2-0 DMA Set Register Map	8
Figure 1.8 DMA Force-Stop Register Map	8
Figure 1.9 DMA Status Register Map	9
Figure 1.10 DSP Program Control Port Map	9
Figure 1.11 DSP Program RAM Data Port Map.....	10
Figure 1.12 DSP Data RAM Address Port Map	10
Figure 1.13 DSP Data RAM Data Port Map	10
Figure 1.14 Timer 0 Compare Register Map	11
Figure 1.15 Timer 1 Set Data Register Map	11
Figure 1.16 Timer 1 Mode Register Map	11
Figure 1.17 Interrupt Mask Register Map	12
Figure 1.18 Interrupt Status Register Map.....	12
Figure 1.19 A-Bus Interrupt Acknowledge Map	12
Figure 1.20 A-Bus Set Register Map	13
Figure 1.21 A-Bus Refresh Register Map	13
Figure 1.22 SCU SDRAM Select Register Map.....	14
Figure 1.23 SCU Version Register Map	14

(Chapter 2 Operation)

Figure 2.1 DMA Transfer Basic Operation	16
Figure 2.2 DMA Transferable Area when Activated from the Main CPU	17
Figure 2.3 DMA Transferable Area when Activated from the DSP	17
Figure 2.4 Direct Mode DMA Transfer Operation	18
Figure 2.5 Indirect Mode DMA Transfer Flow	19
Figure 2.6 Indirect Mode DMA Transfer Operation Details	20
Figure 2.7 Differences in DMA Operations according to the Address Update Bit	22

Figure 2.8 Example of Data Write	23
Figure 2.9 Work RAM Area Contents	24
Figure 2.10 DMA Transfer by Setting Address Add Value	26
Figure 2.11 Blanking Interrupt.....	29
Figure 2.12 Timer 0 Interrupt Process (compare register = when 19 is set)	30
Figure 2.13 Timer 1 Interrupt Process (In sync with Timer 0)	31
Figure 2.14 Timer 1 Interrupt Process (not in sync with Timer 0)	32
Figure 2.15 DSP Program Load Step 1	34
Figure 2.16 DSP Program Load Step 2	35
Figure 2.17 DSP Program Load Step 3	35
Figure 2.18 DSP Data Access Step 1	36
Figure 2.19 DSP Data Access Step 2.....	37
Figure 2.20 DSP Data Access Step 3.....	37
Figure 2.21 DSP Program Execution Start Control from CPU	38
Figure 2.22 DSP Program Forced Stop Control from CPU	38

(Chapter 3 Registers)

Figure 3.1 Level 2-0 Read Address (Register: D0R, D1R, D2R)	41
Figure 3.2 Level 2-0 Write Address (Register: D0W, D1W, D2W)	41
Figure 3.3 Level 0 Transfer Byte Number (Register: D0C)	42
Figure 3.4 Level 2-1 Transfer Byte Number (Register: D1C, D2C).....	42
Figure 3.5 Level 2-0 Address Add Value (Register: D0AD, D1AD, D2AD)	42
Figure 3.6 Communication Units between the SCU and Processor	44
Figure 3.7 Specific Example of Transfer between the SCU and Processor	44
Figure 3.8 Write Address Add Value Indication	45
Figure 3.9 Level 2-0 DMA Authorization Bit (Register: D0EN, D1EN, D2EN)	45
Figure 3.10 Level 2-0 DMA Mode, Address Update, Start Up Factor Select Register (Register: D0MP, D1MP, D2MP)	46
Figure 3.11 DMA Force-Stop Register (Register: DSTP)	47
Figure 3.12 High and Low Level DMA Operation	48
Figure 3.13 DMA Status Register (Register: DSTA)	48

Figure 3.14 DSP Program Control Port (Register: PPAF)	51
Figure 3.15 DSP Program RAM Data Port (Register: PPD)	53
Figure 3.16 DSP Data RAM Address Port (Register: PDA)	53
Figure 3.17 DSP Data RAM Data Port (Register: PDD)	54
Figure 3.18 Time 0 Compare Register (Register: T0C)	55
Figure 3.19 Timer 1 Set Data Register (Register: T1S)	55
Figure 3.20 Timer 1 Mode Register (Register: T1MD)	56
Figure 3.21 Interrupt Mask Register (Register: IMS)	57
Figure 3.22 Interrupt Status Register (Register: IST)	58
Figure 3.23 A-Bus Interrupt Acknowledge Register (Register: AIAK).....	61
Figure 3.24 A-Bus Set [CS0, 1 Space] (Register: ASR0).....	62
Figure 3.25 A-Bus Set [CS2, Dummy Space] (Register: ASR1)	62
Figure 3.26 Result of Previous Read Process	63
Figure 3.27 Timing when Setting the Pre-Charge Insert Bit after Write	63
Figure 3.28 Timing when Setting the Pre-Charge Insert Bit after Read	64
Figure 3.29 Differences in Timing by Setting External Wait Effective Bit	64
Figure 3.30 A-Bus Refresh Register (Register: AREF)	72
Figure 3.31 SCU SDRAM Select Bit (Register: RSEL)	73
Figure 3.32 SCU Version Register (Register: VER)	73

Chapter 4 DSP Control)

Figure 4.1 DSP Internal Block Map	77
Figure 4.2 Jump Command Execution	85
Figure 4.3 Loop Program Execution	86
Figure 4.4 Subroutine Program Execution	91
Figure 4.5 Operation Command Format	92
Figure 4.6 Load Immediate Command Format 1 (Unconditional Transfer)	120
Figure 4.7 Load Immediate Command Format 2 (Conditional Transfer)	120
Figure 4.8 DMA Command Format 1	132
Figure 4.9 DMA Command Format 2	132
Figure 4.10 Jump Command Format	141
Figure 4.11 Loop Bottom Command Format	153
Figure 4.12 End Command Format	156

List of Tables

(Chapter 2 Operation)

Table 2.1 Interrupt Factors	27
Table 2.2 Interrupt Factor General Names	28

(Chapter 3 Registers)

Table 3.1 Register List	40
Table 3.2 Read Address Add Value	43
Table 3.3 Write Address Add Value	43
Table 3.4 Starting Factors	46
Table 3.5 RAM Page Select	53
Table 3.6 Timer 1 Occurrence Selection Contents	56
Table 3.7 Timer Operation Contents	56
Table 3.8 Interrupt Status Bit Contents	59
Table 3.9 A-Bus Interrupt Acknowledge Contents	61
Table 3.10 CS0 Space Burst Cycle Set Values	65
Table 3.11 CS0 Space Normal Cycle Set Values	65
Table 3.12 CS0 Space Burst Length Set Values	65
Table 3.13 CS0 Space Bus Size Set Values	66
Table 3.14 CS1 Space Burst Cycle Set Values	67
Table 3.15 CS1 Space Normal Cycle Set Values	67
Table 3.16 CS1 Space Burst Length Set Values	68
Table 3.17 CS1 Space Bus Size Set Values	68
Table 3.18 CS2 Space Burst Cycle Set Values	69
Table 3.19 CS2 Space Bus Size Set Values	70
Table 3.20 Dummy Space Burst Cycle Set Values	71
Table 3.21 Dummy Space Normal Cycle Set Values	71
Table 3.22 Dummy Space Burst Length Set Values	71
Table 3.23 Dummy Space Bus Size Set Values	72
Table 3.24 A-Bus Refresh Wait Number	72

(Chapter 4 DSP Control)

Table 4.1 List of Commands (1)	80
Table 4.2 List of Commands (2)	81
Table 4.3 List of Commands (3)	82
Table 4.4 List of Commands (4)	83
Table 4.5 Descriptions of Constants	84
Table 4.6 Features of Data Transfer from D0 Bus to DSP	87
Table 4.7 Features of Data Transfer from DSP to D0 Bus	88

SEGA Confidential

CHAPTER 1 OVERVIEW

Chapter 1 Contents

1.1	SCU Overview	2
	System Diagram	2
	Block Diagram	3
1.2	SCU Mapping	4
	Operation of Cache Hit	5
1.3	SCU Register Map	7
	Level 2-0DMA Set Register	8
	DMA Forced-Stop Register	8
	DMA Status Register	9
	DSP Program Control Port	9
	DSP Program RAM Data Port	10
	DSP Data RAM Address Port	10
	DSP Data RAM Data Port	10
	Timer 0 Compare Register	11
	Timer 1 Set Data Register	11
	Timer 1 Mode Register	11
	Interrupt Mask Register	12
	Interrupt Status Register	12
	A-Bus Interrupt Acknowledge Register	12
	A-Bus Set Register	13
	A-Bus Refresh Register	13
	SCU SDRAM Select Register	14
	SCU Version Register	14

1.1 SCU Overview

The SCU (System Control Unit) contains a CPU I/F, A-Bus I/F, and B-Bus I/F. It smoothly interfaces multiple processors connected through their respective I/Fs and buses. Also contained inside are the DMA controller, interrupt controller, and DSP.

The DMA controller controls the internal level 2-0 as well as DSP total 4 channel DMA transfer, and allows the free transfer of data between the CPU, A-Bus, and B-Bus. Using the CPU-Bus, the CPU can access the work area while executing the DMA of the A-Bus and B-Bus. The DSP region must be used in data transfer request from the DSP. For instance, DMA transfer with the A-Bus and B-Bus not using the DSP region cannot request that data be transferred from the DSP.

The interrupt controller includes interrupts from the A-Bus, B-Bus, and System Manager, and controls all interrupts within the SCU. It also supports interrupt by timers and can produce interrupts that are in sync with the screen.

DSP can handle processes that cannot be handled by the main CPU when its load has been exceeded. DSP operates at half the frequency of the main CPU. As a result, one step takes about 70 nsec.

System Diagram

A diagram of the system is shown in Figure 1.1. The Work RAM-H, Work RAM-L, Backup RAM, IPL ROM, and SMPC are connected to the CPU-Bus. The CPU-Bus controls the system reset signal and control pad. The medium that supplies the CD or cartridge software is an external system connected to the A-Bus. VDP1, VDP2, and SCSP are connected to the B-Bus and control picture and sound.

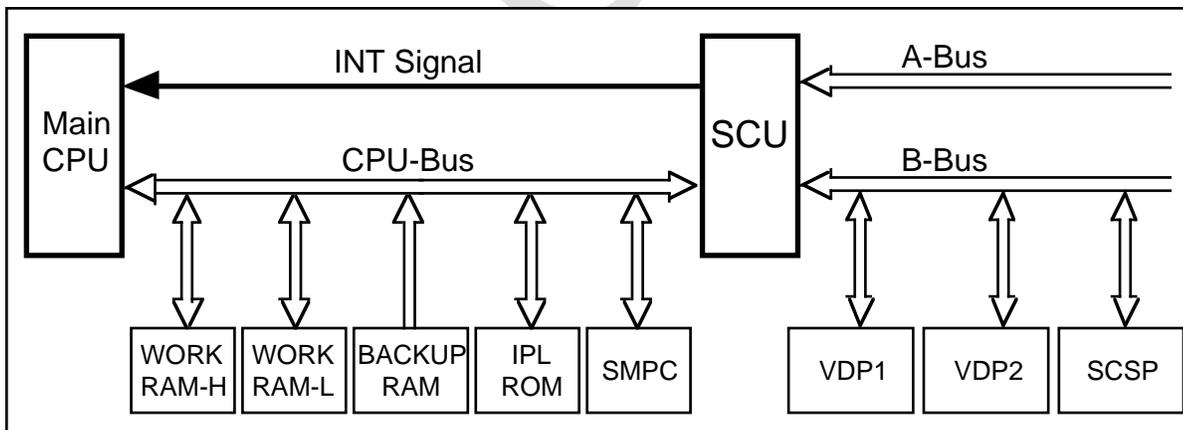


Figure 1.1 Diagram of System



Block Diagram

A block diagram of the SCU is shown in Figure 1.2. As previously mentioned, the CPU interface, A-Bus, and B-Bus interfaces, and the DMA controller, interrupt controller, and DSP are contained in the SCU. All interfaces and controllers are connected by buses, making transfer of data possible.

The CPU I/F and A-Bus I/F connections are through two buses. The upper bus is connected through the register. The lower bus is a connection used in transferring data. Therefore, DMA transfer is done using the lower bus.

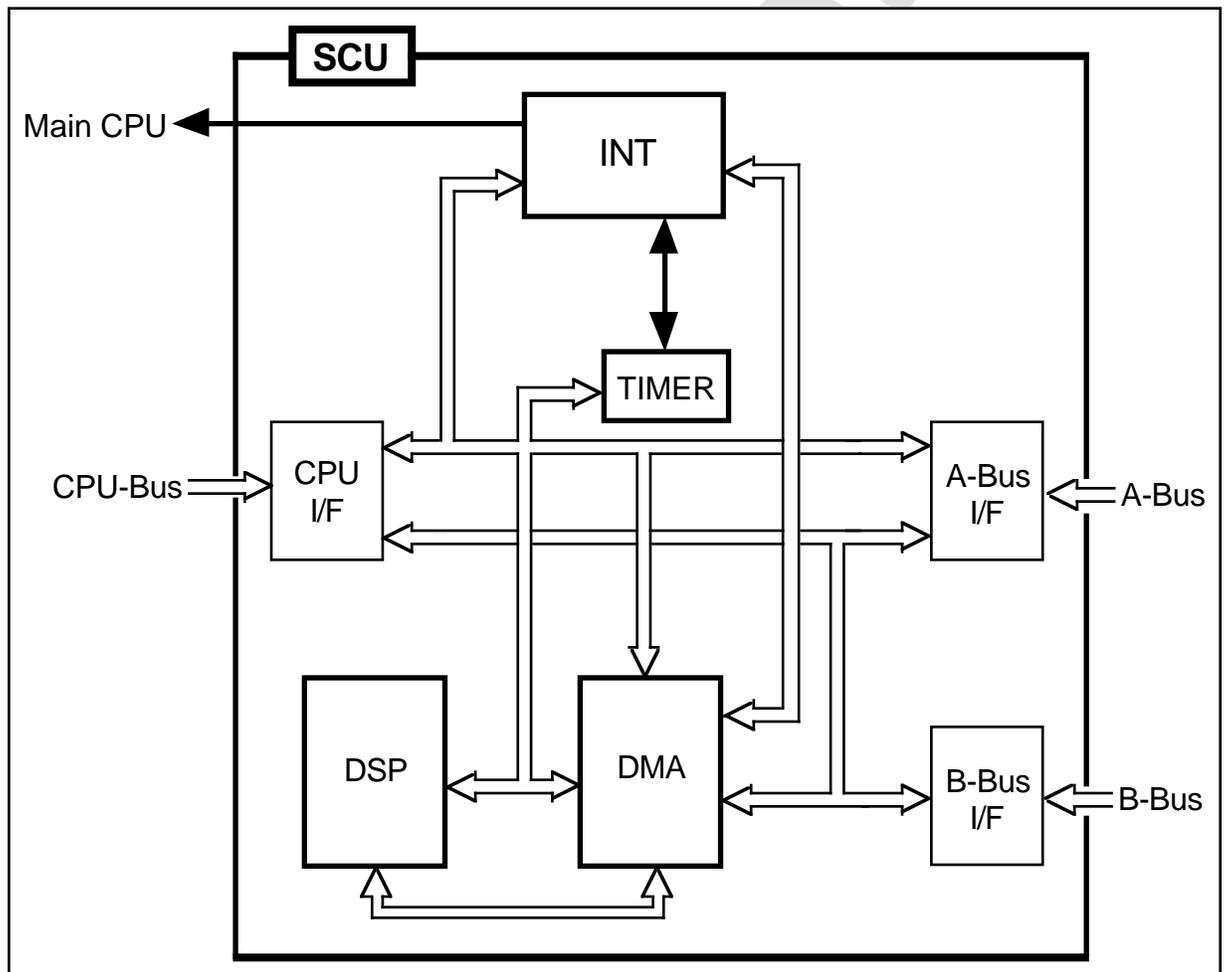


Figure 1.2 Block Diagram

1.2 SCU Mapping

Figure 1.3 shows the mapping operation.

00000000 H	ROM Access Region	512 Kbyte
00080000 H		
00100000 H	SMPC Region	128 byte
00100080 H		
00180000 H	Backup-RAM Region	64 Kbyte
00190000 H		
00200000 H	Work-RAM-L Region	1 Mbyte
00300000 H		
01000000 H	MINIT Region	4 byte
01000004 H		
01800000 H	SINIT Region	4 byte
01800004 H		
02000000 H		
	A-Bus CS0 Region	32 Mbyte
04000000 H	A-Bus CS1 Region	16 Mbyte
05000000 H	A-Bus Dummy Region	8 Mbyte
05800000 H	A-Bus CS2 Region	1 Mbyte
05900000 H		1 Mbyte
05A00000 H	Sound Region	about 1 Mbyte
05B00EE4 H		
05C00000 H	VDP 1 Region	192 Kbyte
05CC0000 H		
05D00000 H	VDP 1 Region	24 byte
05D00018 H		
05E00000 H	VDP 2 Region	512 Kbyte
05E80000 H		
05F00000 H	VDP 2 Region	4 Kbyte
05F01000 H		
05F80000 H	VDP 2 Region	288 byte
05F80120 H		
05FE0000 H	SCU Register Region	208 byte
05FE00D0 H		
06000000 H	Work RAM-H Region	1 Mbyte
06100000 H		
07FFFFFF H		

Indicates areas that can't be accessed

Figure 1.3 SCU Mapping (Cache_address)



Operation of Cache Hit

If a hit is made to the cache during access to an area that is rewritable by non-CPU devices such as the work RAM of an I/O port, an external device, or a SCU register, a value different from the actual value could be returned. When this happens, the cache-through area must be accessed.

Figure 1.4 explains cache hit operations, and Figure 1.5 shows cache-through operations.

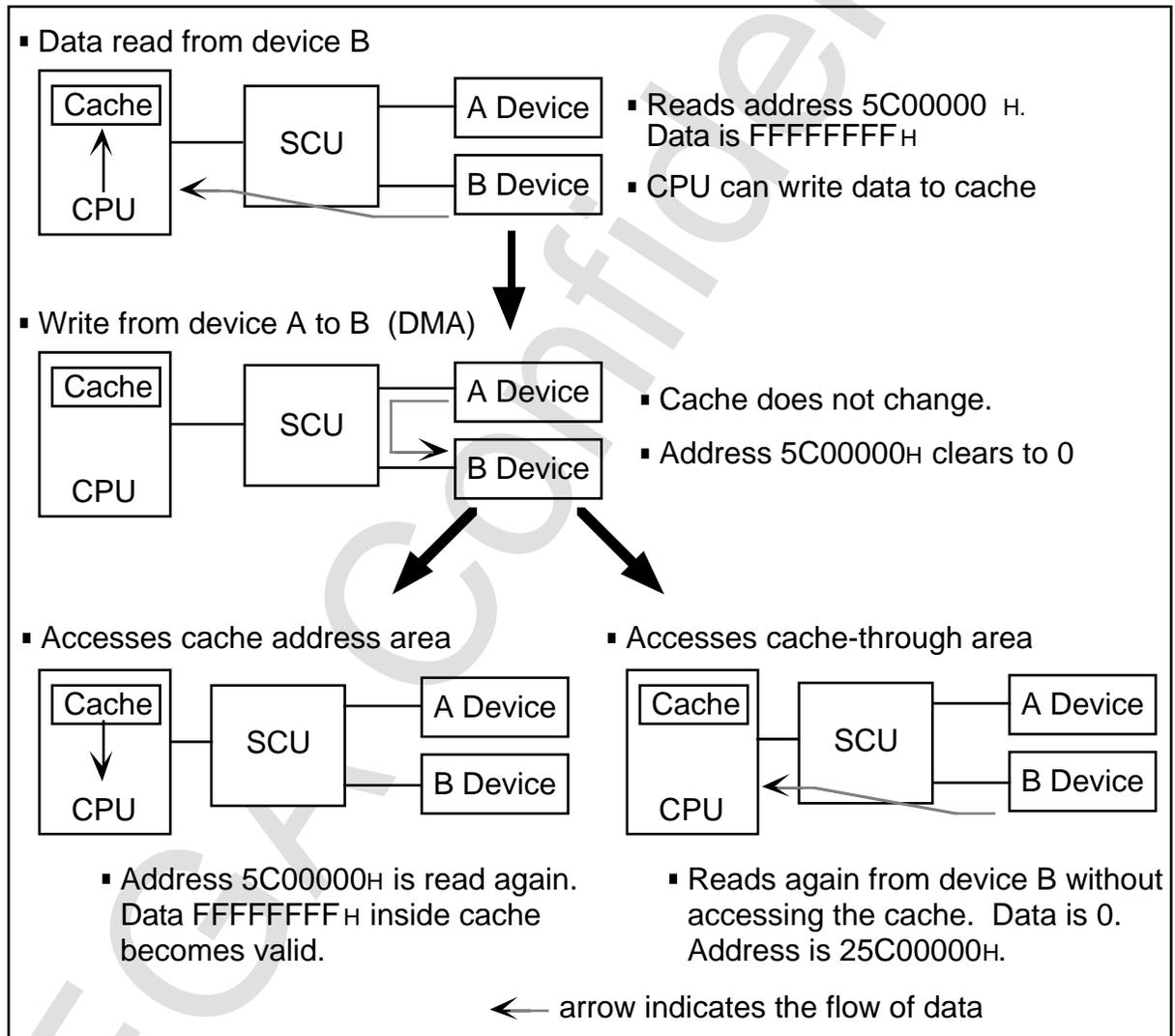


Figure 1.4 Explanation of Cache Hit Operation



Figure 1.5 SCU Mapping (Cache_through_address)



1.3 SCU Register Map

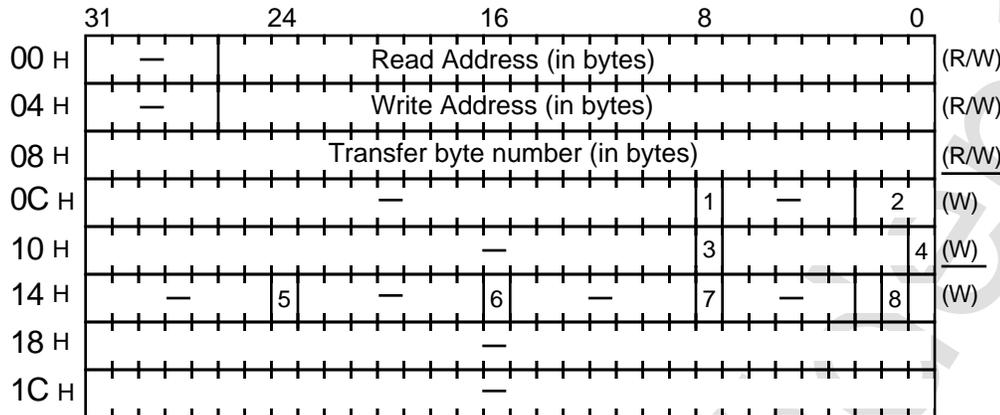
Figure 1.6 shows a map of the SCU register. The SCU register is assigned to the highest address in the SCU mapping region and, as shown in Figure 1.3, maintains a 208 byte area. Next, a map of each register region is shown.

25FE0000 H	Level 0 DMA Set Register	32 byte
25FE0020 H	Level 1 DMA Set Register	32 byte
25FE0040 H	Level 2 DMA Set Register	32 byte
25FE0060 H	DMA Forced Stop	16 byte
25FE0070 H	DMA Status Register	16 byte
25FE0080 H	DSP Program Control Port	4 byte
25FE0084 H	DSP Program RAM DataPort	4 byte
25FE0088 H	DSP Data RAM Address Port	4 byte
25FE008C H	DSP Data RAM DataPort	4 byte
25FE0090 H	Timer 0 Compare Register	4 byte
25FE0094 H	Timer 1 Set Data Register	4 byte
25FE0098 H	Timer 1 Mode Register	4 byte
25FE009C H	Free	4 byte
25FE00A0 H	Interrupt Mask Register	4 byte
25FE00A4 H	Interrupt Status Register	4 byte
25FE00A8 H	A-Bus Interrupt Acknowledge	4 byte
25FE00AC H	Free	4 byte
25FE00B0 H	A-Bus Set Register	8 byte
25FE00B8 H	A-Bus Refresh Register	4 byte
25FE00BC H	Free	8 byte
25FE00C4 H	SCU SDRAM Select Register	4 byte
25FE00C8 H	SCU Version Register	4 byte
25FE00CC H	Free	4 byte
25FE00CF H		

Figure 1.6 SCU Register Map

Level 2-0 DMA Set Register

Figure 1.7 is a map of the Level 2-0 DMA set register. Parameters required for DMA transfer are stored in this register. There are three DMA levels (from level 0 to level 2), as there are in the SCU register map (Figure 1.6). As a result, the addresses in Figure 1.7 are shown as relative addresses.



Inside graphic:

1. Read address add value
2. Write address add value
3. DMA enable bit (=0:Disable / =1:Enable)
4. DMA starting bit
5. DMA mode bit (=0:Direct Mode / =1:Indirect Mode)
6. Read address update bit (=0:Save / =1:Revise)
7. Write address update bit (=0:Save / =1:Update)
8. DMA start factor select bit

Figure 1.7 Level 2-0 DMA Set Register Map

DMA Force-Stop Register

Figure 1.8 is a map of the DMA force-stop register. This register has a bit that forces the DMA operation to stop. However, if the DMA is forced to stop, it can no longer be used. This register should not be used except for debugging.



Inside graphic:

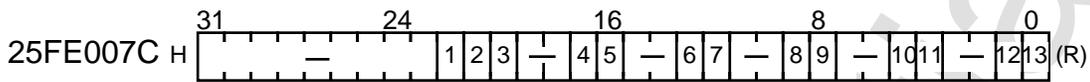
1. DMA force-stop bit (=1:DMA force-stop)

Figure 1.8 DMA Force-Stop Register Map



DMA Status Register

Figure 1.9 is a map of the DMA status register. This register shows level 2-0 condition status.



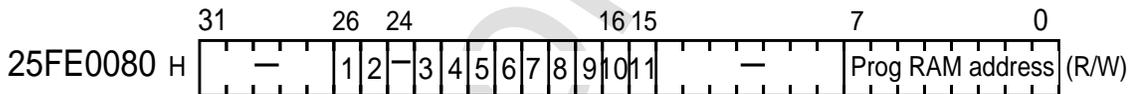
Inside graphic:

- | | |
|--|--|
| 1. DMA DSP-Bus access flag (=0: no access / =1:access) | 8. Level 1 DMA standby (=0:stop/=1:standby) |
| 2. DMA B-Bus access flag (=0: no access / =1:access) | 9. Level 1 DMA in operation (=0:stop/=1:operate) |
| 3. DMA A-Bus access flag (=0: no access / =1:access) | 10. Level 0 DMA stand by (=0:stop/=1:standby) |
| 4. Level 1 DMA interrupt(=0:stop/=1:interrupt) | 11. Level 0 DMA in operation (=0:stop/=1:operate) |
| 5. Level 0 DMA interrupt(=0:stop/=1:interrupt) | 12. DSP side DMA in stand by (=0:stop/=1:standby) |
| 6. Level 2 DMA standby (=0:stop/=1:standby) | 13. DSP side DMA in operation (=0:stop/=1:operate) |
| 7. Level 2 DMA in operation (=0:stop/=1:operate) | |

Figure 1.9 DMA Status Register Map

DSP Program Control Port

Figure 1.10 is a map of the DSP program control port. This is the DSP control register. It stores both the DSP operation start address and end address.



Inside graphic:

- | | |
|--|---|
| 1. EX = cancels pause briefly (=0: no execute / =1:execute) | 7. Overflow flag |
| 2. EX = executes pause briefly (=0: no execute / =1:execute) | 8. Program end interrupt flag |
| 3. D0 bus use DMA transfer execution flag | |
| 4. Sine flag | 9. Program step execute control bit (=0:no execute/=1:execute) |
| 5. Zero flag | 10. Program execute control (=0:stop/=1:execute) |
| 6. Carry flag | 11. Program counter load authorization (=0:no execute/=1:execute) |

Figure 1.10 DSP Program Control Port Map

DSP Program RAM Data Port

Figure 1.11 is a map of the DSP program RAM data port. This port is used as a go-between when transferring program data from the CPU to the DSP.

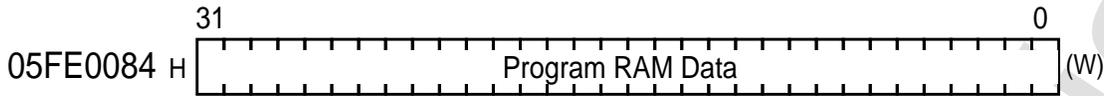


Figure 1.11 DSP Program RAM Data Port Map

DSP Data RAM Address Port

Figure 1.12 is a map of the DSP data RAM address port. This port indicates the data RAM address while accessing the data RAM inside DSP from the CPU.



Figure 1.12 DSP Data RAM Address Port Map

DSP Data RAM Data Port

Figure 1.13 is a map of the DSP data RAM data port. The content of the address shown by the DSP data RAM address port is stored. Data written from the CPU is stored in the DSP data RAM and data read from the CPU can fetch RAM data inside the DSP.



Figure 1.13 DSP Data RAM Data Port Map



Timer 0 Compare Register

Figure 1.14 is the map of the timer 0 compare register. Timer 0 gets in sync with V-Blank-IN interrupt (See 2.2 *Interrupt Control*) and causes interrupt to occur. The operation is explained in section 2.2 and the register contents are explained in chapter 3.

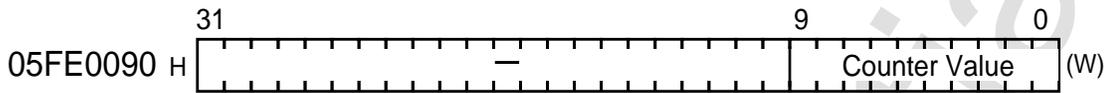


Figure 1.14 Timer 0 Compare Register Map

Timer 1 Set Data Register

Figure 1.15 is the map timer 1 set data register. Timer 1 is *data-set* by H-Blank-IN interrupt (See 2.2 *Interrupt Control*) and decremented by 7 MHz cycles. Interrupt occurs when data is 0. The operation is explained in section 2.2 and the register contents are explained in chapter 3.



Figure 1.15 Timer 1 Set Data Register Map

Timer 1 Mode Register

Figure 1.16 is a map of the timer 1 mode register. This register indicates the timing by which Time 1 is generated. The operation is explained in section 2.2 and the register contents are explained in chapter 3.



Inside graphic:

1. Timer 1 mode bit

=0:occurs at each line

=1:occurs only for lines designated by timer 0

2. Time operation enable bit

=0: Timer operation OFF

=1 : Timer operation ON

Figure 1.16 Timer 1 Mode Register Map

Interrupt Mask Register

Figure 1.17 shows the map of the interrupt mask register. When this bit is 0, interrupt is not masked and occurs as needed. When the bit is 1, interrupt will not occur because it is masked. Chapter 3 has more information about bit 0 (inside graphic, no. 15) to bit 13 (inside graphic, no. 2).



Inside graphic:
 1. A-Bus interrupt mask bit
 2~15 Interrupt mask bit

Figure 1.17 Interrupt Mask Register Map

Interrupt Status Register

Figure 1.18 shows the map of the interrupt status register. Because this register is able to read and write, when reading it shows that interrupt won't occur when bit data is 0, and will occur when bit data is 1. When writing, interrupt is reset if 0 is written, and maintains the current interrupt status when 1 is written. See chapter 3 for details about this register.

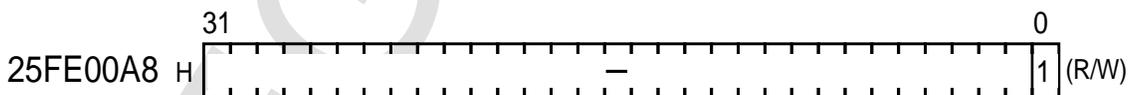


Inside graphic:
 1~30 Interrupt status bit

Figure 1.18 Interrupt Status Register Map

A-Bus Interrupt Acknowledge Register

Figure 1.19 shows a map of the A-Bus interrupt acknowledge. This is a read/write bit that has different meanings when reading vs. when writing. See chapter 3 for details.



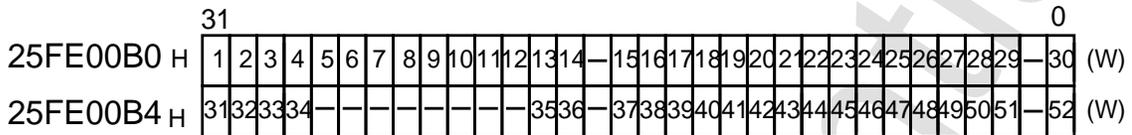
Inside graphic:
 1. READ: A-Bus interrupt acknowledge significant bit (=0:insignificant / =1:significant)
 WRITE: A-Bus interrupt acknowledge significant bit (=0:insignificant / =1:significant)

Figure 1.19 A-Bus Interrupt Acknowledge Register Map



A-Bus Set Register

Figure 1.20 shows the map of the A-Bus set register. Each pre-read significant bit, precharge insertion bit, and external wait significant bit is insignificant at 0 and significant at 1. See chapter 3 for more information.



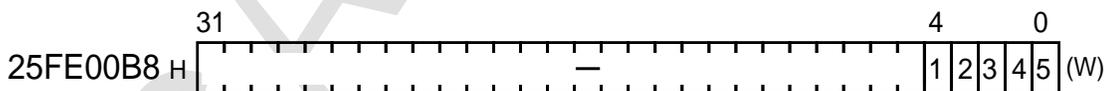
Inside graphic:

- | | |
|---|--|
| <ol style="list-style-type: none"> 1. CS0 space, pre-read significant bit 2. CS0 space, precharge insertion bit after write 3. CS0 space, precharge insertion bit after read 4. CS0 space, external wait significant bit 5~8. CS0 space, burst cycle wait no. set 9~12. CS0 space, single cycle wait no. set 13~14. CS0 space, burst length set 15. CS0 space, bus size set bit (0=16bit 1=8bit) 16. CS1 space, pre-read significant bit 17. CS1 space, precharge insertion bit after write 18. CS1 space, precharge insertion bit after read 19. CS1 space, external wait significant bit 20~23. CS1 space, burst cycle wait no. set 24~27. CS1 space, normal cycle wait no. set 28~29. CS1 space, burst length set bit 30. CS1 space, bus size set bit (0=16bit 1=8bit) | <ol style="list-style-type: none"> 31. CS2 space, pre-read significant bit 32. CS2 space, precharge insertion bit after write 33. CS2 space, precharge insertion bit after read 34. CS2 space, external wait significant bit 35~36. CS2 space, burst length set bit 37. Bus size set bit (0=16 bit 1=8 bit) 38. Spare space, pre-read significant bit 39. Spare space, precharge insertion after write 40. Spare space, precharge insertion after read 41. Spare space, external wait significant bit 42~45. Spare space, burst cycle wait no. set bit 46~49. Spare space, normal cycle wait no. set bit 50~51. Spare space, burst length set bit 52. Spare space, bus size set bit (0=16bit 1=8bit) |
|---|--|

Figure 1.20 A-Bus Set Register Map

A-Bus Refresh Register

Figure 1.21 shows the map of the A-Bus refresh register. This register performs the settings for A-Bus refresh.



Inside graphic:

1. A-Bus refresh output significant bit (=0:insignificant / =1:significant)
- 2~5. A-Bus refresh wait number set bit

Figure 1.21 A-Bus Refresh Register Map

SCU SDRAM Select Register

Figure 1.22 shows the map of the SCU SDRAM select register.



Inside graphic:

1. Work-SDRAM select bit (=0:2 Mbit x2 / =1:4 Mbit x 2)

Figure 1.22 SCU SDRAM Select Register Map

SCU Version Register

Figure 1.23 shows the map of the SCU version register.



Inside graphic:

1~4. Version number'

Figure 1.23 SCU Version Register Map



CHAPTER 2 OPERATION

Chapter 2 Contents

2.1	DMA Transfer	16
	Basic Operation of DMA	16
	DMA Mode	18
	Example of A Specific Use	21
2.2	Interrupt Control	27
	Blanking Interrupt	29
	Timer Interrupt	30
	DSP-End Interrupt	33
	Sound-Request Interrupt	33
	SMPC Interrupt	33
	PAD Interrupt	33
	DMA End Interrupt	33
	DMA-Illegal Interrupt	33
	Sprite Draw End Interrupt	33
2.3	DSP	34
	DSP Control from the Main CPU	34

2.1 DMA Transfer

Basic Operation of DMA

Figure 2.1 shows basic DMA operation. This DMA is basically long word access through the DMA controller buffer, but if the start address and end address are not in long word boundaries, reads and writes are made in byte units, and DMA transfer can be executed.

Figure 2.1 is an example of DMA transfer from transfer source address 1H - 50H to transfer destination address 6H - 55H. However, since the long word boundary in the transfer source is 4H, 1H - 3H is read in byte units. Since the long word boundary in the transfer destination is 8H, the first 2 bytes of read data are written to 6H - 7H in byte units. Moreover, the transfer source end address is 50H, but since the long word boundary is up to 4FH, the data in 50H is read in byte units. On the other hand, since the transfer destination end address is 55H but the long word boundary is up to 53H, the last two bytes read are written to 54H - 55H in byte units.

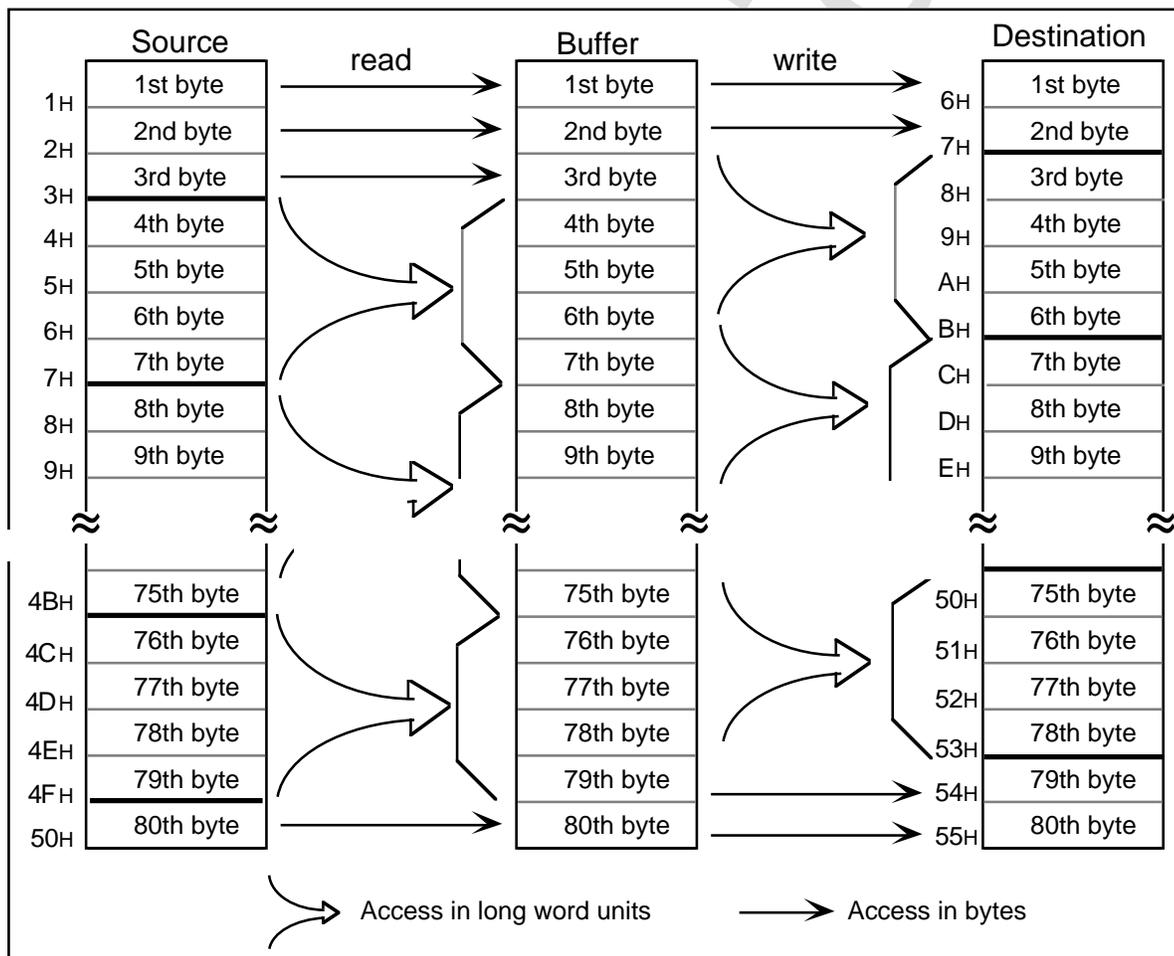


Figure 2.1 DMA Transfer Basic Operation



There are two methods of activating the SCU's DMA transfer control.

- 1) activate DMA from the Main CPU
- 2) activate DMA from the DSP

Figure 2.2 shows the DMA transferable area when activated from the main CPU. Figure 2.3 shows the DMA transferable area when activated from the DSP.

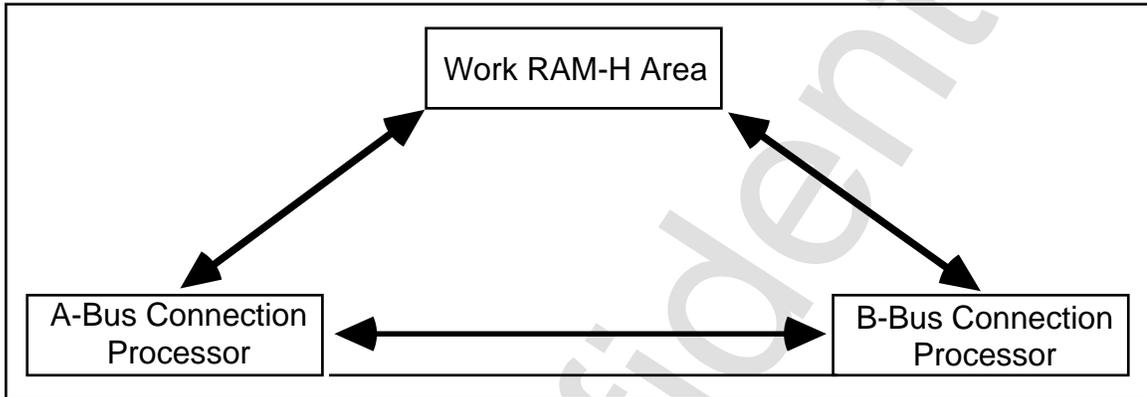


Figure 2.2 DMA Transferable Area when activated from the Main CPU

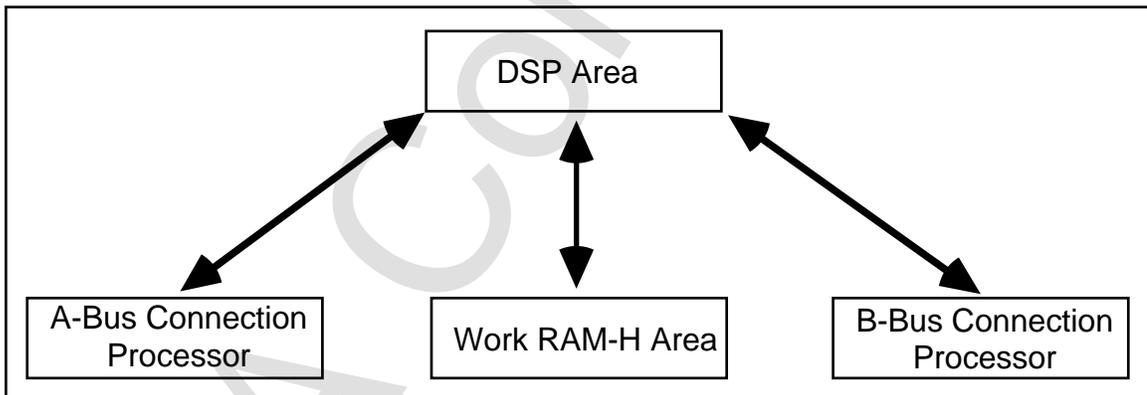


Figure 2.3 DMA Transferable Area when activated from the DSP

DMA Mode

The SCU DMA mode has the following two modes:

- 1) Direct Mode
- 2) Indirect Mode

Direct Mode

Data is transferred only in byte numbers shown as transfer byte numbers directly using address values of separate level DMA set registers, and from the address memory shown by the read address register to the address memory shown by the written address register. One transfer is implemented per start up, then DMA ends. Figure 2.4 shows the DMA transfer operation of the direct mode.

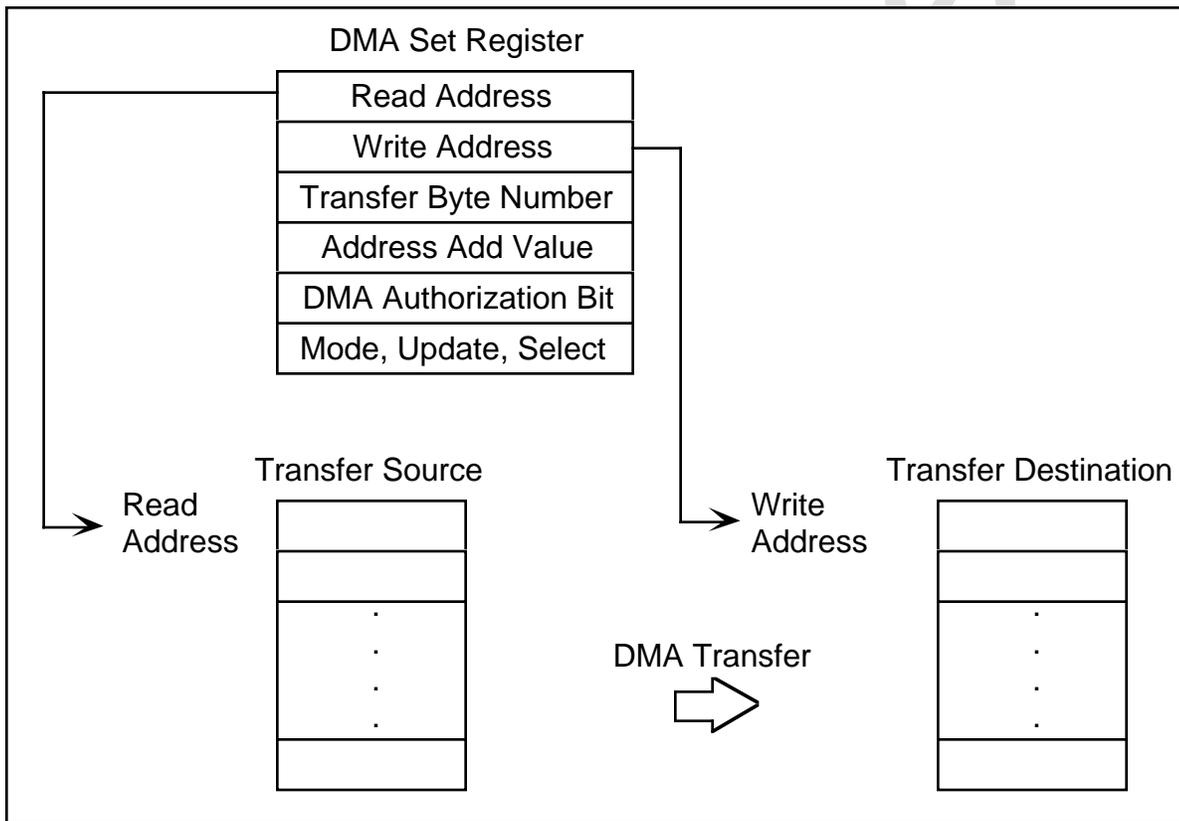


Figure 2.4 Direct Mode DMA Transfer Operation Map



Indirect Mode

The indirect mode implements DMA transfer by indirectly using the DMA set register at a level different from the Direct mode mentioned earlier. The address value and byte number stored by the Direct mode in the set register are stored in the indirect mode temporary buffer by the Indirect mode, and DMA transfer is repeated until the end code is detected. Thus, the Indirect mode can implement more than one DMA transfer when activated once. Figure 2.5 shows the execution flow of Indirect mode DMA.

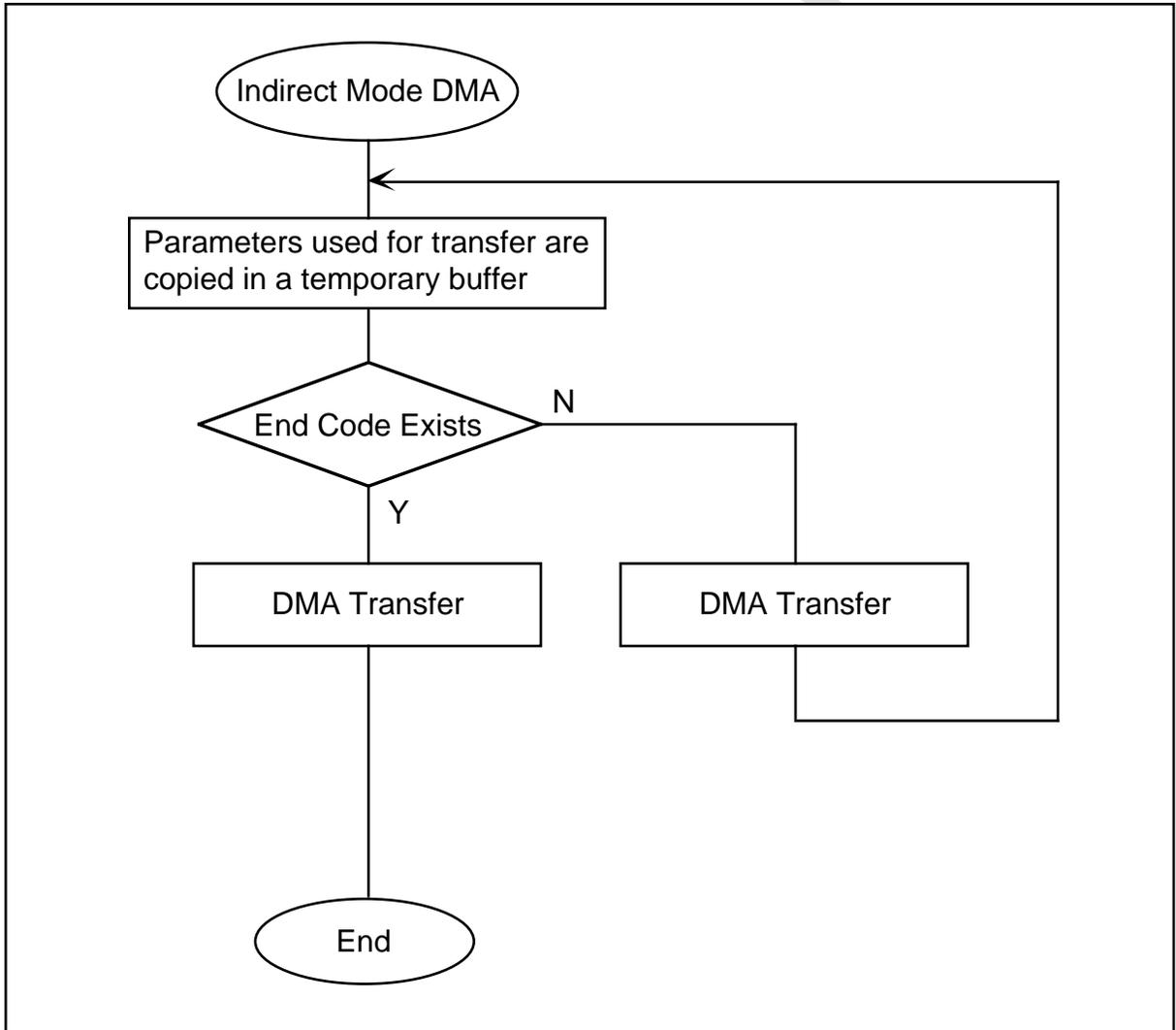


Figure 2.5 Indirect Mode DMA Transfer Flow

When the Indirect mode is activated, parameters of a 3 long word segment from the address first written in the write address register (DxW) is read and stored in a temporary buffer. Next, the actual DMA is executed using the parameters. On completion of DMA, the address parameters of DxW+Ch are read and similarly executed. This operation is repeated until the end code is detected.

The indirect mode address is incremented in 4 byte units.

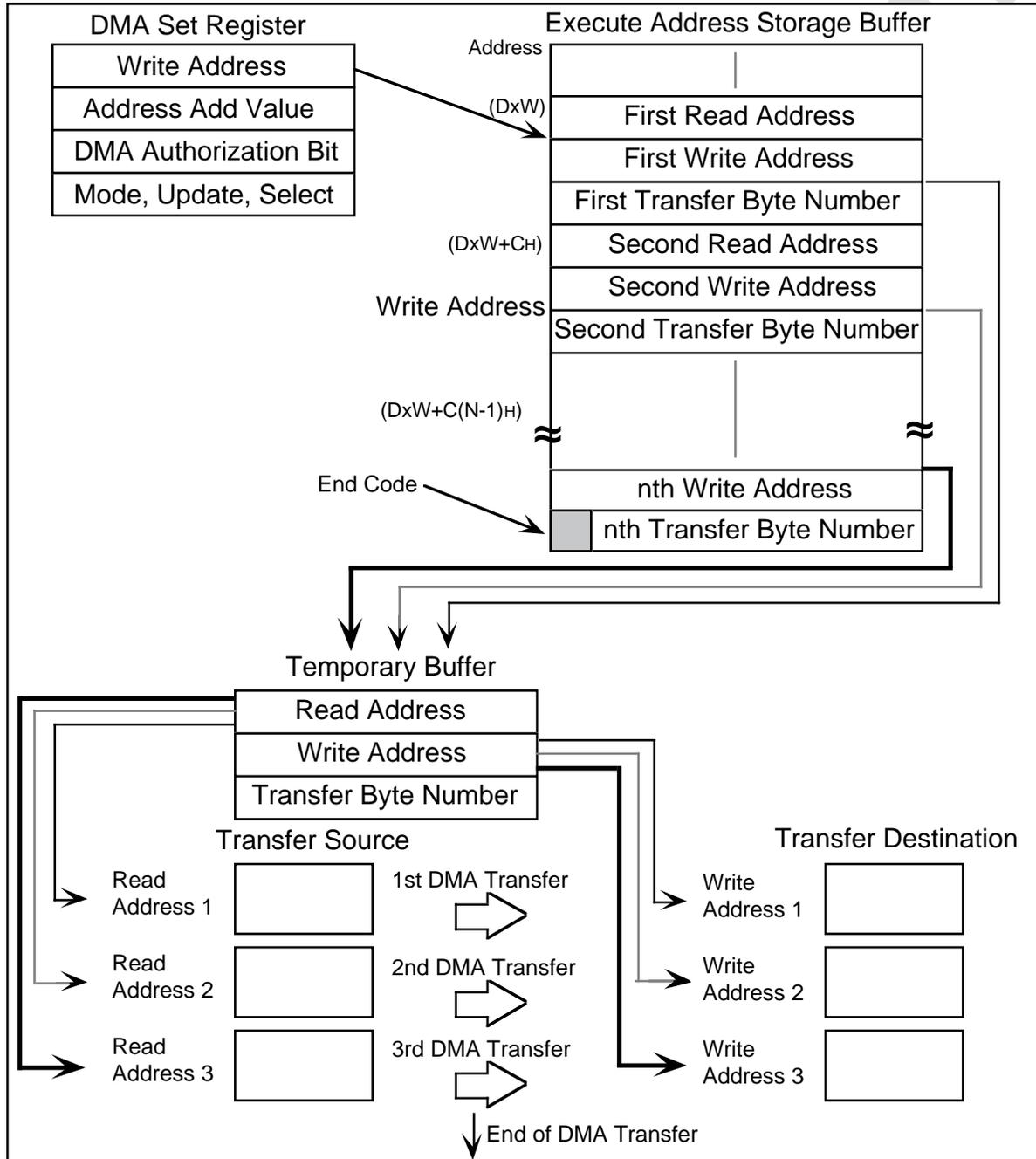


Figure 2.6 Indirect Mode DMA Transfer Operation Details



Example of a Specific Use

Direct Mode

A 1 Kbyte transfer can be thought of as level 0 DMA from address 2000000H (A-Bus area) to address 6000000H (work RAM). DMA (direct mode) can be executed when operating in accordance with the following procedures.

- 1) Write the read address (200000H) to the read address register D0R. (Loads the address that is read to address 25EF0000H from the CPU.)
- 2) Write the write address (6000000H) to the write address register D0W. (Loads the address that is written to address 25EF0004H from the CPU.)
- 3) Write the transfer byte number (400H) to transfer byte number register D0C. (Loads the transfer byte number from the CPU to address 25EF0008H.)
- 4) Write the address add value (101H) to address add value register D0AD. (Loads the address add value from the CPU to address 25EF000CH. Details of the address add value are listed in the address add value of this section. The address add value indicated in the normal DMA is 101H.)
- 5) The DMA mode is 0, and the address update bit and DMA start factor are set as necessary and written to mode/address/update/DMA start factor register D0MD. For example, when address update is handled as the save mode and V-Blank-IN is handled as the start factor, 0 is written to D0MD. (Loads 0 in address 25EF0014H from the CPU.)
- 6) Set 1 in the DMA enable bit. When the start factor set by step 5) occurs, DMA is activated and 1 Kbyte of data is transferred by level 0 from address 2000000H (A-Bus area) to address 6000000H (work RAM).
- 7) After DMA has ended, DMA is activated each time the start factor set in step 5) occurs. The operation at that time changes according to the values of the read address update bit (D0RUP) and write address update bit (D0WUP). Figure 2.7 shows DMA operation changes by the address update bit.

Steps 1) to 5) do not have to be done in the same order. (When the start factor is set in the DMA starting bit, DMA starts each time the DMA operation bit is set to 1 by the CPU.)

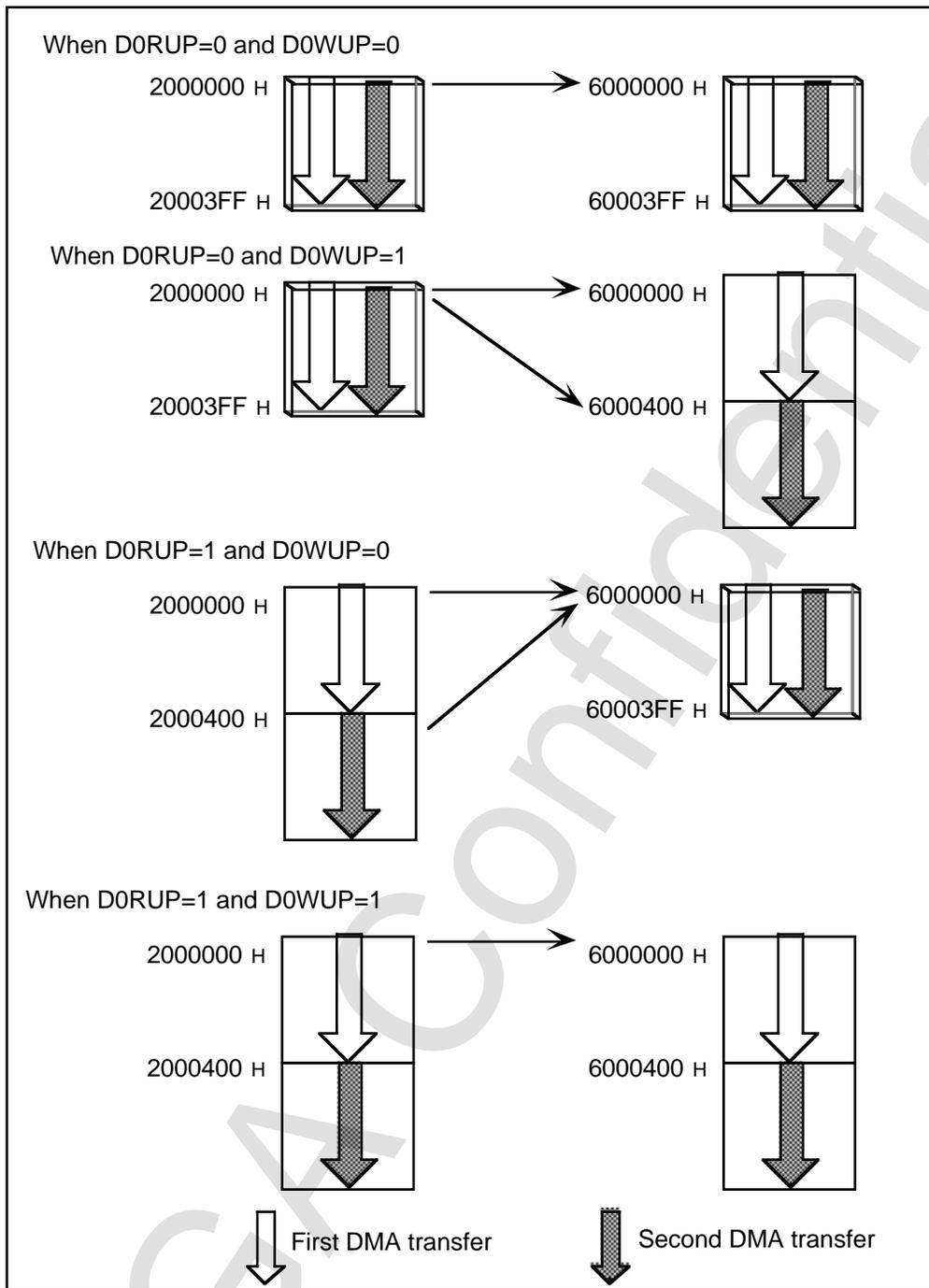


Figure 2.7 Differences in DMA Operations according to the Address update Bit

When the read address update bit is 0, the same address is referred to (read to) both the first and second time. When the read address update bit is 1, the second read starts after the address following the first read.

When the write address update bit is 0, write is executed to the same address for both the first and second time. When the write address update bit is 1, the second write starts after the address following the first write.



Indirect Mode

The Indirect mode is used when executing DMA transfer more than once by starting once. The Indirect mode is not set in a register as is the Direct mode, but uses a method of executing DMA by accessing the register through RAM. For example, consider a case in which three DMA transfers are to be continuously (consecutively) executed at level 0 through work RAM area (6000000H).

- (a) 20HByte DMA transfer from 4000000H to 5C00000H
- (b) 10HByte DMA transfer from 5E00000H to 6080000H
- (c) 15HByte DMA transfer from 5A00000H to 6081000H

DMA (Indirect mode) can be executed if operated in accord with the following steps.

- 1) As shown in Figure 2.8, data is written in long word units from the work RAM area (6000000H).

6000000H	4000000H
	5C00000H
	20H
600000CH	5E00000H
	6080000H
	10H
6000018H	5A00000H
	6081000H
	80000015H
6000024H	

Figure 2.8 Example of Data Write

- 2) DMA parameter source address (6000000H) is written to the write address register (D0W).
- 3) The address add value (101H) is loaded to the address add value register D0AD. (The address add value is written from the CPU to address 25FE000CH.) Information on the address add value is described in the address add value of this section. The address add value indicates 101H in normal DMA.
- 4) The DMA mode is 1 and the address update bit and DMA start factor are set as required and written to mode/address/update /DMA start factor register D0MP. For example, when address update is handled as the retain mode and V-Blank-IN is handled as the start factor, 1000000H is written to D0MD. (Loads 1000000H in address 25FE0014H from the CPU.)

5) "1" is set in the DMA enable bit, DMA is activated when the start factor set by step 4) occurs. DMA transfer (a) to (c) is executed in order until the DMA end code is detected. The DMA end code is the end notification code of the DMA indirect mode that exists only in the work RAM area. DMA transfer continues as long as "1" of this bit remains undetected.

Steps 1) to 4) do not need to be done in the same order. The read address register (D0R), transfer byte number register (D0C), and address add value register (D0AD), which must be set in the Direct mode, do not need to be set in the Indirect mode.

When the DMA transfers listed below are registered in memory, DMA transfer is restarted after the above process ends. Restart can be done only by repeating the operation in step (4) above.

(d) 30HByte DMA transfer from 5000000H to 6100000H.

(e) 25HByte DMA transfer from 5100000H to 6200000H.

The contents from the work RAM area 6000000H are shown below in Figure 2.9. DMA starts each time the start factor set by (5) occurs.

6000000H	4000000H
	5C00000H
	20H
600000CH	5E00000H
	6080000H
	10H
6000018H	5A00000H
	6081000H
	80000015H
6000024H	5000000H
	6090000H
	30H
6000030H	5100000H
	60A0000H
	80000025H
600003CH	

Figure 2.9 Work RAM Area Contents



The operation at restart differs depending on whether the DMA mode is in save mode or update mode. Recognition of the save/update mode of the Indirect mode is performed and judged by the write address update bit.

- For Save mode (write address update bit = 0), after one DMA transfer is completed, because the address accessing the parameters is saved at 6000000H, (a) ~ (c) DMA transfer is re-implemented.
- For update mode (write address update bit = 1), after one DMA transfer is completed, because the address accessing the parameters is updated at 6000024H, (d) ~ (e) DMA transfer is implemented.

Address Add Value

DMA normally accesses continuous areas, but by setting the address add value, the addresses of fixed intervals can be accessed. This function is effective when changing part of continuously arranged parameters like the VDP1 command table. An example is 32 blocks as one 20H byte table from address 5C00000H, among which the parameters of each 8 byte block are rewritten one time. Change parameters that have 40H bytes from address 6000000H are set by the following steps and the transfer process is implemented when transferring via level 0 of DMA.

- 1) Write the read address 6000000H to read address register D0R.
- 2) Write the write address 5C00008H to write address register D0W.
- 3) Write transfer byte number 40H to transfer byte number register D0C.
- 4) Write the address add value 105H to address add value register D0AD. Here, the low 3 bits (5=101B) updates the address for each 20H.
- 5) Set the DMA mode to 0 and set the address update bit and DMA start factor as required. Write to the mode/address/update/DMA start factor register D0MD. For example, 0 is written to D0MD when V-Blank-IN is the starting factor and address update is in a retain mode.

- 6) Set the DMA enable bit to 1. DMA is activated when the starting factor set in step 5) occurs and the slanted line area in Figure 2.10 is rewritten once.

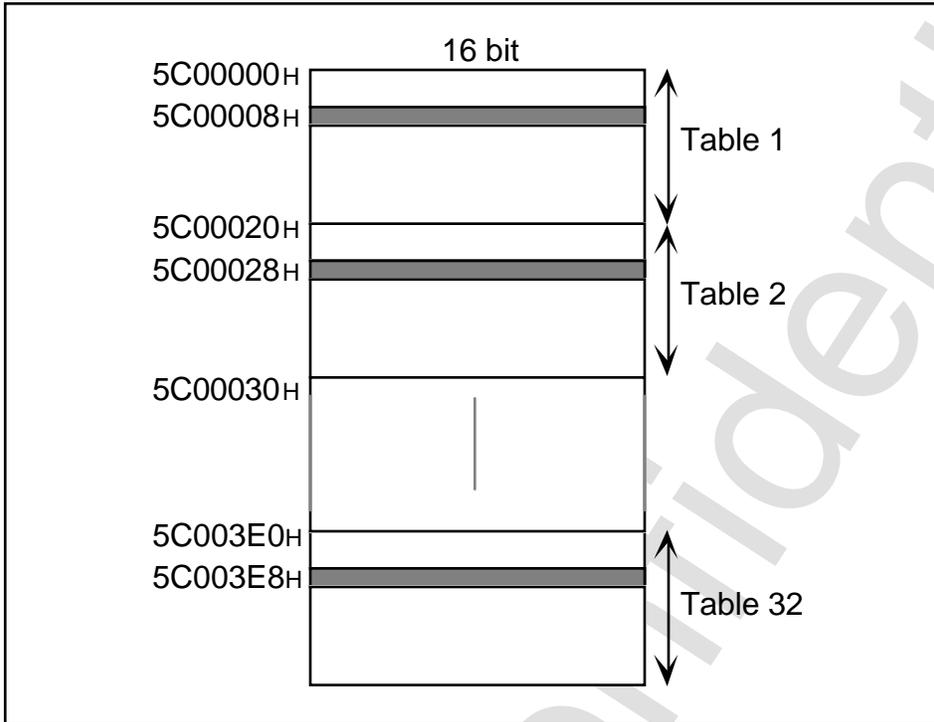


Figure 2.10 DMA Transfer by Setting Address Add Value

Steps 1) through 5) do not have to be in the same order.



2.2 Interrupt Control

Table 2.1 shows the bit allocation of interrupt factors. Bit allocation shows the interrupt register status. Level 1 is the lowest interrupt level and level F is the highest. Details are given below for each interrupt factor.

Table 2.1 Interrupt Factors

Bit Allocation	Interrupt Factors	Interrupt Source	Vector Number	Level
bit 0	V-Blank-IN	VDP2	Vector 40	Level F
bit 1	V-Blank-OUT	VDP2	Vector 41	Level E
bit 2	H-Blank-IN	VDP2	Vector 42	Level D
bit 3	Timer 0	SCU	Vector 43	Level C
bit 4	Timer 1	SCU	Vector 44	Level B
bit 5	DSP End	SCU	Vector 45	Level A
bit 6	Sound Request	SCSP	Vector 46	Level 9
bit 7	System Manager	SM	Vector 47	Level 8
bit 8	PAD Interrupt	PAD	Vector 48	Level 8
bit 9	Level-2 DMA End	A-Bus	Vector 49	Level 6
bit 10	Level-1 DMA End	A-Bus	Vector 4A	Level 6
bit 11	Level-0 DMA End	A-Bus	Vector 4B	Level 5
bit 12	DMA-illegal	SCU	Vector 4C	Level 3
bit 13	Sprite Draw End	VDP1	Vector 4D	Level 2
bit 14	--			
bit 15	--			
bit 16	External Interrupt 00	A-Bus	Vector 50	Level 7
bit 17	External Interrupt 01	A-Bus	Vector 51	Level 7
bit 18	External Interrupt 02	A-Bus	Vector 52	Level 7
bit 19	External Interrupt 03	A-Bus	Vector 53	Level 7
bit 20	External Interrupt 04	A-Bus	Vector 54	Level 4
bit 21	External Interrupt 05	A-Bus	Vector 55	Level 4
bit 22	External Interrupt 06	A-Bus	Vector 56	Level 4
bit 23	External Interrupt 07	A-Bus	Vector 57	Level 4
bit 24	External Interrupt 08	A-Bus	Vector 58	Level 1
bit 25	External Interrupt 09	A-Bus	Vector 59	Level 1
bit 26	External Interrupt 10	A-Bus	Vector 5A	Level 1
bit 27	External Interrupt 11	A-Bus	Vector 5B	Level 1
bit 28	External Interrupt 12	A-Bus	Vector 5C	Level 1
bit 29	External Interrupt 13	A-Bus	Vector 5D	Level 1
bit 30	External Interrupt 14	A-Bus	Vector 5E	Level 1
bit 31	External Interrupt 15	A-Bus	Vector 5F	Level 1

Table 2.2 shows by what general names the interrupt factors are called. Later descriptions are based on the general name.

Table 2.2 Interrupt Factor General Names

General Names	Specific Names
Blanking Interrupt	V-Blank-IN
	V-Blank-OUT
	H-Blank-IN
Timer Interrupt	Timer 0
	Timer 1
DMA End Interrupt	Level 2-DMA End Interrupt
	Level 1-DMA End Interrupt
	Level 0-DMA End Interrupt



Blanking Interrupt

There are three types of blanking interrupt, V-Blank-IN, V-Blank-OUT, and H-Blank-IN. Figure 2.11 details blanking interrupt. Blanking interrupt is synchronous to the display, and notifies the user whether a drawing is at the beginning or end.

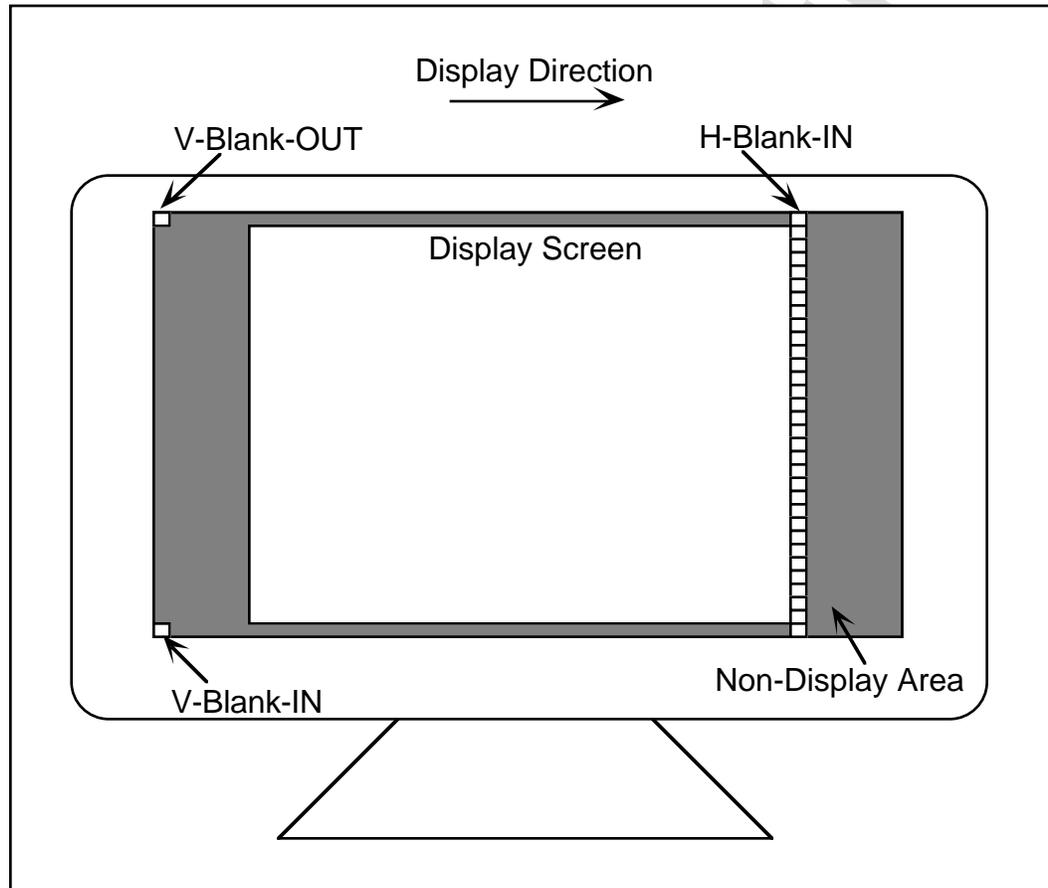


Figure 2.11 Blanking Interrupt

V-Blank-IN

Indicates the end of a display, after which nothing will be displayed on the screen even when attempting to display data.

V-Blank-OUT

V-Blank-OUT indicates the beginning of a display. Although a display may be about to begin, how long before interrupt occurs must be taken into consideration since it takes time (an interval) for the actual display to materialize. V-Blank-OUT also clears Time 0 data.

H-Blank-IN

H-Blank-IN indicates the draw end of one line. Timer 0 data is incremented by this timing.

Timer Interrupt

Time interrupt includes Timer 0 and Timer 1. Time interrupt is synchronous with the blanking interrupt mentioned earlier and can cause interrupt to occur at dots (points) on the screen.

Timer 0

Values are cleared by V-Blank-OUT interrupt reception and counted by H-Blank-IN interrupt reception. Timer 0 interrupt occurs when values compared to the Timer 0 compare register (see register details) are the same. Figure 2.12 shows the Timer 0 occurrence process.

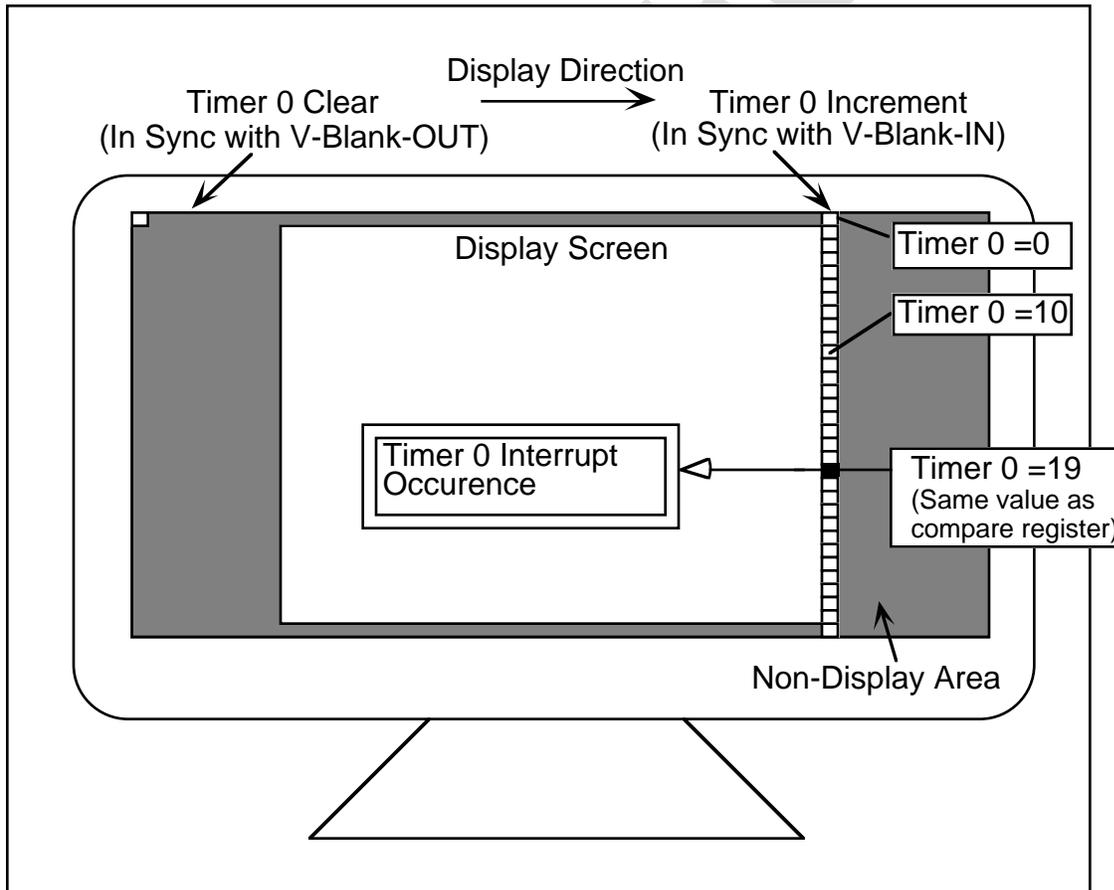


Figure 2.12 Timer 0 Interrupt Occurrence Process
(compare register = example when set to 19)



Timer 1

Data of the Timer 1 data set register (see register details) is set by Timer 1 with H-Blank-In interrupt receiving. Count down is done at a frequency (1 dot painting) of 7 MHz or about 1/4 the system clock. When the value of Timer 1 becomes 0, interrupt of Timer 1 occurs. Interrupt can also be made to occur at 1 point by combining it with Timer 0 according to the Timer 1 mode register value (see register details), and interrupt can be caused to occur at each line independently of Timer 0. Figure 2.13 shows the process up to when Timer 1 interrupt is caused to occur in sync with Timer 0.

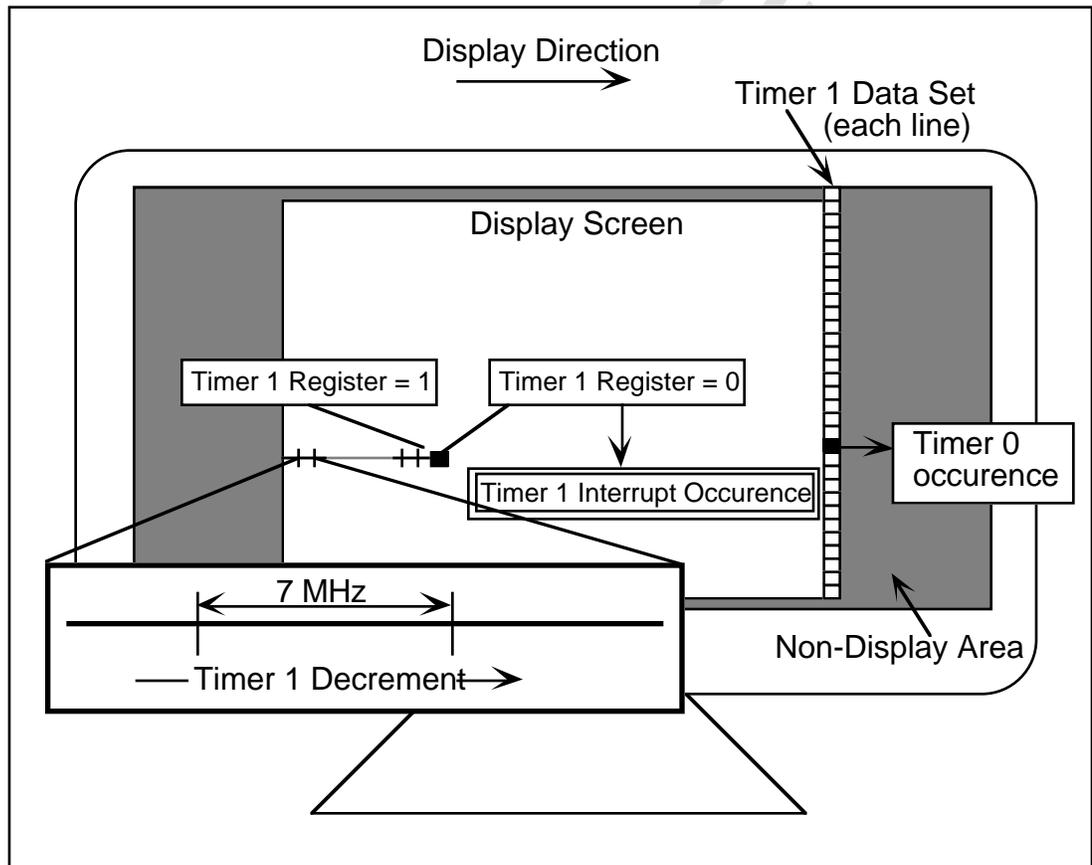


Figure 2.13 Timer 1 Interrupt Process (In sync with Timer 0)

Figure 2.14 shows the process up to when Timer 1 is caused to occur out of sync with Timer 0. There is no change when operationally in sync but a judgment is made for each line and interrupt made to occur.

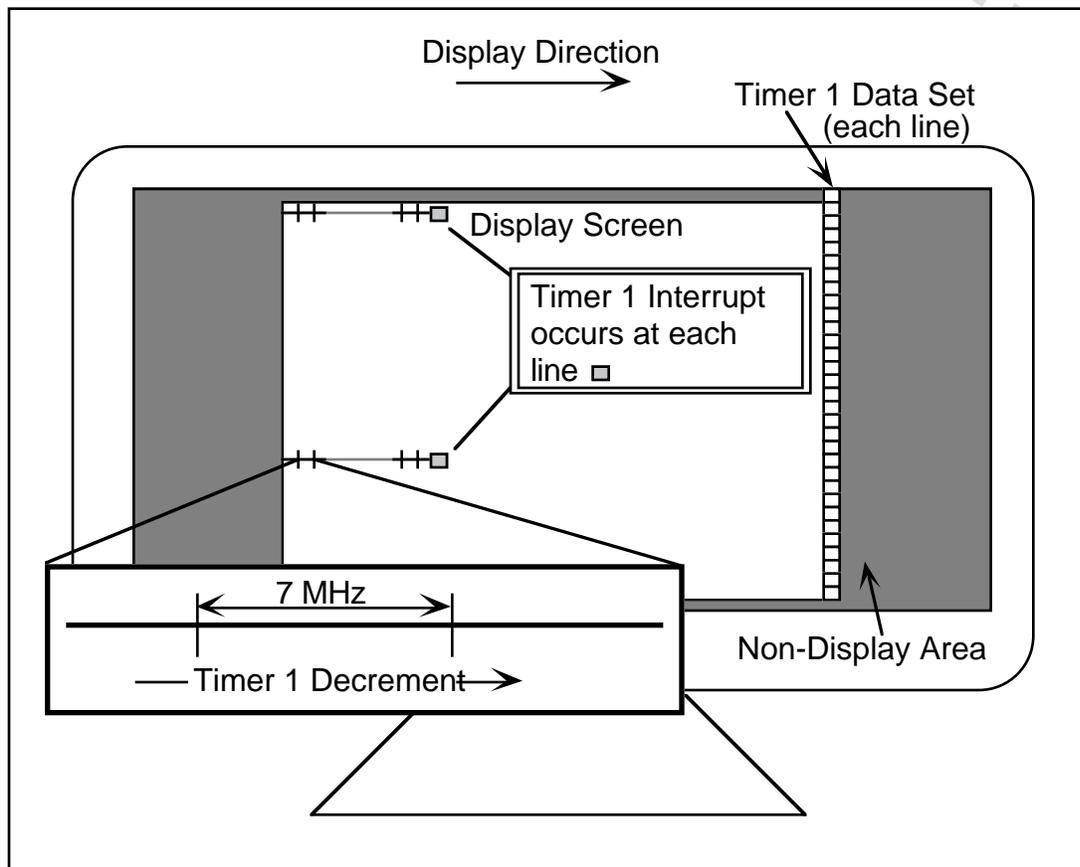


Figure 2.14 Timer 1 Interrupt Process (out of sync with Timer 0)



DSP-End Interrupt

The program execution control flag (see section 3.3, *E flag of the Program Control Port*) of the program control port (see section 3.3, *Program Control Port*) is set by the DSP ENDI command (see section 4.5, *“Command” ENDI command*) and gives notice when the program has ended. By this, the main CPU can retrieve the results calculated by the DSP.

Sound-Request Interrupt

This interrupt occurs from the SCSP. For example, to display the volume level meter on the screen when a CD (Compact Disk) is connected, interrupt from SCSP is used and reported to the main.

SMPC Interrupt

Detailed information about interrupt that occurs from SMPC is listed in the SMPC User's Manual.

PAD Interrupt

The occurrence of this interrupt depends on the action of the user. PAD is given as one example but other items, such as a mouse, may be connected.

DMA End Interrupt

Divided by level, this interrupt notifies the user when DMA transfer has ended. There are three DMA levels from level 2 to level 0.

DMA-Illegal Interrupt

Notifies user that DMA cannot be executed by interrupt when executing DMA that cannot be done using certain parameters.

Sprite Draw End Interrupt

Notifies user via VDP1 that draw has ended.

SEGA

2.3 DSP

DSP Control from the Main CPU

This allows control of the DSP from the main CPU. DSP items that can be controlled from the CPU include:

- 1) Load DSP program
- 2) Access DSP data
- 3) Begin DSP program execution
- 4) Forced stop of DSP program

Load DSP Program

There are two methods in which the DSP program is loaded: by using the DSP DMA command, and by writing directly to the DSP program RAM area from the main CPU. Program data can be loaded if controlled from the main CPU in the order shown below.

- 1) Set the program control port bits 16 and 17 to 0.
- 2) Write the transfer start address to the program RAM address of the same port.
If DSP is not stopped, it cannot be loaded.
- 3) Write sequence program data in long word units to the program RAM data port.

Figures 2.15 to 2.17 show each step of control from the CPU.

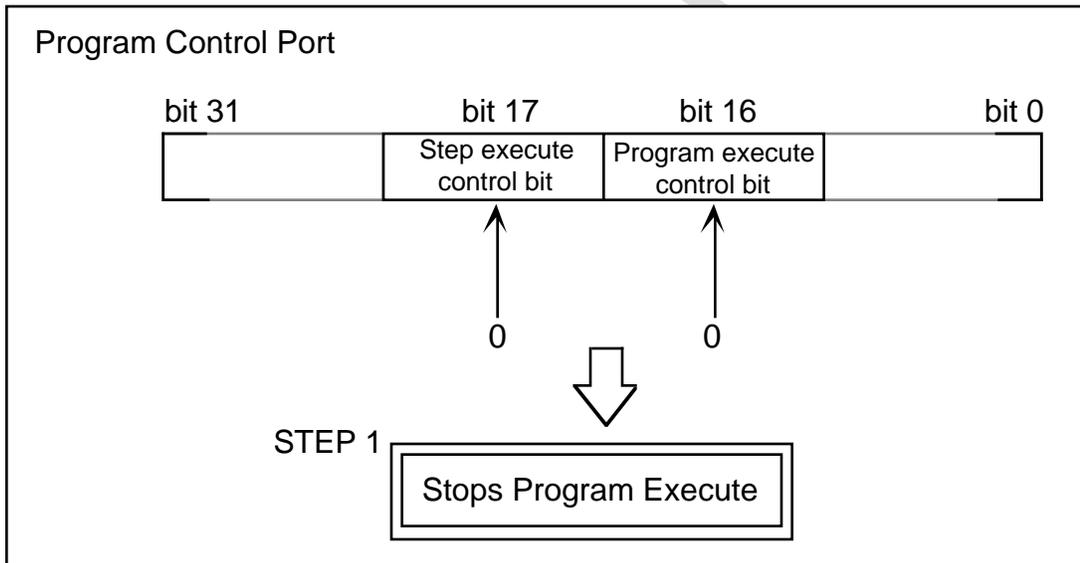


Figure 2.15 DSP Program Load Step 1



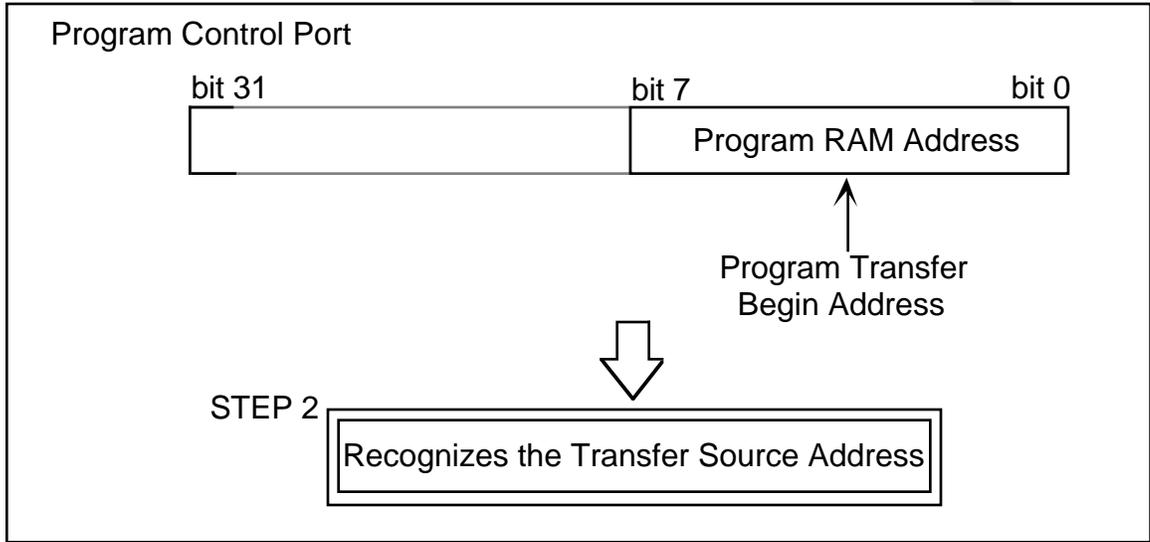


Figure 2.16 DSP Program Load Step 2

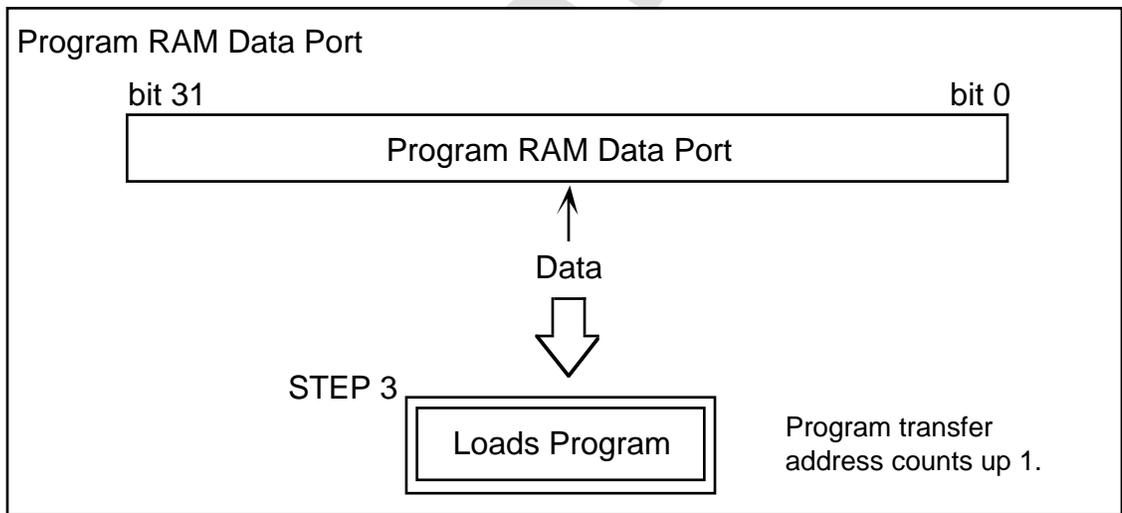


Figure 2.17 DSP Program Load Step 3

SECRET

DSP Data Access

In order to access DSP data, the DMA command of DSP can be used, but there is also a method that accesses the DSP data RAM area from the main CPU. Data can be accessed if controlled from the CPU in the following sequence.

- 1) Set the program control port bit 16 and bit 17 to 0.
- 2) Write the access start address to the data RAM address port.
If DSP is not stopped, it cannot be set.
- 3) Sequence data is accessed in long-word units through the data RAM data port.

Control methods from the CPU for each step are shown from Figure 2.18 to Figure 2.20.

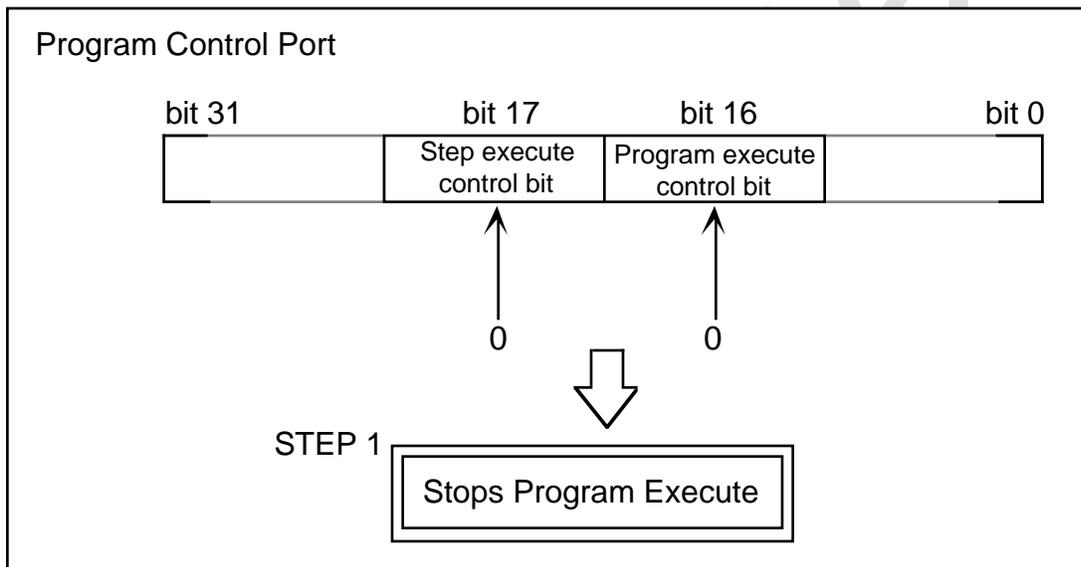


Figure 2.18 DSP Data Access Step 1



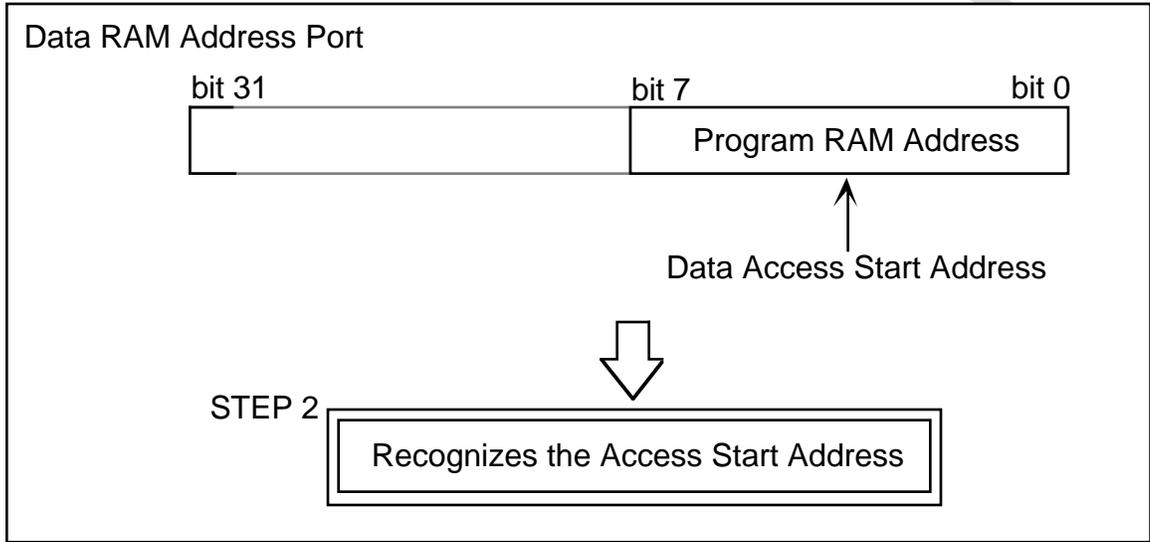


Figure 2.19 DSP Data Access Step 2

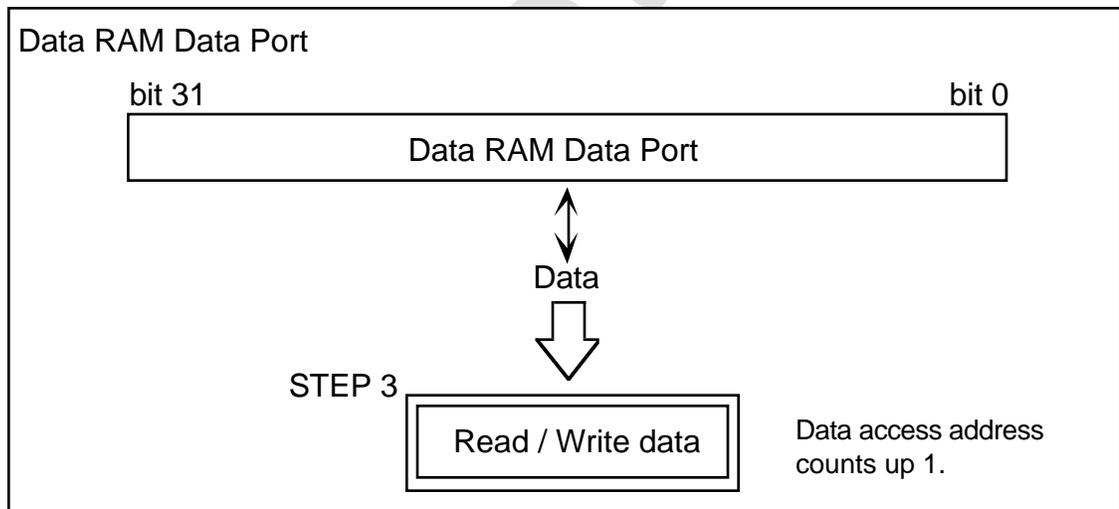


Figure 2.20 DSP Data Access Step 3

DSP Program Execute Start

Execution of the DSP program is begun by writing of the program control port 1 to bit 16 (see Figure 2.21). When the write is recognized, DSP begins execution from the address stored in the program RAM address of the program control port. The execution start address must be set before writing "1" to bit 16 of the program control port.

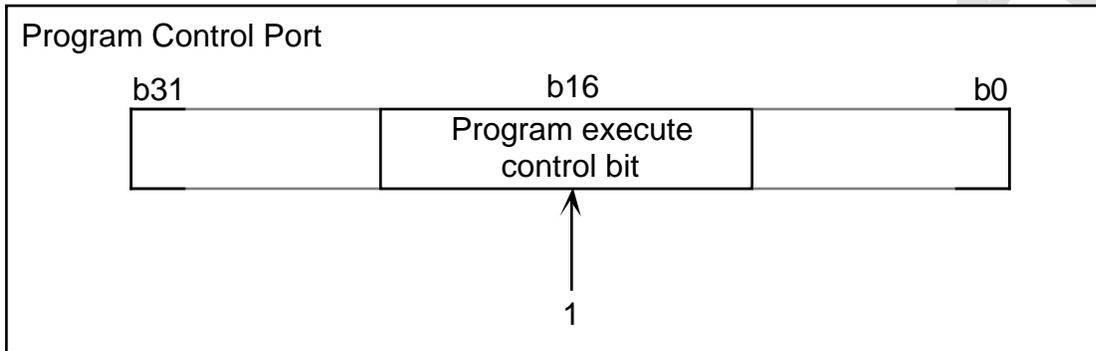


Figure 2.21 DSP Program Execution Start Control from CPU

DSP Program Forced Stop

In contrast to execution start, DSP program execution forced stop is done by writing the program control port 0 to bit 16 of the program control port. Figure 2.22 shows the forced stop control.

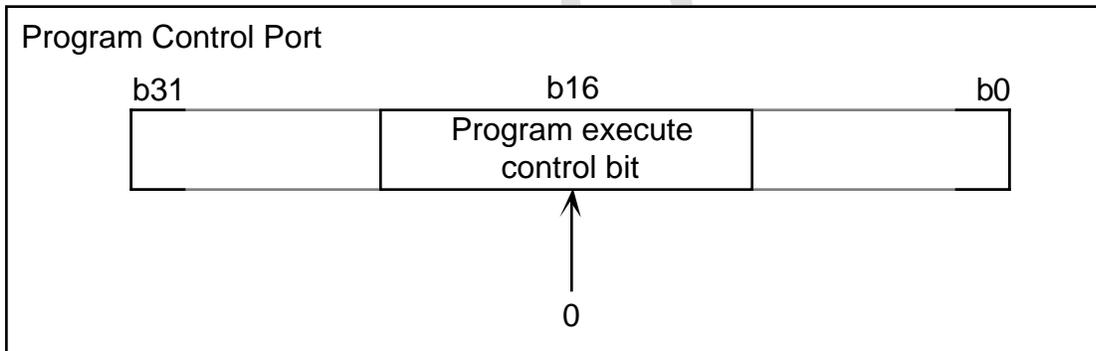


Figure 2.22 DSP Program Forced Stop Control from CPU



CHAPTER 3 REGISTERS

Chapter 3 Contents

3.1	Register List	40
3.2	DMA Control Registers	41
	Level 2-0 DMA Set Register	41
	DMA Enable Register	45
	DMA Mode, Address Update, Start Factor Select Register	46
	DMA Forced Stop Register	47
	DMA Status Register	47
3.3	DSP Control Ports	51
	DSP Program Control Port	51
	DSP Program RAM Data Port	53
	DSP Data RAM Address Port	53
	DSP Data RAM Data Port	54
3.4	Timer Registers	55
	Timer 0 Compare Register	55
	Timer 1 Set Data Register	55
	Timer 1 Mode Register	56
3.5	Interrupt Control Registers	57
	Interrupt Mask Register	57
	Interrupt Status Register	58
3.6	A-Bus Control Registers	61
	A-Bus Interrupt Acknowledge Register	61
	A-Bus Set Register	62
	A-Bus Refresh Register	72
3.7	SCU Control Registers	73
	SCU SDRAM Select Register	73
	SCU Version Register	73

3.1 Register List

A list of SCU registers is given in Table 3.1. Headings are divided for each register type and each register is explained.

Table 3.1 Register List

Type	Register Name	Lead Address	End Address	Size
DMA Control Registers	Level 0-DMA Set Register	25FE0000H	25FE0017H	24 byte
	Level 1-DMA Set Register	25FE0020H	25FE0037H	24 byte
	Level 2-DMA Set Register	25FE0040H	25FE0057H	24 byte
	DMA Force-End Register	25FE0060H	25FE0063H	4 byte
	DMA Status Register	25FE007CH	25FE007FH	4 byte
DSP Control Ports	DSP Program Control Port	25FE0080H	25FE0083H	4 byte
	DSP Program RAM Data Port	25FE0084H	25FE0087H	4 byte
	DSP Data RAM Address Port	25FE0088H	25FE008BH	4 byte
	DSP RAM Data Port	25FE008CH	25FE008FH	4 byte
Timer Registers	Timer 0 Compare Register	25FE0090H	25FE0093H	4 byte
	Timer 1 Set Data Register	25FE0094H	25FE0097H	4 byte
	Timer 1 Mode Register	25FE0098H	25FE009BH	4 byte
Interrupt Control Registers	Interrupt Mask Register	25FE00A0H	25FE00A3H	4 byte
	Interrupt Status Register	25FE00A4H	25FE00A7H	4 byte
A-Bus Control Registers	A-Bus Interrupt Acknowledge	25FE00A8H	25FE00ABH	4 byte
	A-Bus Set Register	25FE00B0H	25FE00B7H	8 byte
	A-Bus Refresh Register	25FE00B8H	25FE00BBH	4 byte
SCU Control Registers	SCU SDRAM Select Register	25FE00C4H	25FE00C7H	4 byte
	SCU Version Register	25FE00C8H	25FE00CBH	4 byte



3.2 DMA Control Registers

Level 2-0 DMA Set Register

There are three DMA levels, beginning at the highest priority level of 2 to the lowest priority level of 0. These are explained below.

- Read Address

Figure 3.1 is the read address register. The DMA mode includes a direct mode and an indirect mode. The value of the meaning changes for each mode.

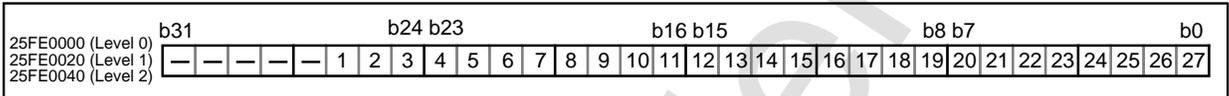


Figure 3.1 Level 2-0 Read Address (Register: D0R, D1R, D2R) Initial value undefined

Read Address (1~27 [bit 26 ~ 0] in Figure 3.1)

DxR 26-0[x=2-0] (R/W) DMA level 2-0 Read address bit 26-0

When in the Direct mode, values being stored are transfer source addresses. However, this has no meaning when in the Indirect mode. The register of that level prohibits writing while DMA is operating. All address values are expressed in bytes.

- Write Address

The write address register is shown in Figure 3.2. The DMA mode includes a direct mode and indirect mode; the value of the meaning changes with each mode.

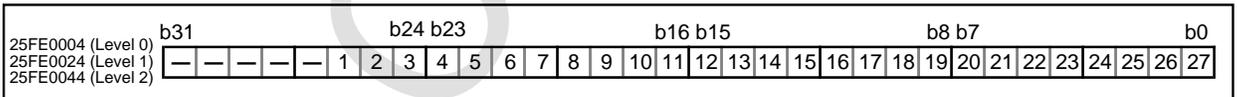


Figure 3.2 Level 2-0 Write Address (Register: D0W, D1W, D2W) Initial value undefined

Write Address (1~27 [bit 26 ~ 0] in Figure 3.2)

DxW 26-0[x=2-0] (R/W) DMA level 2-0 Write address bit 26-0

When in the Direct mode, the value being stored is the transfer source address. However, when in the Indirect mode, the address of the location where the transfer source address of DMA transfer is executed the first time is stored. The register of that level prohibits writing while DMA is operating. All address values are expressed in bytes.

Read Address Add Value (1 [bit 8] in Figure 3.5)

DxRA[x=2-0] (W) DMA level 2-0 Read address Addition data bit

Designates the add byte number of the read address. Table 3.2 shows the read address add value. Since this is effective only for the CS2 space of the A-Bus, everything else should set 1B. The register of that level prohibits writing while DMA is operating.

Table 3.2 Read Address Add Value

DxRA (X=2-0)	Description
0	Nothing is added
1	4 Bytes are added

Write Address Add Value (2~4 [bit 2~0] in Figure 3.5)

DxWA3-0[x=2-0] (W) DMA level 2-0 Write address Addition data bit 3-0

Designates the add byte number of the write address. Table 3.3 shows the write address add value. This value is always effective when writing data to the B-Bus, but is effective only for 000B or 010B data when writing to the CS2 space of the A-Bus. Data should be set to 010B when writing anywhere except to A-Bus or B-Bus. The register of that level prohibits writing while DMA is operating.

Table 3.3 Write Address Add Value

DxWA (X=2-0)	Description
000B	Nothing is added
001B	2 Bytes are added
010B	4 Bytes are added
011B	8 Bytes are added
100B	16 Bytes are added
101B	32 Bytes are added
110B	64 Bytes are added
111B	128 Bytes are added

There are provisions (as in Figure 3.6) for the write address add value. As shown in Figure 3.6, communication between the SCU and B-Bus is in 32-bit units, but in 16-bit units between the B-Bus and processor. Thus, when transferring A ~ D data from the SCU to the processor, as shown in Figure 3.7, the SCU can transfer A ~ D to the B-Bus at one time but the B-Bus can only transfer to the processor after dividing A ~ B and C ~ D. From this, the difference between address 2 and address 1 can be written and indicated as the address add value since the write address add value of B-Bus is 2 byte units, as shown in Figure 3.8.

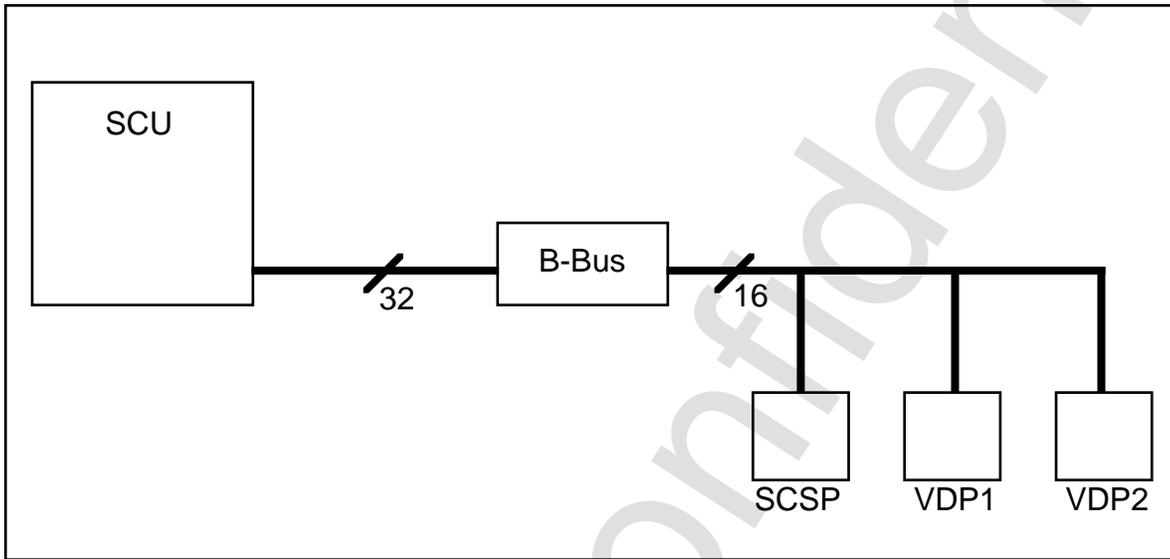


Figure 3.6 Communication Units Between the SCU and Processor

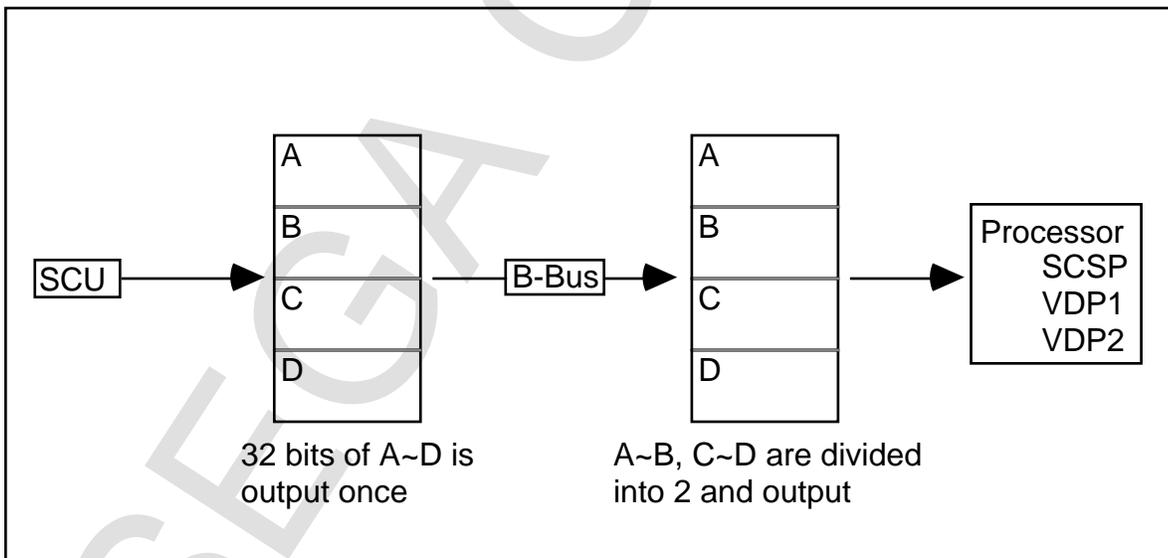


Figure 3.7 Specific Example of Transfer Between the SCU and Processor



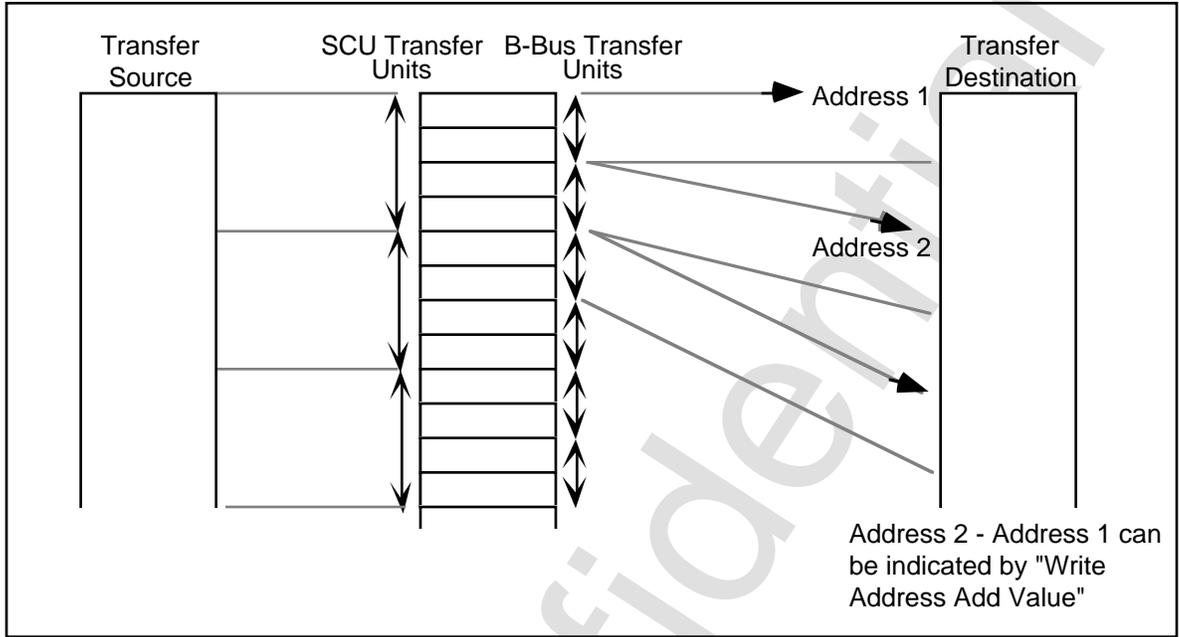


Figure 3.8 Write Address Add Value Indication

DMA Enable Register

This register enable execution of DMA. The register of that level prohibits writing while DMA is operating. Figure 3.9 shows the format of this register.

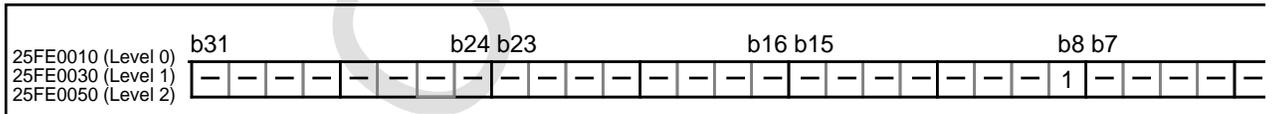


Figure 3.9 Level 2-0 DMA Enable Bit (Register: D0EN, D1EN, D2EN) Initial Value 0000000H

DMA Enable Bit (1 [bit 8] in Figure 3.9)

DxEN[x=2-0] (W) DMA level 2-0 ENable bit

This bit enables DMA to be executed. This flag is set to 1 when DMA is enabled. Other required data must be set in advance since DMA begins after the flag is set.

DMA Starting Bit (2 [bit 0] in Figure 3.9)

DxGO[x=2-0] (W) DMA level 2-0 GO bit

This bit starts execution of DMA. The starting factor bit is significant only when 111B, and when DMA is started, this bit is set to 1. DMA starts one time per set.

DMA Mode, Address Update, Start Factor Select Register

This register designates the DMA mode (direct or indirect), address update (save or update set value), and selection of the start factor. Registers of that level prohibit writing while DMA is operating. Figure 3.10 shows the register.

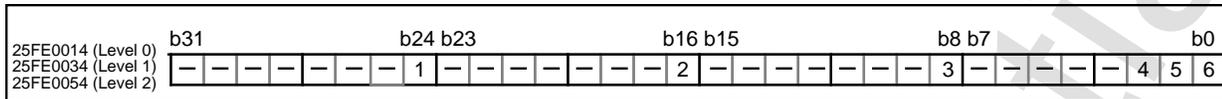


Figure 3.10 Level 2-0 DMA Mode, Address Update, Start Factor Select Register (Register : D0MD, D1MD, D2MD) Initial Value 00000007H

DMA Mode Bit (1 [bit 24] in Figure 3.10)

DxMOD[x=2-0] (W) DMA level 2-0 MODE bit

Decides the DMA mode. "0" shows the direct mode, and "1" shows the indirect mode.

Read Address Update Bit (2 [bit 16] in Figure 3.10)

DxRUP[x=2-0] (W) DMA level 2-0 Read update UP bit

This bit decides whether to save or update the value at the time it is set for read address. 0 means save and 1 means update. See "Example of a Specific Use" in section 2.1 "DMA Transfer" for more information on how to operate it.

Write Address Update Bit (3 [bit 8] in Figure 3.10)

(DxWUP[x=2-0] (W) DMA level 2-0 Write update UP bit

This bit decides whether to save or update the value at the time it is set for write address. "0" means save and "1" means update. See "Example of A Specific Use" in section 2.1 "DMA Transfer" for more information on how to operate it.

DMA Starting Factor Select Bit (4~6 [bit 2~0] in Figure 3.10)

DxFT2-0[x=2-0] (W) DMA level 2-0 starting FacTor bit 2-0

DMA sets the DMA enable bit and starts by receiving an outside signal selected by the starting factor select bit. When the starting factor bit is 111B, DMA starts by setting the DMA starting bit.

Table 3.4 Starting Factors

Starting Factor Bits (x=2-0)			Starting Factors
DxFT2	DxFT1	DXFT0	
0	0	0	V-BLANK-IN signal receive and enable bit setting
0	0	1	V-BLANK-OUT signal receive and enable bit setting
0	1	0	H-BLANK-IN signal receive and enable bit setting
0	1	1	Timer 0 signal receive and enable bit setting
1	0	0	Timer 1 signal receive and enable bit setting
1	0	1	Sound Req signal receive and enable bit setting
1	1	0	Sprite draw end signal receive and enable bit setting
1	1	1	Enable bit setting and DMA starting factor bit setting



DMA Forced Stop Register

This is a bit in DMA control which causes DMA forced stops. This register is positioned at address 05FE0060H (32 bit area) within the SCU. Its operation is shown by the map below.

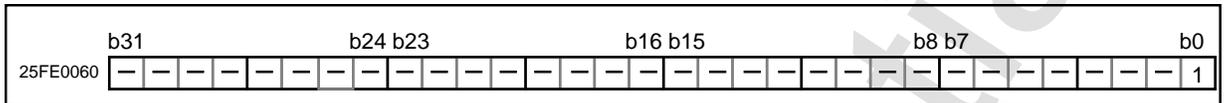


Figure 3.11 DMA Force-Stop Register (Register: DSTP) Initial Value 00000000H

DMA Force-Stop bit (1 [bit 0] in Figure 3.11)

DSTOP (W) DMA STOP control bit

DSTOP=1 : Stops DMA while in operation.

DMA Status Register

- Access, Interruption, Stand by, Operation Registers

This register shows the DMA bus access indication and the DMA condition for each level. The four DMA conditions are interrupt, standby, operation, and stop. Explained first are the high level and low level DMA operational relationships.

When high level DMA is operating, as shown in Figure 3.15, and launching low level DMA currently interrupted, the operation will not occur at the time when the low level DMA is launched (it will not be in operation). It will wait for a period of time and then go into operation mode. This period is called Standby (or Wait period), and this condition always exists prior to the DMA operation. Low level DMA operates after high level DMA is completed.

When starting high level DMA while low level DMA is operating, operation will not begin at the moment that high level DMA is started but will begin to operate after temporarily being on standby. At this time, low level DMA is interrupted and cannot start until high level DMA has stopped (operation ends).

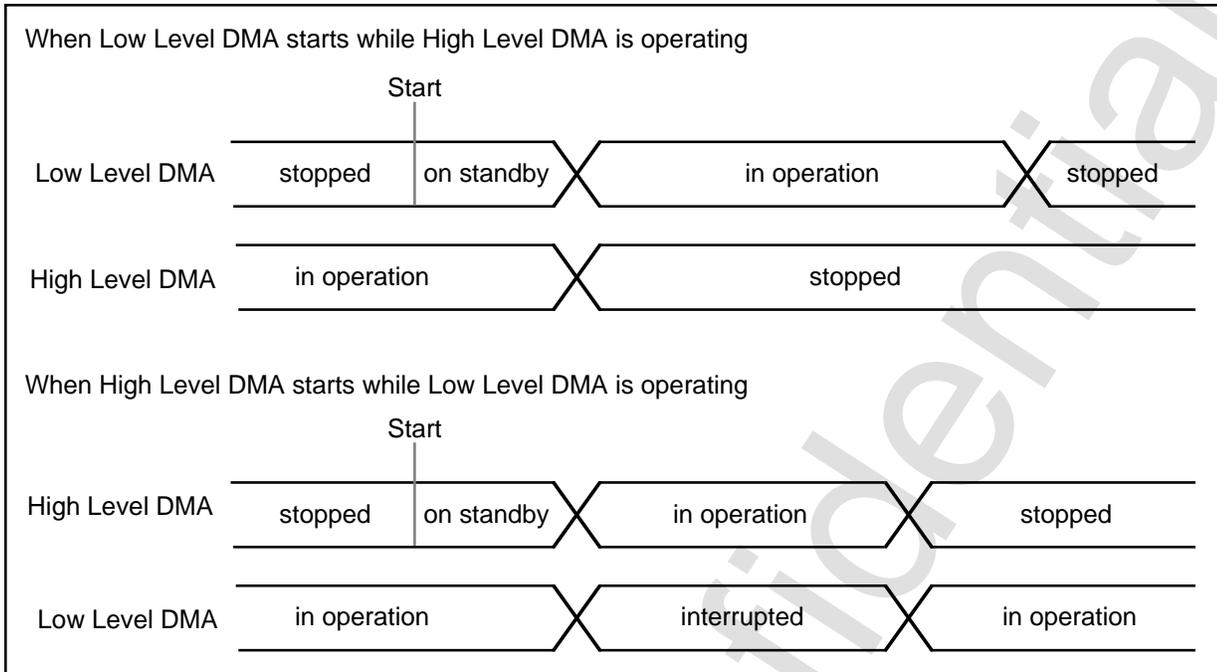


Figure 3.12 High Level DMA Operation

A 0 bit during interrupt or operation confirms that the DMA operation is stopped. Figure 3.13 shows access, interrupt, stand by, and operation registers.

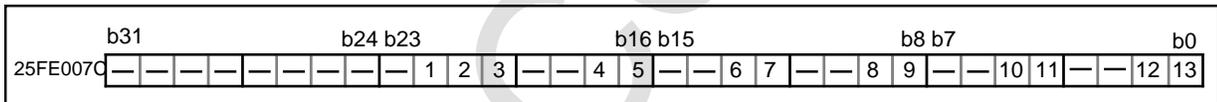


Figure 3.13 DMA Status Register (Register: DSTA) Initial Value 0000000H

DMA DSP Bus Access Flag (1 [bit 22] in Figure 3.13)

DACS D (R) DMA ACceSs DSP-Bus

Shows whether the DSP bus is being accessed during DMA. 1 means accessing. 0 means not accessing.

DMA B Bus Access Flag (2 [bit 21] in Figure 3.13)

DACS B (R) DMA ACceSs B-Bus

Shows whether the B bus is being accessed during DMA. 1 means accessing. 0 means not accessing.

DMA A Bus Access Flag (3 [bit 20] in Figure 3.13)

DACS A (R) DMA ACceSs A-Bus

Shows whether the A bus is being accessed during DMA. 1 means accessing. 0 means not accessing.



Level-1 DMA Interrupt Flag (4 [bit 17] in Figure 3.13)

D1BK (R) DMA level 1 Back ground flag

Shows Level-1 DMA transfer execution is interrupted by the effect of high level DMA. A 1 shows that it is currently being interrupted. A 0 shows that level 1 DMA is not interrupted.

Level-0 DMA Interrupt Flag (5 [bit 16] in Figure 3.13)

D0BK (R) DMA level 0 Back ground flag

Shows Level-0 DMA transfers execution is interrupted by the effect of high level DMA. A 1 shows that it is currently being interrupted. A 0 shows that level 0 DMA is not interrupted.

Level-2 DMA Stand by Flag (6 [bit 13] in Figure 3.13)

D2WT (R) DMA level 2 Wait flag

Level-2 DMA transfer execution is currently shown in on standby (in wait condition). A 1 shows the current standby condition. A 0 shows that level 2 DMA is not on standby.

Level-2 DMA Operation Flag (7 [bit 12] in Figure 3.13)

D2MV (R) DMA level 2 Move flag

Level-2 DMA transfer execution is currently shown in operation. A 1 shows that it is currently in operation. A 0 shows level 2 DMA is not in operation. Also, when both D2WT and D2MV are 0, it shows that level 2 DMA is stopped.

Level-1 DMA Stand by Flag (8 [bit 19] in Figure 3.13)

D1WT (R) DMA level 1 Wait flag

Level-1 DMA transfer execution is currently shown on standby. A 1 shows the current standby condition. A 0 shows that level 1 DMA is not on standby.

Level-1 DMA Operation Flag (9 [bit 8] in Figure 3.13)

D1MV (R) DMA level 1 Move flag

Level-1 DMA transfer execution is currently shown in operation. A 1 shows that it is currently in operation. A 0 shows level 1 DMA is not in operation. Also, when D1WT, D1MV, D1BK are all 0, it shows that level 1 DMA is stopped.

Level-0 DMA Stand by Flag (10 [bit 5] in Figure 3.13)

D0WT (R) DMA level 0 Wait flag

Level-0 DMA transfer execution is shown to be currently on standby. A 1 shows the current standby condition. A 0 shows level 0 DMA is not on standby.

Level-0 DMA Operation Flag (11 [bit 4] in Figure 3.13)

D0MV (R) DMA level 0 MoVe flag

Level-0 DMA transfer execution is shown to be currently in operation. A 1 shows that it is currently in operation. A 0 shows that level 0 DMA is not in operation. Also, when all D0WT, D0MV, D0BK are 0 it indicates that level 0 DMA is stopped.

DSP DMA Stand by Flag (12 [bit 1] in Figure 3.13)

DDWT (R) DMA DSP WaiT flag

DMA transfer execution of the DSP statement is shown to be currently on standby. A 1 shows the current standby condition. A 0 shows that DSP issue DMA is not on standby.

DSP DMA Operation Flag (13 [bit 0] in Figure 3.13)

DDMV (R) DMA DSP MoVe flag

DMA transfer execution of the DSP statement is shown to be currently in operation. A 1 shows that it is currently in operation. A 0 shows that DSP issue DMA is not in operation. Also, when DDWT, DDMV, D0BK are all 0, it shows that DSP DMA is stopped.



3.3 DSP Control Ports

DSP Program Control Port

The DSP program control port is shown in Figure 3.14.

	b31				b24 b23				b16 b15				b8 b7				b0													
25FE080	—	—	—	—	1	2	—	3	4	5	6	7	8	9	10	11	—	—	—	—	—	—	12	13	14	15	16	17	18	19

Figure 3.14 DSP Program Control Port (Register: PPAF) Initial Value 00000000H

Execute Pause Reset Flag (1 [bit 26] in Figure 3.14)

PR (W) execute Pause Reset flag

When the program execute control flag (see below) is 1, the program pause is reset if 1 is written to the flag and execution begins. The condition does not change when it does not pause or when the program execute flag is 0.

Execute Pause Flag (2 [bit 25] in Figure 3.14)

EP (W) Execute Pause flag

When the program execute control flag (see below) is 1, the executing program pauses if 1 is written to the flag. This condition does not change when it pauses or when the program execute flag is 0.

D0-Bus DMA Execution Flag (3 [bit 23] in Figure 3.14)

T0 (R) Transfer 0

This flag becomes 1 when executing DMA using the D0-Bus.

Sine Flag (4 [bit 22] in Figure 3.14)

S (R) Sign flag

This flag becomes 1 when the operation result is negative.

Zero Flag (5 [bit 21] in Figure 3.14)

Z (R) Zero flag

This flag becomes 1 when the operation result is 0.

Carry Flag (6 [bit 20] in Figure 3.14)

C (R) Carry flag

This flag becomes 1 when carry occurs in the operation result.

Overflow Flag (7 [bit 19] in Figure 3.14)

V (R) oVerflow flag

This flag becomes 1 when the operation results causes overflow (or underflow). This flag is reset by the read out.

Program End Interrupt Flag (8 [bit 18] in Figure 3.14)

E (R) End flag

This flag becomes 1 and causes program end interrupt to occur when the program ended by the ENDI command is detected. This flag is reset by the read out.

Step Execute Control Bit (9 [bit 17] in Figure 3.14)

ES (W) Execute Step control bit

The program executes 1 step if a 1 is written while the program is stopped (when the program execute control flag is 0). Invalid while executing.

Program Execute Control Flag (10 [bit 16] in Figure 3.14)

EX (R/W) Execute control flag

Controls execution of program. Execution begins by writing 1 and stops by writing 0. When this flag is read out, it can be determined whether execution is in progress (1) or is stopped (0).

Program Counter Transfer Enable Bit (11 [bit 15] in Figure 3.14)

LE (W) Load Enable bit

This bit decides whether or not the program RAM address (see below) is to be loaded to the program counter. The program RAM address is loaded to the program counter if 1 is written to the bit. The address can not be loaded when the program is being executed (when the program execute control flag is 1).

Program RAM Address (12~19 [bit 7~0] in Figure 3.14)

P7-0 (R/W) Program RAM address bit 7-0

Stores the address of the program RAM. Also, is able to set the begin address and read the stop address.



DSP Data RAM Data Port

Details of the DSP data RAM data port are shown in Figure 3.17. The data RAM data is accessed from this port. The data RAM address of the DSP data RAM address port increases by 1 when accessed. However, access is prohibited while the program is being executed (when program execute control flag is 1). This port can read and write.

	b31				b24 b23								b16 b15								b8 b7				b0							
25FE008C	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Figure 3.17 DSP Data RAM Data Port (Register: PDD) Initial Value Undefined



3.4 Timer Registers

Timer 0 Compare Register

The Timer 0 compare register is shown in Figure 3.18. (Timer 0 is a counter that increases on receiving an H-Blank-IN signal, and that is cleared by a V-Blank-END signal.)

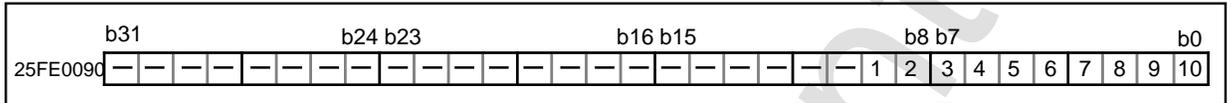


Figure 3.18 Timer 0 Compare Register (Register: T0C) Initial Value Undefined

Timer 0 Compare Data (1~10 [bit 9~0] in Figure 3.18)

T0C9-0 (W) Timer 0 Compare data bit 9-0

When the value of Timer 0 is equal to the value of this register, timer 0 interrupt will occur.

Timer 1 Set Data Register

The Timer 1 set data register is shown in Figure 3.19. (Timer 1 sets the data of this register by the H-Blank-IN signal receive, automatically counts down by 7 MHz, and when the Timer 1 value is 0, executes interrupt.)

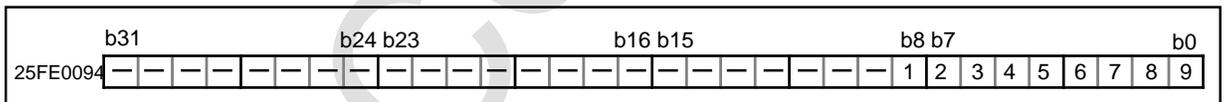


Figure 3.19 Timer 1 Set Data Register (Register: T1S) Initial Value Undefined

Timer 1 Set Data (1~9 [bit 8~0] in Figure 3.19)

T1S8-0 (W) Timer 1 Set data bit 8-0

Sets the value that is set in Timer 1.

Timer 1 Mode Register

Details of the Timer 1 mode register are shown in Figure 3.20. How occurrence of Time is set is decided by this register.

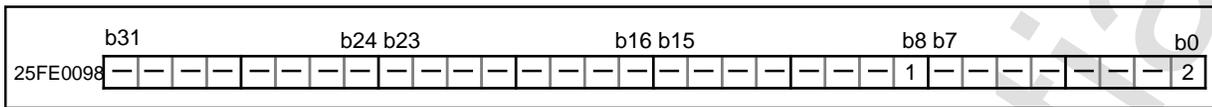


Figure 3.20 Timer 1 Mode Register (Register: T1MD) Initial Value 00000000H

Timer 1 Mode Bit (1 [bit 8] in Figure 3.20)

T1MD (W) Timer 1 MoDe bit

This bit specifies the occurrence of Timer 1. Table 3.6 shows what happens when it occurs.

Table 3.6 Timer 1 Occurrence Selection

T1MD	Occurrence Selection
0	Interrupt occurs at each line
1	Occurs only at lines indicated by Timer 0

Timer Enable Bit (2 [bit 0] in Figure 3.20)

TENB (W) Timer ENaBle bit

This bit turns the timer operation ON/OFF. Operation details are shown in Table 3.7.

Table 3.7 Timer Operation Contents

TENB	Timer Operation
0	Timer operation off
1	Timer operation on



DSP End Interrupt Mask Bit (10 [bit 5] in Figure 3.21)
IMS5 (W) Interrupt MaSk bit bit 5
 Indicates whether to mask the DSP end interrupt.

Timer 1 Interrupt Mask Bit (11 [bit 4] in Figure 3.21)
IMS4 (W) Interrupt MaSk bit bit 4
 Indicates whether to mask the Timer 1 interrupt.

Timer 0 Interrupt Mask Bit (12 [bit 3] in Figure 3.21)
IMS3 (W) Interrupt MaSk bit bit 3
 Indicates whether to mask the Timer 0 interrupt.

H-Blank-IN Interrupt Mask Bit (13 [bit 2] in Figure 3.21)
IMS2 (W) Interrupt MaSk bit bit 2
 Indicates whether to mask the H-Blank-IN interrupt.

V-Blank-OUT Interrupt Mask Bit (14 [bit 1] in Figure 3.21)
IMS1 (W) Interrupt MaSk bit bit 1
 Indicates whether to mask the V-Blank-OUT interrupt.

V-Blank-IN Interrupt Mask Bit (15 [bit 0] in Figure 3.21)
IMS0 (W) Interrupt MaSk bit bit 0
 Indicates whether to mask the V-Blank-IN interrupt.

Interrupt Status Register

Figure 3.22 shows the interrupt status register.

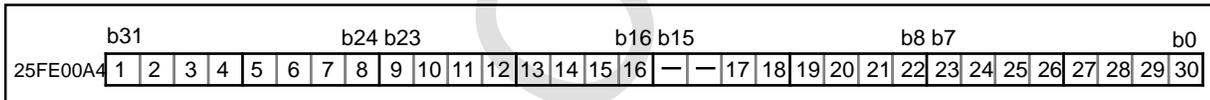


Figure 3.22 Interrupt Status Register (Register: IST) Initial Value 0000000H

These status registers are all read/write registers; the read and write meanings are as shown in Table 3.8.



Table 3.8 Interrupt Status Bit Contents

Access	Status	Result
Read	0	Interrupt does not occur
	1	Interrupt does occur
Write	0	Resets interrupt
	1	Maintains current interrupt status

External Interrupt Status Bit (1~16 [bit 31-16] in Figure 3.22)

IST31-16 (R/W) Interrupt Status bit bit 31-16

Shows the status of 16 external interrupts from external interrupt 15 (1 in Figure 3.25) to external interrupt 0 (16 in Figure 3.25).

Sprite Draw End Interrupt Status Bit (17 [bit 13] in Figure 3.22)

IST13 (R/W) Interrupt Status bit bit 13

Shows interrupt status of sprite draw end.

DMA Illegal Interrupt Status Bit (18 [bit 12] in Figure 3.22)

IST12 (R/W) Interrupt Status bit bit 12

Shows interrupt status of DMA illegal.

Level-0-DMA End Interrupt Status Bit (19 [bit 11] in Figure 3.22)

IST11 (R/W) Interrupt Status bit bit 11

Shows interrupt status of level-0-DMA end.

Level-1-DMA End Interrupt Status Bit (20 [bit 10] in Figure 3.22)

IST10 (R/W) Interrupt Status bit bit 10

Shows interrupt status of level-1-DMA end.

Level-2-DMA End Interrupt Status Bit (21 [bit 9] in Figure 3.22)

IST9 (R/W) Interrupt Status bit bit 9

Shows interrupt status of level-2-DMA end.

PAD Interrupt Status Bit (22 [bit 8] in Figure 3.22)

IST8 (R/W) Interrupt Status bit bit 8

Shows status of interrupt from PAD.

System Manager Interrupt Status Bit (23 [bit 7] in Figure 3.22)

IST7 (R/W) Interrupt Status register bit bit 7

Shows status of interrupt from the system manager.

Sound Request Interrupt Status Bit (24 [bit 6] in Figure 3.22)

IST6 (R/W) Interrupt Status bit bit 6

Shows status of sound request interrupt.

DSP End Interrupt Status Bit (25 [bit 5] in Figure 3.22)

IST5 (R/W) Interrupt Status bit bit 5
Shows status of DSP end interrupt.

Timer 1 Interrupt Status Bit (26 [bit 4] in Figure 3.22)

IST4 (R/W) Interrupt Status bit bit 4
Shows status of Timer 1 interrupt.

Timer 0 Interrupt Status Bit (27 [bit 3] in Figure 3.22)

IST3 (R/W) Interrupt Status bit bit 3
Shows status of Timer 0 interrupt.

H-Blank-IN Interrupt Status Bit (28 [bit 2] in Figure 3.22)

IST2 (R/W) Interrupt Status register bit bit 2
Shows status of H-Blank-IN interrupt.

V-Blank-OUT Interrupt Status Bit (29 [bit 1] in Figure 3.22)

IST1 (R/W) Interrupt Status bit bit 1
Shows status of V-Blank-OUT interrupt.

V-Blank-IN Interrupt Status Bit (30 [bit 0] in Figure 3.22)

IST0 (R/W) Interrupt Status bit bit 0
Shows status of V-Blank-IN interrupt.



3.6 A-Bus Control Registers

A-Bus Interrupt Acknowledge Register

Figure 3.23 shows the A-Bus interrupt acknowledge register.

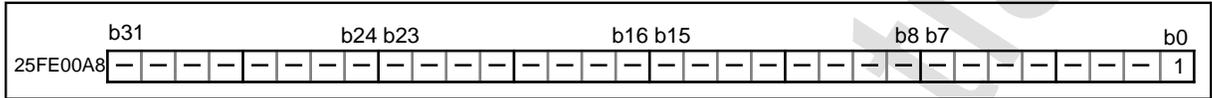


Figure 3.23 A-Bus Interrupt Acknowledge Register (Register: AIACK) Initial Value 00000000H

A-Bus Interrupt Acknowledge (1 [bit 0] in Figure 3.23)

AIACK (R/W) A-Bus Interrupt ACKnnowledge

This shows the effectiveness or ineffectiveness of interrupts from the devices that exist on the A-Bus. This bit can read and write. The meaning of the bit is shown in Table 3.9. If interrupt is requested, the A-Bus interrupt acknowledge cycle occurs, the interrupt classification data (16 bit) is fetched, and by means of its contents, the current interrupt condition can be acknowledged. If this cycle occurs, and since the AIACK bit must be 0 and the A-Bus interrupt becomes ineffective, the AIACK bit must be reset to receive interrupt from the A-Bus.

Table 3.9 A-Bus Interrupt Acknowledge Contents

Access	Status	Contents
Read	0	Invalid A-Bus interrupt
	1	Valid A-Bus interrupt
Write	0	Invalid A-Bus interrupt
	1	Valid A-Bus interrupt

A-Bus Set Register

There are a total of four types of spaces arranged as spaces connected to the A-Bus, chip select 0 ~ 2 (hereafter referred to as CS) which includes three types of spaces that are output and one type of dummy space that CS does not output.

The register relating to the A-Bus is determined by the connecting devices and therefore must be set to include all devices. Make sure that there is no excessive change in the value after it has been set.

CS0, CS1, and CS2 Dummy Space A-Bus Set Registers

Figure 3.24 shows the CS0 and CS1 spaces, and Figure 3.25 shows the CS2 spaces and dummy spaces of the A-Bus set register.

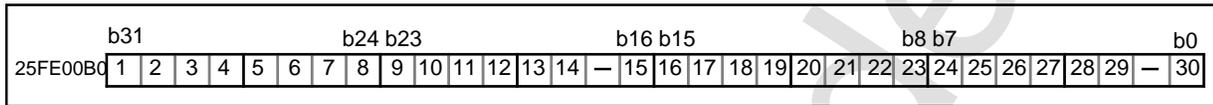


Figure 3.24 A-Bus Set Register [CS0, CS1 Spaces] (Register: ASR0) Initial Value 00000000H

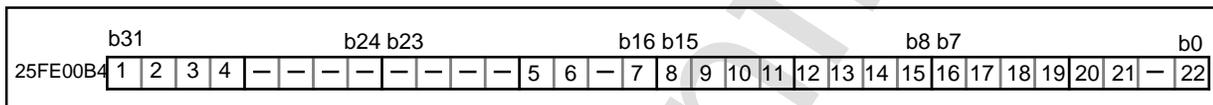


Figure 3.25 A-Bus Set Register [CS2, Dummy Spaces] (Register: ASR1) Initial Value 00000000H

CS0 Space Previous Read Bit (1 [bit 31] in Figure 3.24)

A0PRD (W) A-Bus CS0 Previous Read bit

This bit decides whether the data previous read process of CS0 space is effective or not. The time period from when access begins until data output is reduced by the previous data read process. This is effective only for data that is stored in the address following the accessed data; other addresses do not change with normal access. A 1 shows it is effective, 0 shows it is not effective. Figure 3.26 shows the result when the previous read is effective.



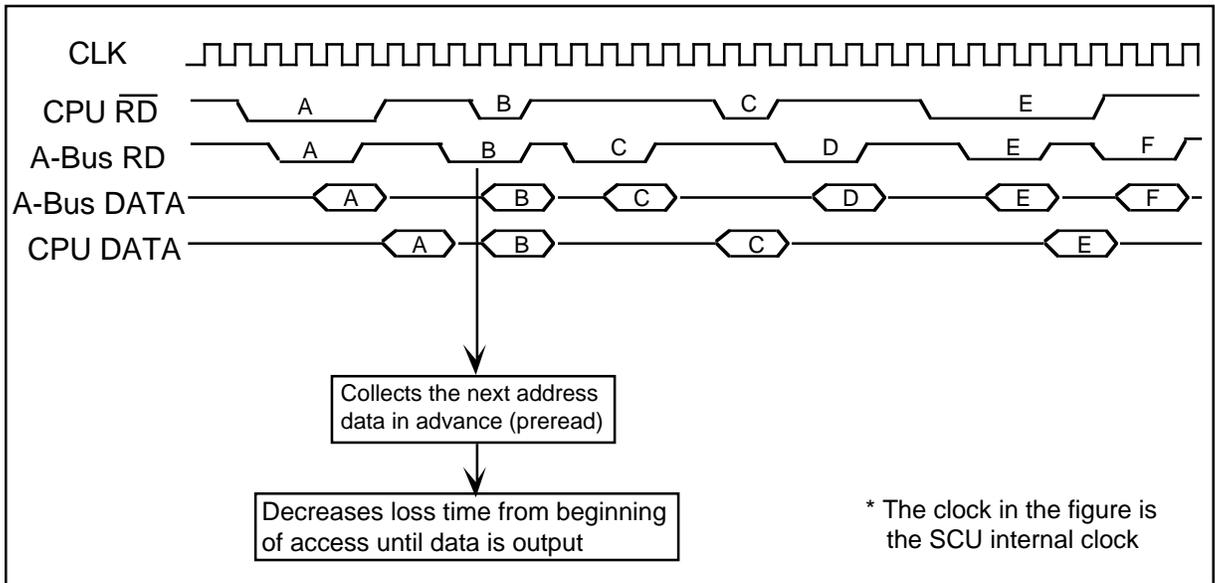


Figure 3.26 Result of Previous Read Process

Pre-charge Insert Bit After CS0 Space Write (2 [bit 30] in Figure 3.24)

A0WPC (W) A-Bus CS0 after Write Pre-Charge insert bit

After data is written in the CS0 space, 1 clock no-process condition can be inserted. This is the bit that decides whether the process is effective or ineffective: 1 shows it is effective; 0 shows it is ineffective. This bit does not affect the operation after CS0 space read. The operation when this bit has been set is shown in Figure 3.27.

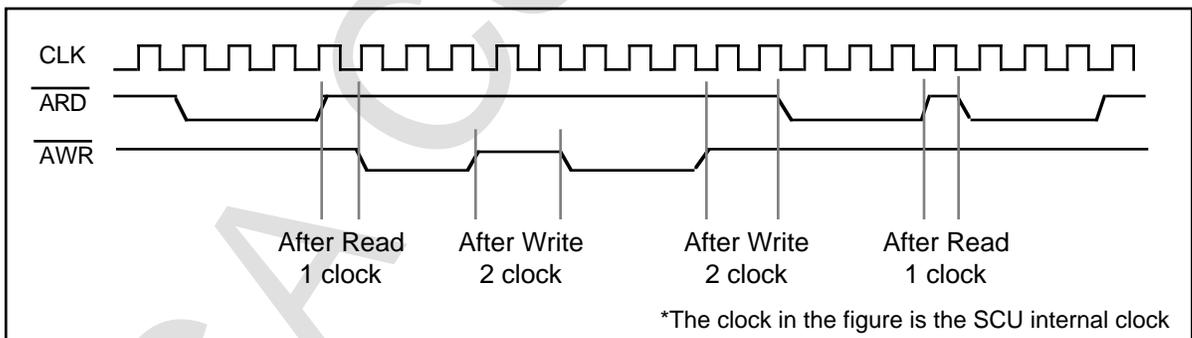


Figure 3.27 Timing when Setting the Pre-Charge Insert Bit after Write

Pre-charge Insert Bit After CS0 Space Read (3 [bit 29] in Figure 3.24)

A0RPC (W) A-Bus CS0 Previous Read bit

After CS0 space data is read, 1 clock no-process condition can be inserted. This is the bit that decides whether the process is effective or ineffective: 1 shows it is effective; 0 shows it is ineffective. This bit does not affect the operation after CS0 space write. The operation when this bit has been set is shown in Figure 3.28. Depending on the type of device, this bit is set because a fixed period is required after CS is set to High until the next CS is set to Low. This is true for write as well.

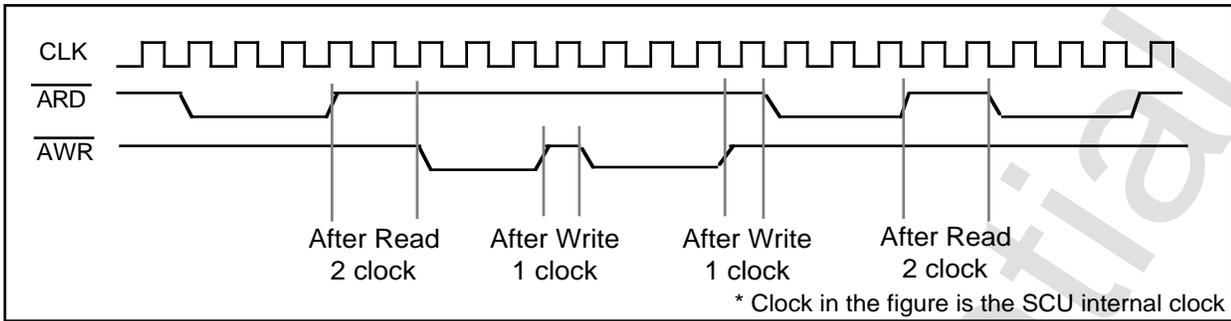


Figure 3.28 Timing when Setting the Pre-Charge Insert Bit after Read

CS0 External Wait effective Bit (4 [bit 28] in Figure 3.24)

A0EWT (W) A-Bus CS0 External Wait effective bit

Wait can be inserted by force by the external signal when accessing the CS0 space via the A-Bus. Whether the process will be effective or not is decided by this bit. A 1 shows that the process is effective, 0 shows that the process is ineffective. When the process is effective, wait will continue as long as the external wait signal is "Low" at the time of the SCU wait sampling. Figure 3.29 shows the difference in timing charts when external wait is effective or ineffective.

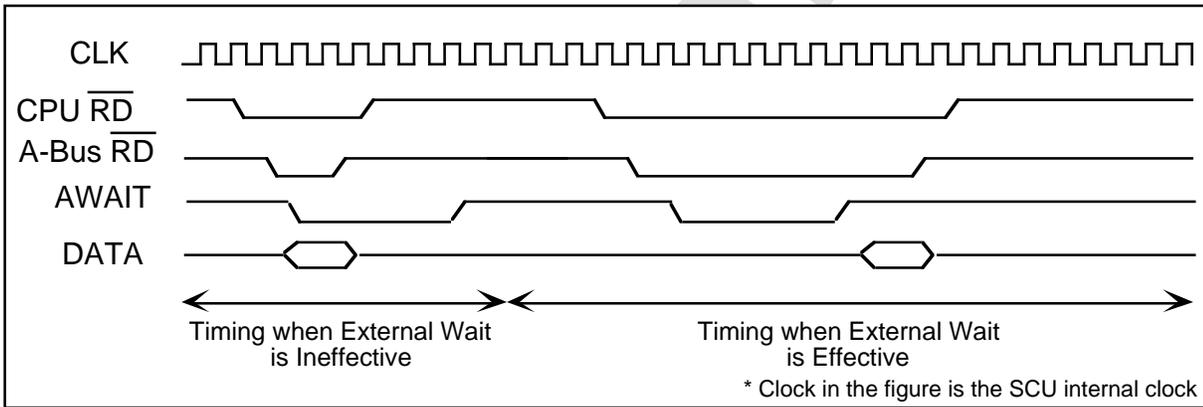


Figure 3.29 Differences in Timing by Setting External Wait Effective Bit

CS0 Space Burst Cycle Wait Number Set Bit (5~8 [bit 27~24] in Figure 3.24)

A0BW3-0 (W) A-Bus CS0 Burst cycle Wait bit 3-0

In the CS0 space, the wait number is set for 1 cycle while a burst access is being performed. Table 3.10 shows the set values.



Table 3.10 CS0 Space Burst Cycle Set Values

Bit				Wait Number
A0BW3	A0BW2	A0BW1	A0BW0	
0	0	0	0	No wait (wait does not sample)
0	0	0	1	1-cycle wait
:	:	:	:	
1	1	1	0	14-cycle wait
1	1	1	1	15-cycle wait

CS0 Normal Cycle Wait Number Set Bit (9~12 [bit 23~20] in Figure 3.24)

A0NW3-0 (W) A-Bus CS0 Normal cycle Wait bit 3-0

In the CS0 space, the wait number is set for 1 cycle during normal access.

Table 3.11 shows the set values.

Table 3.11 CS0 Space Normal Cycle Set Values

Bit				Wait Number
A0NW3	A0NW2	A0NW1	A0NW0	
0	0	0	0	No wait (does not sample waits)
0	0	0	1	1 cycle wait
:	:	:	:	
1	1	1	0	14 cycle wait
1	1	1	1	15 cycle wait

CS0 Burst Length Set Bit (13~14 [bit 19~18] in Figure 3.24)

A0LN1-0 (W) A-Bus CS0 burst Length bit 1-0

In the CS0 space, the length (boundary) to be accessed is designated during burst access. Table 3.12 shows the length set values.

Table 3.12 CS0 Space Burst Length Set Values

Bit		Access Values
A0LN1	A0LN0	
0	0	No burst access
0	1	4 address burst access
1	0	256 address burst access
1	1	No boundary

CS0 Space Bus Size Set Bit (15 [bit 16] in Figure 3.24)

A0SZ (W) A-Bus CS0 bus Size bit

Sets the A-Bus size in the CS0 space. Table 3.13 shows the set values.

Table 3.13 CS0 Space Bus Set Values

A0SZ	Bus Size Settings
0	Indicates 16 bit bus
1	Indicates 8 bit bus

CS1 Space Previous Read Effective Bit (16 [bit 15] in Figure 3.24)

A1PRD (W) A-Bus CS1 Previous Read bit

This bit decides whether the data previous read process of CS1 space is effective or not. The data previous read processes reduces the time from access start until data output. This is effective only for data that is stored in address that follows the accessed data. Other addresses do not change with normal addresses. A 1 shows it is effective, a 0 shows it is not effective. See Figure 3.26 for the result when previous read is effective.

Pre-charge Insert Bit After CS1 Space Write (17 [bit 14] in Figure 3.24)

A1WPC (W) A-Bus CS1 after Write Pre-Charge insert bit

Non-process conditions of 1 clock can be inserted after writing data to CS1 space. This is the bit that decides whether the process is effective or ineffective. A 1 shows it is effective, a 0 shows it is ineffective. This bit has no effect on the operation after read. Figure 3.26 shows the operation when this bit has been set.

Pre-charge Insert Bit After CS1 Space Read (18 [bit 13] in Figure 3.24)

A1RPC (W) A-Bus CS1 Read Pre-Charge insert bit

One clock worth of non-process condition can be inserted after reading data to CS1 space. This is the bit that decides whether the process is effective or ineffective. A 1 shows it is effective, a 0 shows it is ineffective. This bit has no effect on the operation after write. Figure 3.28 shows the operation when this bit has been set.



CS1 Space External Wait Effective Bit (19 [bit 12] in Figure 3.24)

A1EWT (W) A-Bus CS1 External Wait effective bit

Wait can be entered by force by an external signal when accessing the CS1 space via the A-Bus; however, whether the process will be effective or not is decided by this bit. A 1 shows that the process is effective, a 0 shows that the process is ineffective. When the process is effective, wait will continue as long as the external signal is "Low." Figure 3.29 shows differences in timing charts when external wait is effective vs. ineffective.

CS1 space Burst Cycle Wait Number Set Bit (20~23 [bit 11~8] in Figure 3.24)

A1BW3-0 (W) A-Bus CS1 Burst cycle Wait bit 3-0

In the CS1 space, the wait number is set for 1 cycle while a burst access is performed. Table 3.14 shows the set values.

Table 3.14 CS1 Space Burst Cycle Set Values

Bit				Wait Number
A1BW3	A1BW2	A1BW1	A1BW0	
0	0	0	0	No wait (Does not sample wait)
0	0	0	1	1 cycle wait
:	:	:	:	
1	1	1	0	14 cycle wait
1	1	1	1	15 cycle wait

CS1 Normal Cycle Wait Number Set Bit (24~27 [bit 7~4] in Figure 3.24)

A1NW3-0 (W) A-Bus CS1 Normal cycle Wait bit 3-0

In the CS1 space, the wait number is set for 1 cycle during a normal access. Table 3.15 shows the set values.

Table 3.15 CS1 Space Normal Cycle Set Values

Bit				Wait Number
A1NW3	A1NW2	A1NW1	A1NW0	
0	0	0	0	No wait (Does not sample wait)
0	0	0	1	1 cycle wait
:	:	:	:	
1	1	1	0	14 cycle wait
1	1	1	1	15 cycle wait

CS1 space Burst Length Bit (28~29 [bit 3~2] in Figure 3.24)

A1LN1-0 (W) A-Bus CS1 burst Length bit 1-0

The access length (boundary) is indicated while burst accessing in CS1 space. Table 3.16 shows length values.

Table 3.16 CS1 Space Burst Length Set Values

Bit		Access Settings
A1LN1	A1LN0	
0	0	No burst access
0	1	4 Address burst access
1	0	256 Address burst access
1	1	No boundary

CS1 space Bus Size Set Bit (30 [bit 0] in Figure 3.24)

A1SZ (W) A-Bus CS1 bus Size bit

Sets the A-Bus bus size in the CS1 space. Table 3.17 shows the set values.

Table 3.17 CS1 Space Bus Size Set Values

A1SZ	Bus Size Settings
0	Indicates 16-bit bus
1	Indicates 8-bit bus

CS2 Space Previous Read Effective Bit (1 [bit 31] in Figure 3.25)

A2PRD (W) A-Bus CS2 Previous Read bit

This bit decides whether the data in the previous read process of CS2 is effective or not. The data previous read process reduces the time from access start until data output. This is effective only for data that is stored in the address that follows the accessed data. Other addresses do not change with normal addresses. A 1 shows it is effective, a 0 shows it is not effective. See Figure 3.25 for the effect when previous read is effective.



Pre-charge Insert Bit After Writing CS2 Space (2 [bit 30] in Figure 3.25)

A2WPC (W) A-Bus CS2 after Write Pre-Charge insert bit

A no-process condition of 1 clock can be inserted after writing data to CS2. This is the bit that decides whether the process is effective or ineffective. A 1 shows it is effective, a 0 shows it is ineffective. This bit has no effect on the operation after read. Figure 3.27 shows the operation when this bit has been set.

Pre-charge Insert Bit After Reading CS2 Space (3 [bit 29] in Figure 3.25)

A2RPC (W) A-Bus CS2 Read Pre-Charge insert bit

A no-process condition of 1 clock can be inserted after reading data to CS2. This is the bit that decides whether the process is effective or ineffective. A 1 shows it is effective, a 0 shows it is ineffective. This bit does not affect the operation after write. Figure 3.28 shows the operation when this bit has been set.

CS2 Space External Wait Effective Bit (4 [bit 28] in Figure 3.25)

A2EWT (W) A-Bus CS2 External Wait effective bit

Wait can be entered by force by an external signal when accessing the CS2 space via the A-Bus. Whether the process will be effective or not is decided by this bit. A 1 shows that the process is effective, a 0 shows that the process is ineffective. When the process is effective, wait will continue as long as the external signal is "Low." Figure 3.29 shows differences in timing charts when external wait is effective vs. ineffective.

CS2 Space Burst Length Bit (5~6 [bit 19~18] in Figure 3.25)

A2LN1-0 (W) A-Bus CS2 burst Length bit 1-0

The access length (boundary) is indicated while burst accessing in CS2. Table 3.18 shows the length settings.

Table 3.18 CS2 Space Burst Length Set Values

Bit		Access Settings
A2LN1	A2LN0	
0	0	No burst access
0	1	4 Address burst access
1	0	256 Address burst access
1	1	No border

CS2 Bus Size Set Bit (7 [bit 16] in Figure 3.25)

A2SZ (W) A-Bus CS2 bus Size bit

Sets the A-Bus bus size in the CS2 space. Table 3.19 shows the set values.

Table 3.19 CS2 Space Bus Size Set Values

A2SZ	Bus Size Settings
0	Indicates 16-bit bus
1	Indicates 8-bit bus

Dummy Space Previous Read Effective Bit (8 [bit 15] in Figure 3.25)**A3PRD (W) A-Bus CS3 Previous Read bit**

This bit decides whether the data previous read process of dummy space is effective or not. The data previous read process reduces the time from access start until data output. This is effective only for data that is stored in address that follows the accessed data. Other addresses do not change with normal addresses. A 1 shows it is effective, a 0 shows it is not effective. See Figure 3.26 for the result when previous read is effective.

After Pre-charge Insert Bit Dummy Space Write (9 [bit 14] in Figure 3.25)**A3WPC (W) A-Bus CS3 after Write Pre-Charge insert bit**

Non-process conditions of 1 clock can be inserted after writing data to dummy space. This is the bit that decides whether the process is effective or ineffective. A 1 shows it is effective, a 0 shows it is ineffective. This bit has no effect on the operation after read. Figure 3.27 shows the operation when this bit has been set.

After Pre-charge Insert Bit Dummy Space Read (10 [bit 13] in Figure 3.25)**A3RPC (W) A-Bus CS3 Read Pre-Charge insert bit**

Non-process conditions of 1 clock can be inserted after reading data to dummy space. This is the bit that decides whether the process is effective or ineffective. A 1 shows it is effective, a 0 shows it is ineffective. This bit does not affect the operation after write. Figure 3.28 shows the operation when this bit has been set.

Dummy Space External Wait Effective Bit (11 [bit 12] in Figure 3.25)**A3EWT (W) A-Bus CS3 External Wait effective bit**

Wait can be entered by force by an external signal when accessing the dummy space via the A-Bus. Whether the process will be effective or not is decided by this bit. A 1 shows that the process is effective, a 0 shows that the process is ineffective. When the process is effective, wait will continue as long as the external signal is "Low." Figure 3.29 shows differences in timing charts for when external wait is effective vs. when it is ineffective.

Dummy Space Burst Cycle Wait Number Set Bit (12~15 [bit 11~8] in Figure 3.25)**A3BW3-0 (W) A-Bus CS3 Burst cycle Wait bit 3-0**

In dummy space, the wait number is set for 1 cycle while a burst access is performed. Table 3.20 shows the set values.



Table 3.20 Dummy Space Burst Cycle Set Values

Bit				Wait Number
A3BW3	A3BW2	A3BW1	A3BW0	
0	0	0	0	No wait (wait not sampled)
0	0	0	1	1 cycle wait
:	:	:	:	
1	1	1	0	14 cycle wait
1	1	1	1	15 cycle wait

Dummy Space Normal Cycle Wait Number Bit (16~19 [bit 7~4] in Figure 3.25)

A3NW3-0 (W) A-Bus CS 3 Normal cycle Wait bit 3-0

In the dummy space, the wait number is set for 1 cycle during normal accessing. Table 3.21 shows the set values.

Table 3.21 Dummy Space Normal Cycle Set Values

Bit				Wait Number
A3NW3	A3NW2	A3NW1	A3NW0	
0	0	0	0	No wait (wait not sampled)
0	0	0	1	1 cycle wait
:	:	:	:	
1	1	1	0	14 cycle wait
1	1	1	1	15 cycle wait

Dummy Space Burst Length Set Bit (20~21 [bit 3~2] in Figure 3.25)

A3LN1-0 (W) A-Bus CS 3 burst Length bit 1-0

In the dummy space, the length (boundary) to be accessed is designated during burst access. Table 3.22 shows the length set values.

Table 3.22 Dummy Space Burst Length Set Values

Bit		Access Settings
A3LN1	A3LN0	
0	0	No burst access
0	1	4 address burst access
1	0	256 address burst access
1	1	No boundary

Dummy Space Bus Size Set Bit (22 [bit 0] in Figure 3.25)

A3SZ (W) A-Bus CS3 bus SiZe bit

Sets the A-Bus bus size in the dummy space. Table 3.23 shows the set values.

Table 3.23 Dummy Space Bus Size Set Values

A3SZ	Bus Size Settings
0	Indicates 16 bit bus
1	Indicates 8 bit bus

A-Bus Refresh Register

Figure 3.30 shows the A-Bus refresh register.

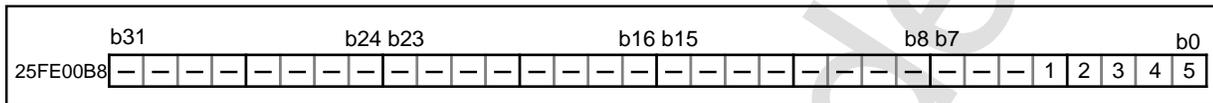


Figure 3.30 A-Bus Refresh Register (Register: AREF) Initial Value 00000000H

A-Bus Refresh Output Effective Bit (1 [bit 4] in Figure 3.30)

ARFEN (W) A-Bus ReFresh ENable bit

Makes effective the refresh cycle output of A-Bus. A 1 indicates it is effective, a 0 indicates it is not effective.

A-Bus Refresh Wait Number Set Bit (2~5 [bit 3~0] in Figure 3.30)

ARWT3-0 (W) A-Bus ReFresh WaiT bit 3-0

Sets the A-Bus refresh cycle wait number. Table 3.24 shows the details.

Table 3.24 A-Bus Refresh Wait Number

Bit				Wait Number
ARWT3	ARWT2	ARWT1	ARWT0	
0	0	0	0	No wait
0	0	0	1	1 cycle wait
:	:	:	:	
1	1	1	0	14 cycle wait
1	1	1	1	15 cycle wait



(This page is blank in the original Japanese document.)

SEGA Confidential



CHAPTER 4 DSP CONTROL

Chapter 4 Contents

4.1	DSP Internal BLOCK MAP	76
4.2	List of Commands	80
4.3	Operand Execution Methods	85
	Jump Command Execution	85
	Loop Program Execution	86
	DMA Command Execution	87
	End Command Execution	88
4.4	Special Process Execution	89
	Loading a Program by the DMA Command	89
	Repeating One Command	89
	Executing a SubRoutine Program	90
4.5	More About Commands	91
	Operation Commands	91
	Load Immediate Command	120
	DMA Command	132
	Jump Commands	141
	Loop Bottom Commands	153
	END Command	156

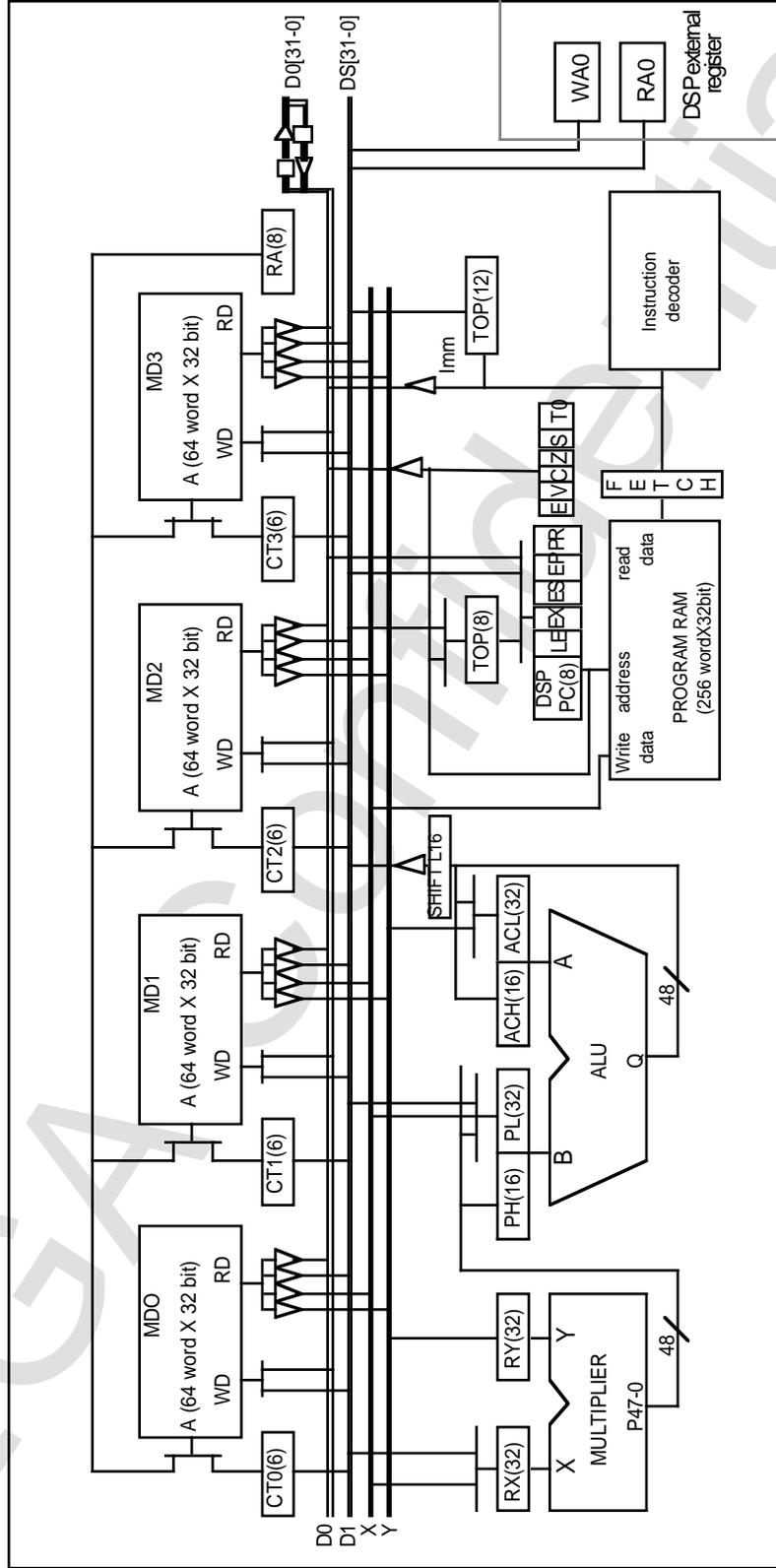
4.1 DSP Internal BLOCK MAP

Figure 4.1 (on the next page) is an internal block map of the DSP.

SEGA Confidential



Figure 4.1 Block Map Inside DSP



- ALU This arithmetic unit is able to output up to 48 bits. Normal calculations are executed at 32 bits. Only product sum operations become 48-bit operations.
- MULTIPLIER This multiplier outputs a low-order 48 bit from among the 64 bit results obtained by 32 bit X 32 bit. The calculation results are in 48 bit data; the high-order 16 bit is stored in PH and the low order 32 bit is stored in PL (see below).
- TOP (W) This is an 8 bit register that stores the lead address. The jump command and subroutine execution process store the lead address in this register and execute the process.
- LOP (W) This is a 12 bit register that stores the loop counter. The number of loops is set by the process of repeating 1 command.
- CT0-3 (W) This is a 6 bit register that stores the access address of data RAM0-3.
- MDO-3 (R/W) This is a 32 bit unit data port that stores the data of data RAM0-3. There are 64 data ports in each data RAM.
- RA (W) This is the address that stores the register for accessing the data RAM. This register is 8 bit. The RAM designation number (0-3) is stored by a high-order 2 bit. The RAM access address is stored by a low-order 6 bit.
- RX (W) This is the 32 bit X-bus connection register that stores the multiplier input data.
- RY (W) This is the 32 bit Y-bus connection register that stores the multiplier input data.
- PH (W) This register stores the high-order 16 bit within the 48 bit multiplier output data. There is also an input data storage register that stores the high-order 16 bit within ALU arithmetic unit input data B (48bit).
- PL (W) This register stores the low-order 32 bit within the 48 bit of multiplier output data. There is also an input data storage register that stores the low-order 32 bit within ALU arithmetic unit input data B (48bit).



- ACH (W) This register stores the high-order 16 bit within 48 bit data showing the ALU calculation results. There is also an input data storage register that stores the high-order 16 bit within ALU arithmetic unit input data A(48bit).
- ACL (W) This register stores the low-order 32 bit within the 48 bit data showing the ALU calculation results. There is also an input data storage register that stores the low-order 32 bit within ALU arithmetic unit input data A(48bit).
- D0 Bus This is a 32 bit data bus for external access. It operates at 28 MHz. It is used in accessing the main CPU.
- X-Bus, Y-Bus This is a 32 bit data bus for acquiring arithmetic unit input data. It operates at 14 MHz.
- RAO (W) This is a 32 bit external address register used in external → DSP DMA transfer. Since it takes a 4 byte unit value, the external address should be shifted right 2 bits.
- WAO (W) This is a 32 bit external address register used in DSP → external DMA transfer. Since it takes a 4 byte unit value, the external address should be shifted right 2 bits.

4.2 List of Commands

A list of commands used by DSP is given in Tables 4.1 to 4.4.

Table 4.1 List of Commands (1)

Type	Command	Overview of Operation
Operation Commands		
ALU Control	NOP	No operation
	AND	Takes the AND operation of [ACL] and [PL].
	OR	Takes the OR operation of [ACL] and [PL].
	XOR	Takes the exclusive OR of [ACL] and [PL].
	ADD	Adds [ACL] and [PL].
	SUB	Subtracts [PL] from [ACL].
	AD2	Adds [ACH][ACL] and [PH][PL].
	SR	Shifts [ACL] right 1 bit, stores LSB in carry flag
	RR	Rotates [ACL] right 1 bit, stores LSB in carry flag
	SL	Shifts [ACL] left 1 bit, stores 0 in LSB of [ACL], stores MSB in carry flag.
	RL	Rotates [ACL] left 1 bit, stores MSB in carry flag.
	RL8	Rotates [ACL] left 8 bits, stores b24 in carry flag.
	X-Bus Control	NOP
MOV [s], X		Transfers data from data RAM to [RX]
MOV MUL, P		[MULTIPLIER] data is transferred to [PH] [PL]
MOV [s], P		Transfers data from data RAM to [PL]
Y-Bus Control	NOP	No operation
	MOV [s], Y	Transfers data from data RAM to [RY]
	CLR A	Clears to 0 [ACH] and [ACL]
	MOV ALU, A	Transfers [ALU] data to [ACH][ACL]
	MOV [s], A	Transfers data from data RAM to [ACL]
D1-Bus Control	NOP	No operation
	MOV SImm, [d]	SImm (short immediate) data is stored in a register or a data RAM designated by [d].
	MOV [s], [d]	Data is transferred to the RAM designated by [s] or data RAM designated by [d] from the register.
Load Immediate Commands	MVI Imm, [d]	Stores Imm (immediate) data in register or in data RAM designated by [d]
	MVI Imm, [d], Z	When Z (zero flag) of the program control port is 1, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm, [d], NZ	When Z (zero flag) of the program control port is 0, Imm (immediate) data is stored in register or in data RAM designated by [d]



Table 4.2 List of Commands (2)

Type	Command	Overview of Operation
Load Immediate commands	MVI Imm , [d] , S	When S (sine flag) of the program control port is 1, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm , [d] , NS	When S (sine flag) of the program control port is 0, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm , [d] , C	When C (carry flag) of the program control port is 1, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm , [d] , NC	When C (carry flag) of the program control port is 0, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm , [d] , T0	When T0 (flag while executing D0 bus DMA) of the program control port is 1, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm , [d] , NT0	When T0 (flag while executing D0 bus DMA) of the program control port is 0, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm , [d] , ZS	When either S (sine flag) or Z (zero flag) of the program control port is 1, Imm (immediate) data is stored in register or in data RAM designated by [d]
	MVI Imm , [d] , NZS	When both S (sine flag) and Z (zero flag) of the program control port are 0, Imm (immediate) data is stored in register or in data RAM designated by [d]
DMA Commands	DMA D0, [RAM], SImm	SImm (short immediate) data is set in the transfer word number counter ([TN0]) as the transfer counter, and transfers data to the RAM area designated by [RAM] from outside using D0-Bus. Transfer begin address ([RA0]) and transfer word number counter ([TN0]) are updated to the value when transfer ends.
	DMA [RAM], D0, SImm	SImm (short immediate) data is set in the transfer word number counter ([TN0]) as the transfer counter, and transfers data from the RAM area designated by [RAM] using D0-Bus to the outside. Transfer begin address ([WA0]) and transfer word number counter ([TN0]) are updated to the value when transfer ends.

Table 4.3 List of Commands (3)

Type	Command	Overview of Operation
DMA Commands	DMA D0, [RAM], [s]	Sets data within the data RAM designated by [s] as the transfer counter to the transfer word number counter ([TN0]), and transfers data to the RAM area designated by [RAM] from outside using D0-Bus. Transfer begin address ([RA0]) and transfer word number counter ([TN0]) are updated to the value when transfer ends.
	DMA [RAM], D0, [s]	Sets data within the data RAM designated by [s] as the transfer counter to the transfer word number counter ([TN0]), and transfers data to the outside from the RAM area designated by [RAM] using D0-Bus. Transfer begin address ([WA0]) and transfer word number counter ([TN0]) are updated to the value at the time that transfer ends.
	DMAH D0, [RAM], SImm	SImm (short immediate) data is set in the transfer word number counter ([TN0]) as the transfer counter, and transfers data to the RAM area designated by [RAM] from outside using D0-Bus. Transfer begin address ([RA0]) and transfer word number counter ([TN0]) keep the value when transfer begins.
	DMAH [RAM], D0, SImm	SImm (short immediate) data is set as the transfer counter in the transfer word number counter ([TN0]), and transfers data from the RAM area designated by [RAM] to the outside using D0-Bus. Transfer begin address ([WA0]) and transfer word number counter ([TN0]) keep the value at the time that transfer ends.
	DMAH D0, [RAM], [s]	Sets data within the data RAM designated by [s] as the transfer counter to the transfer word number counter ([TN0]), and transfers data to the RAM area designated by [RAM] from outside using D0-Bus. Transfer begin address ([RA0]) and transfer word number counter ([TN0]) keep the value at the time that transfer begins.
	DMAH [RAM], D0, [s]	Sets data within the data RAM designated by [s] as the transfer counter to the transfer word number counter ([TN0]), and transfers data to the outside from the RAM area designated by [RAM] using D0-Bus. Transfer begin address ([WA0]) and transfer word number counter ([TN0]) keep the value at the time that transfer begins.
JUMP Commands	JMP Imm	Moves to the address shown by Imm (immediate)
	JMP Z, Imm	Moves to the address shown by Imm (immediate) when the Z (zero flag) of the program control port is 1.
	JMP NZ, Imm	Moves to the address shown by Imm (immediate) when the Z (zero flag) of the program control port is 0.



Table 4.4 List of Commands (4)

Type	Command	Overview of Processing
JUMP Commands	JMP S, Imm	When S (sine flag) of the program control port is 1, moves to address displayed by Imm (immediate)
	JMP NS, Imm	When S (sine flag) of the program control port is 0, moves to address displayed by Imm (immediate)
	JMP C, Imm	When C (carry flag) of the program control port is 1, moves to address displayed by Imm (immediate)
	JMP NC, Imm	When C (carry flag) of the program control port is 0, moves to address displayed by Imm (immediate)
	JMP T0, Imm	When T0 (flag while executing D0 Bus DMA) of the program control port is 1, moves to address displayed by Imm (immediate)
	JMP NT0, Imm	When T0 (flag while executing D0 Bus DMA) of the program control port is 0, moves to address displayed by Imm (immediate)
	JMP ZS, Imm	When either Z (zero flag) or S (sine flag) of the program control port is 1, moves to address displayed by Imm (immediate)
	JMP NZS, Imm	When either Z (zero flag) or S (sine flag) of the program control port is 0, moves to address displayed by Imm (immediate)
LOOP BOTTOM Commands	BTM	When loop counter ([LOP]) is any number but 0, the top address register ([TOP]) is stored in the program counter and the loop counter ([LOP]) is decremented. No operation is done when 0.
	LPS	When loop counter ([LOP]) is any number but 0, the program counter stops, the next command is executed, loop counter ([LOP]) is decremented. This is repeated until the loop counter is 0.
END Commands	END	Program stops and EX (program execute control flag) of the program control port is reset.
	ENDI	Program stops and EX (program execute control flag) of the program control port is reset, and E (program end interrupt flag) is set.

- Description of Constants
Follow the notation in Table 4.5.

Table 4.5 Descriptions of Constants

Notation	Description	Example
Binary	Place a "%" before numbers	%0010, %1111
Digital	Place nothing before nor after numbers	2, 10, 16, 32
Hexadecimal	Place a "\$" before numbers	\$05, \$0A, \$FF



4.3 Operand Execution Method

DSP controls and executes registers as shown for the following commands.

Jump Command Execution

Jump command execution is attained by storing the jump destination address (Immediate Data) in the program RAM address of the program control port. But you should be aware that commands that are pre-fetched will be executed. The conditional JUMP command examines the condition of the program control port flag, and then, if the conditions are met, stores the jump destination address in the program RAM address of the program control port. See the section on Jump commands under 4.5 "Commands" for the command format. Figure 4.2 is a flowchart of the Jump command execution.

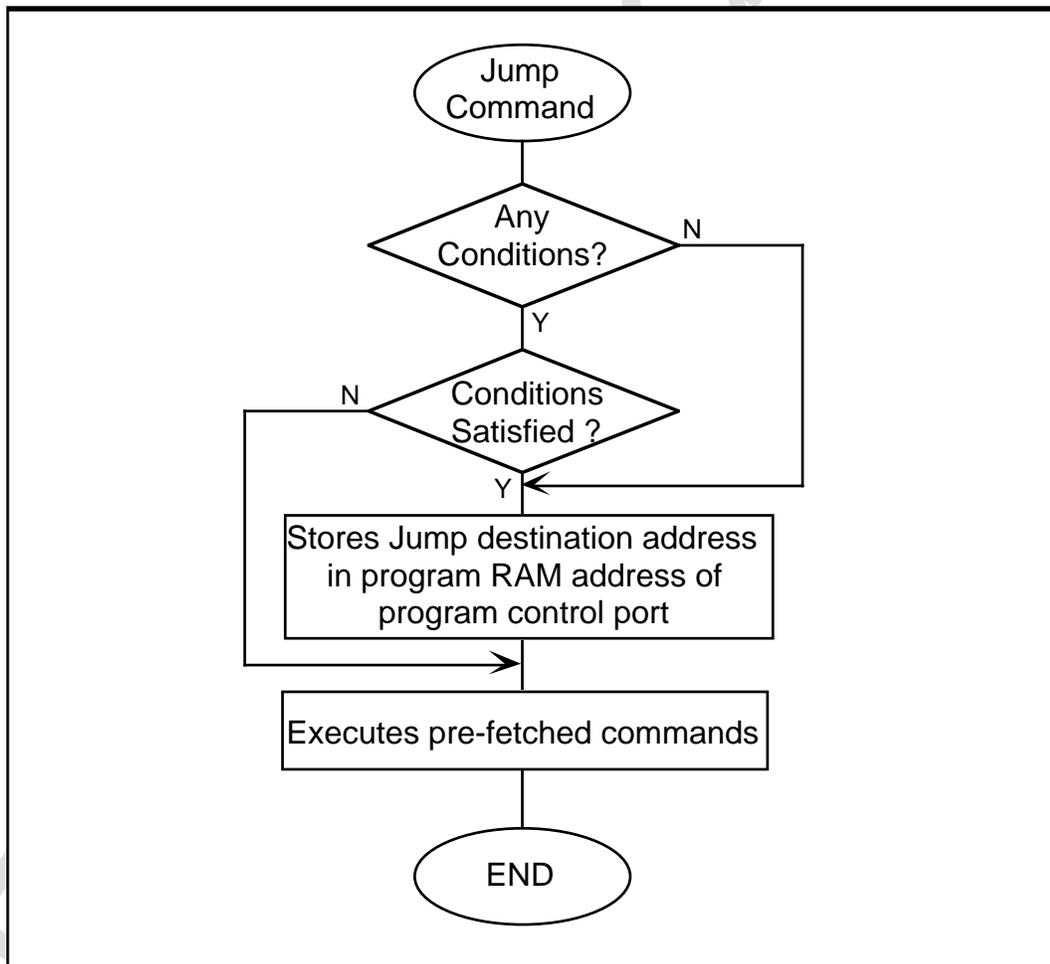


Figure 4.2 Jump Command Execution

Loop Program Execution

Execution of programs between the address designated by the top address register ([TOP]) and the BTM command of the DSP (see Loop Bottom command under 4.5 "Command") are repeated only the number of times indicated by the loop counter. Thus, in order to realize this process, it must be executed after setting values in the top address register and loop counter. Values can be set by the DSP load immediate command (see section on Load Immediate Command under 4.5 "Command"). Execution flow of the Loop program is shown in Figure 4.3.

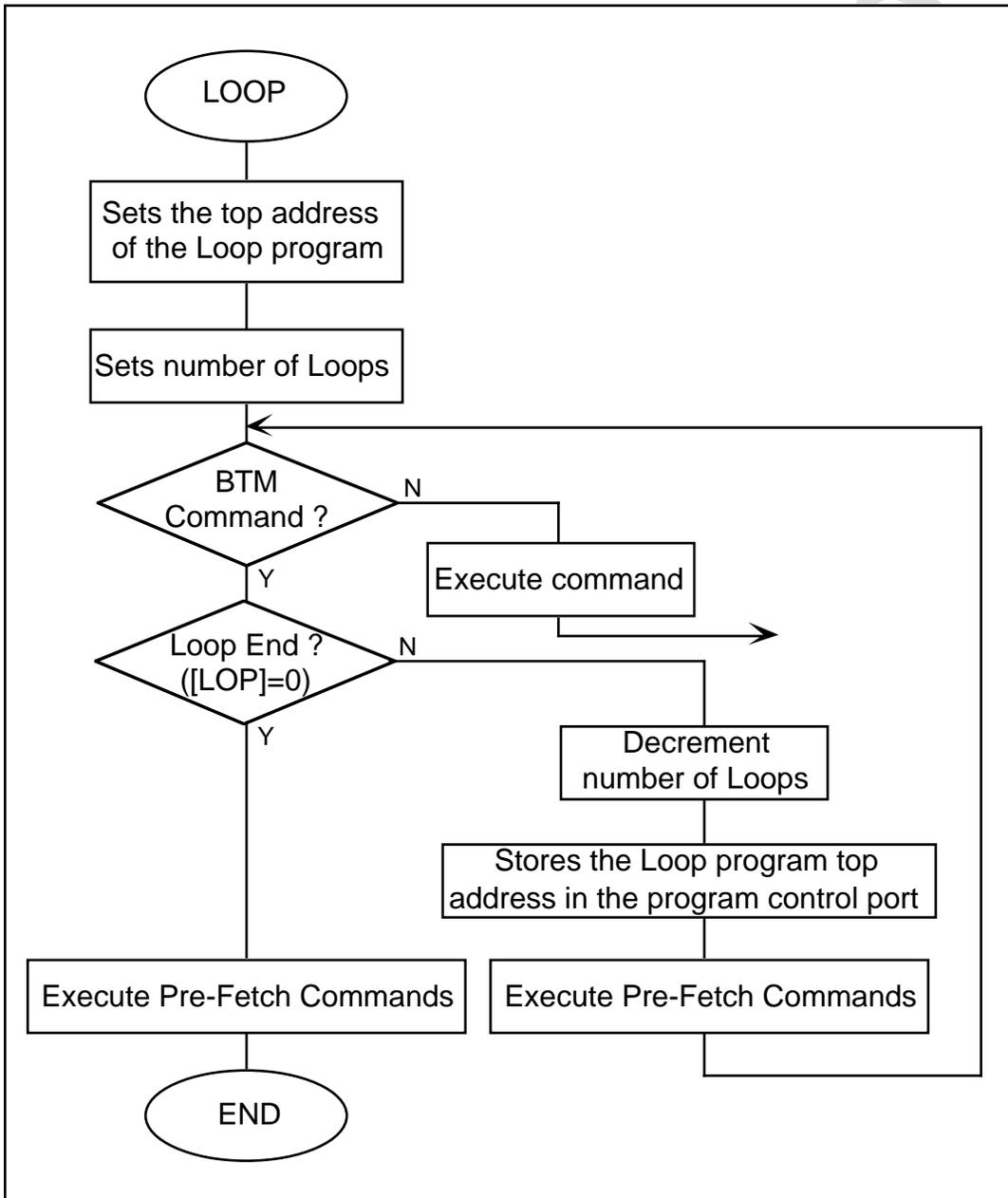


Figure 4.3 Loop Program Execution



DMA Command Execution

This sets the DMA controller register from the DSP and explains the actual process of DMA transfer. The DMA command is divided into the two types, shown below, depending on the transfer direction (read / write).

- 1) Data transfer from the D0-Bus to the DSP.
- 2) Data transfer from the DSP to the D0-Bus.

- Data transfer from D0-Bus to DSP

DSP data RAM transfer begin address and external memory transfer begin address are set in registers ([CT0-3] and [RA0]), and transfer is begun by the DMA command. The command formats up to the DMA command are shown below. See 4.5 "Commands" for more information.

```
MOV  SImm , [CT0]      ; Sets DSP data RAM0 transfer begin address
MVI  Imm , [RA0]       ; Sets external memory transfer begin address
DMA  D0 , [MD0] , SImm ; Begins DMA transfer using the D0 Bus
```

Table 4.6 is a collection of the features of DMA transfer. Because DMA transfer is executed by 1 long word units, setting of the transfer word number (SImm of the DMA command mentioned above) must be done in long word units.

Table 4.6 Features of Data Transfer from D0 Bus to DSP

Item	Feature
Flag Set	T0 flag of the program control port is set
Start and End	Follows the data ready signal from outside. Transfer is done by this signal in long word units. DMA transfer is ended by the end signal from outside, and the program control port T0 flag is reset by this timing.
Address Update	Each time 1 long word is transferred, 1 is added to the DSP data RAM transfer address ([CT0-3]), and the external memory transfer address ([RA0]) is added according to the address add number.
Hold Status	If the DMA command Hold bit (see item 4.5 "Commands" DMA command section) is set to 1, the transfer word number ([TN0]) and external memory transfer address ([RA0]) keep the transfer begin values.

- Data transfer from DSP to D0-Bus

The DSP data RAM transfer begin address and external memory begin address are set in registers ([CT0-3]) and ([WA0]), and transfer is begun by the DMA command. The command formats up to the DMA command are shown below. See item 4.5 for more information.

```
MOV  SImm , [CT0]      ; Sets DSP data RAM0 transfer begin address
MVI  Imm , [WA0]      ; Sets external memory transfer begin address
DMA  [MD0] , D0, SImm ; Begins DMA transfer using the D0 Bus
```

Table 4.7 is a collection of the features of DMA transfer. Because DMA transfer is executed in single long word units, setting of the transfer word number (SImm of the DMA command mentioned above) must be done in long word units.

Table 4.7 Features of Data Transfer from DSP to D0 Bus

Item	Feature
Flag Set	T0 flag of the program control part is set
Start and End	Obeys the data ready signal from outside. Transfer is done by this signal in 1 long word units. DMA transfer is ended by the end signal from outside, and the program control port T0 flag is reset by this timing.
Address Change	Each time 1 long word is transferred, 1 is added to the DSP data RAM transfer address ([CT0-3]), and the external memory transfer address ([WA0]) is added according to the address add number.
Hold Status	If the DMA command Hold bit (see item 4.5 "Commands" DMA command section) is set to 1, the transfer word number ([TN0]) and external memory transfer address ([WA0]) keep the transfer begin values.

END Command Execution

When the END command is recognized, the program control port program RAM address add process is stopped and the program execution control bit (EX flag) is reset. Execution of the DSP program is stopped accordingly. But data transfer by the DMA command continues ignoring this END command until the transfer is completed. The value of the program address when the program terminates stops at the address that follows the address stored in END command.



4.4 Special Process Execution

DSP can execute the following special processes.

- 1) Loading a Program by the DMA command
- 2) Repeating One Command
- 3) Execution of subroutine program

Loading a Program by the DMA command

Loading from the CPU was explained earlier as one method of loading a program (see section 2.3), but a program can be loaded in the DSP program RAM by using the DSP DMA command as well. Loading a program is done in the following formats.

```
MVI Imm , [RA0] ; Sets external memory transfer begin address
DMA D0 , [PRG], SImm ; Sets transfer word number, begins transfer
MVI Imm , [PC] , SImm ; Sets program execution start address
```

Repeating One Command

The format for repeating 1 command is shown below. The 1 command repeat execution command (see LPS command in section 4.5 "Command" under the part on Loop Bottom) repeat the following commands. The repeat number executes one time more than the set value.

```
MVI Imm , [LOP] ; Sets number of repetitions
LPS ; Repeat execution comand
### ; This command is repeatedly executed
```

Executing a SubRoutine Program

There are no special commands (mnemonic) in the DSP program for executing subroutines. By combining the Load Immediate command to the [PC] with the Loop Bottom command, subroutines are created in the form shown in Figure 4.4.

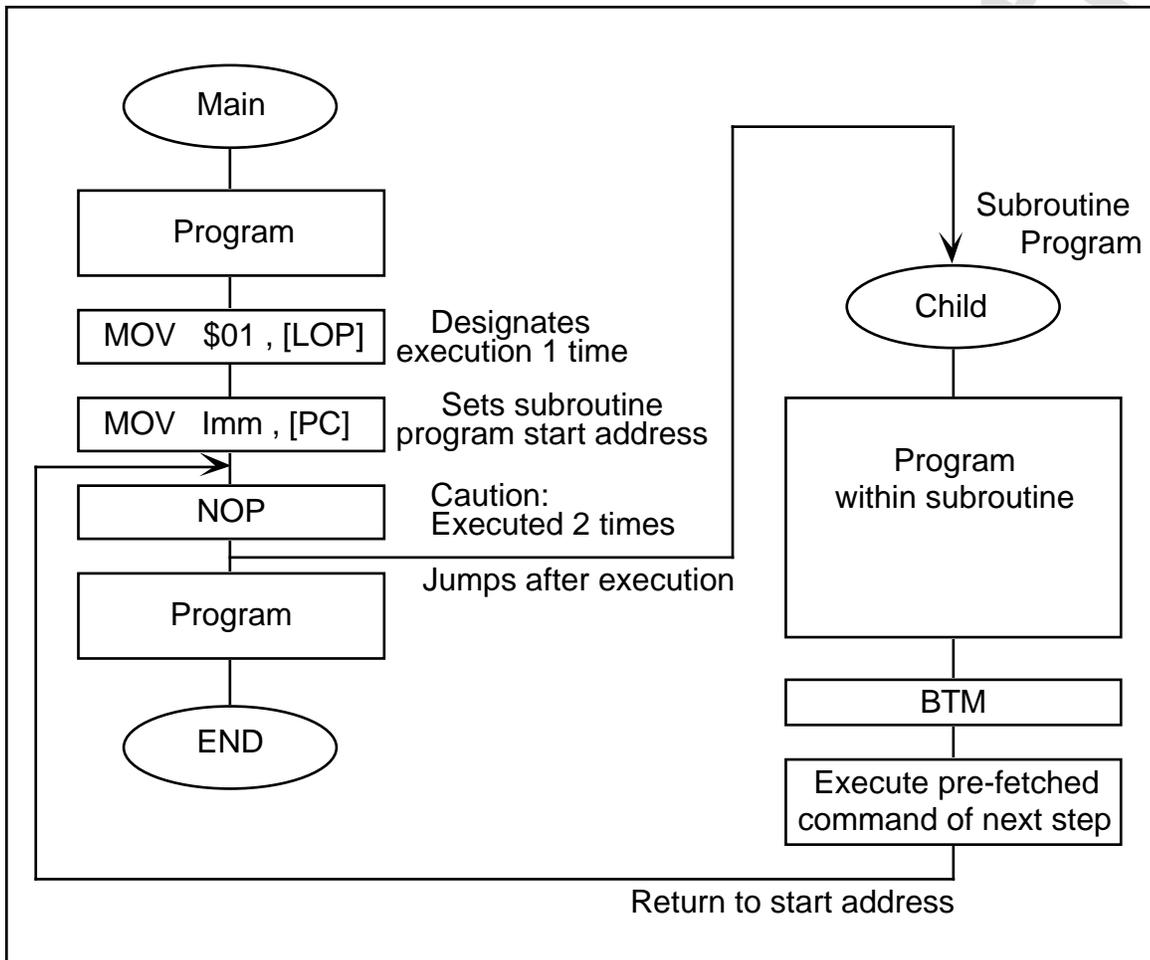


Figure 4.4 Subroutine Program Execution



4.5 More About Commands

Operation Commands

Operation commands use each X, Y, and D1 bus as well as an arithmetic logic unit (ALU). Operation commands can be classified into the following four control types.

- 1) ALU control command
- 2) X-Bus control command
- 3) Y-Bus control command
- 4) D1-Bus control command

The operation command format is as shown in Figure 4.5.

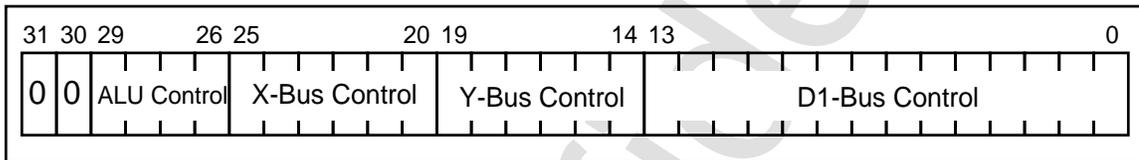


Figure 4.5 Operation Command Format

Operation commands can execute these four types of commands concurrently. Mnemonics should list the ALU control command to the far left. Other required commands should be listed and separated by a space or tab.

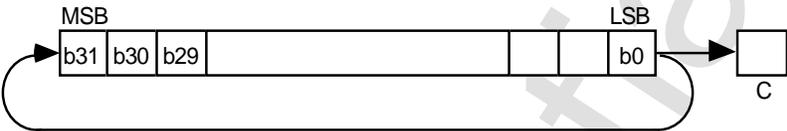
- ALU Control Command
ALU control commands operate using the ALU. The following pages show more about ALU control commands.

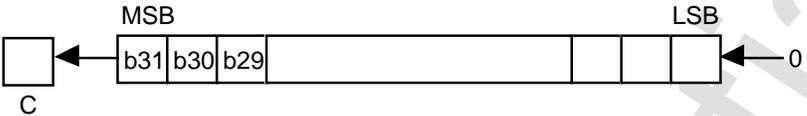
SEGA Confidential



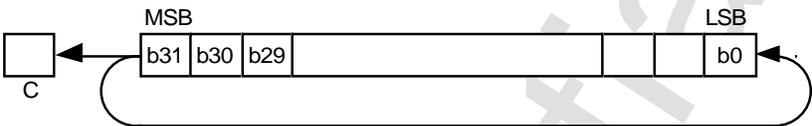
SR		Right Shift 1 Bit
Operation Description	Shifts the value of [ACL] right 1 bit, and the value of bit 0 is stored in C flag. 	
Label	SR	
Instruction Code		
Flag	S ; 1 when operation result MSB is 1,0 when 0. Z ; 1 when operation result is 0, otherwise it is 0. C ; 1 when the value of b0 of input data is 1, 0 when 0. ACL ; Shifts 1 bit to the right, most significant bit (b31) does not change	
Comments		

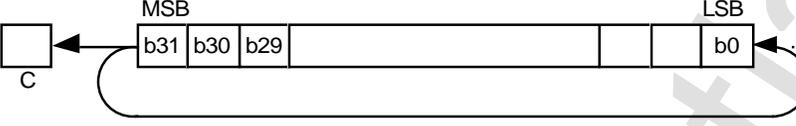
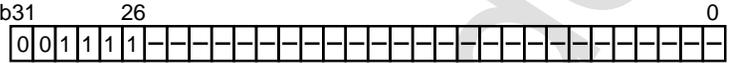


RR	Right Rotate 1 Bit
Operation Description	Rotates the [ACL] value right 1 bit. 
Label	RR
Instruction Code	
Flag	S ; 1 when operation result MSB is 1,0 when 0. Z ; 1 when operation result is 0, otherwise it is 0. C ; 1 when the value of b0 of input data is 1, 0 when 0. ACL ; Shifts 1 bit to the right, least significant bit (b0) moves to the most significant bit (b31).
Comments	

SL	Left Shift 1 Bit
Operation Description	Shifts the [ACL] value left 1 bit. 
Label	SL
Instruction Code	
Flag	S ; 1 when operation result MSB is 1,0 when 0. Z ; 1 when operation result is 0, otherwise it is 0. C ; 1 when the value of b31 of input data is 1, 0 when 0. ACL ; Shifts 1 bit to the left; least significant bit (b0) is 0.
Comments	



RL	Left Rotate 1 Bit
Operation Description	Rotates the [ACL] value left 1 bit. 
Label	RL
Instruction Code	
Flag	S ; 1 when operation result MSB is 1,0 when 0. Z ; 1 when operation result is 0, otherwise it is 0. C ; 1 when the value of b31of input data is 1, 0 when 0. ACL ; Shifts 1 bit to the left, most significant bit (b31) moves to the least significant bit (b0).
Comments	

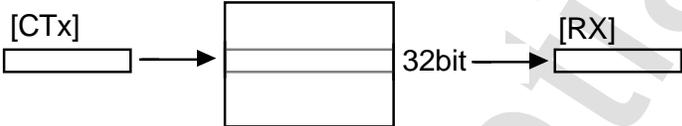
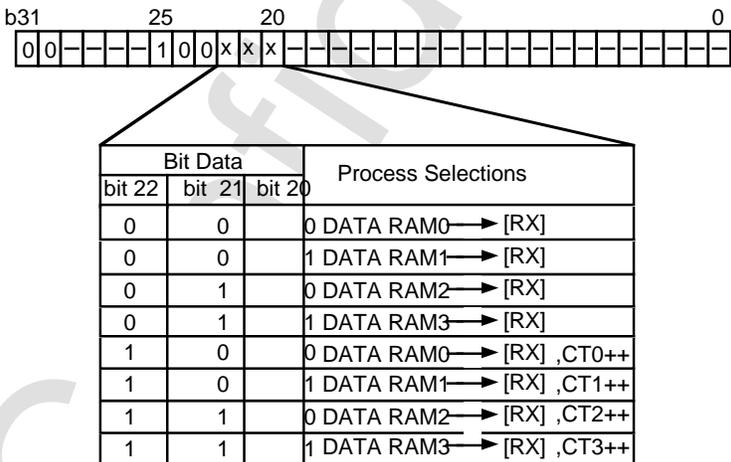
RL8	Left Rotate 8 Bits
Operation Description	Rotates the [ACL] value left 8 bits. 
Label	RL8
Instruction Code	
Flag	S ; 1 when operation result MSB is 1,0 when 0. Z ; 1 when operation result is 0, other wise 0. C ; 1 when the value of b24of input data is 1, 0 when 0. ACL ; Rotates 8bits to the left.
Comments	



- X-Bus Control Commands

X-Bus control commands transfer data using the X-Bus to the RX register and PH, PL registers. The following pages show more about X-Bus control commands.

SEGA Confidential

MOV [s],X	Transfer (Memory →[RX])																																							
<p>Operation Description</p>	<p>Data is transferred to [RX] from the data RAM address displayed by [CTx(x=0~3)].</p>  <p>The diagram shows a box labeled [CTx] with an arrow pointing to a larger box representing a 32-bit data bus. From the right side of this bus, another arrow points to a box labeled [RX].</p>																																							
<p>Label</p>	<p>MOV [Source RAM],X Source RAM = M0 ~ M3 *,MC0 ~ MC3 *</p>																																							
<p>Instruction Code</p>	 <p>The instruction code is shown as a 32-bit field from bit 31 to bit 0. Bits 31 and 30 are 0. Bit 29 is 1. Bits 28, 27, 26, and 25 are dashes. Bit 24 is 0. Bit 23 is 0. Bits 22 and 21 are labeled 'x'. Bit 20 is 'x'. Bits 19 through 0 are dashes.</p> <table border="1" data-bbox="730 850 1266 1176"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Process Selections</th> </tr> <tr> <th>bit 22</th> <th>bit 21</th> <th>bit 20</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td></td> <td>0 DATA RAM0 → [RX]</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1 DATA RAM1 → [RX]</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>0 DATA RAM2 → [RX]</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>1 DATA RAM3 → [RX]</td> </tr> <tr> <td>1</td> <td>0</td> <td></td> <td>0 DATA RAM0 → [RX], CT0++</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>1 DATA RAM1 → [RX], CT1++</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>0 DATA RAM2 → [RX], CT2++</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1 DATA RAM3 → [RX], CT3++</td> </tr> </tbody> </table>	Bit Data			Process Selections	bit 22	bit 21	bit 20	0	0		0 DATA RAM0 → [RX]	0	0	1	1 DATA RAM1 → [RX]	0	1	0	0 DATA RAM2 → [RX]	0	1	1	1 DATA RAM3 → [RX]	1	0		0 DATA RAM0 → [RX], CT0++	1	0	1	1 DATA RAM1 → [RX], CT1++	1	1	0	0 DATA RAM2 → [RX], CT2++	1	1	1	1 DATA RAM3 → [RX], CT3++
Bit Data			Process Selections																																					
bit 22	bit 21	bit 20																																						
0	0		0 DATA RAM0 → [RX]																																					
0	0	1	1 DATA RAM1 → [RX]																																					
0	1	0	0 DATA RAM2 → [RX]																																					
0	1	1	1 DATA RAM3 → [RX]																																					
1	0		0 DATA RAM0 → [RX], CT0++																																					
1	0	1	1 DATA RAM1 → [RX], CT1++																																					
1	1	0	0 DATA RAM2 → [RX], CT2++																																					
1	1	1	1 DATA RAM3 → [RX], CT3++																																					
<p>Flag</p>	<p>* RX ; becomes data selected by multiple choice. CTx(x=0 ~ 3) ; incremented as long as b22 = 1. No change when b22 = 0.</p>																																							
<p>Comments</p>	<p>* [Mx(x=0 ~ 3)] designates DATA RAMx(x=0~3). [MCx(x=0 ~ 3)] designates DATA RAMx(x=0~3) and after transfer, increments [CTx(x=0~3)].</p>																																							

MOV [s],P	Transfer (Memory →[PL])																																							
<p>Operation Description</p>	<p>Data is transferred to [PL] from the data RAM address displayed by [CTx(x=0~3)]. The value of [PH] is changed by the [PL] sign extension.</p>																																							
<p>Label</p>	<p>MOV [Source RAM],P Source RAM = M0 ~ M3,MC0 ~ MC3 *</p>																																							
<p>Instruction Code</p>	<table border="1" data-bbox="730 861 1266 1186"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Process Selections</th> </tr> <tr> <th>bit 22</th> <th>bit 21</th> <th>bit 20</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0 DATA RAM0 → [PL]</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1 DATA RAM1 → [PL]</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>0 DATA RAM2 → [PL]</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>1 DATA RAM3 → [PL]</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>0 DATA RAM0 → [PL] ,CT0++</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>1 DATA RAM1 → [PL] ,CT1++</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>0 DATA RAM2 → [PL] ,CT2++</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1 DATA RAM3 → [PL] ,CT3++</td> </tr> </tbody> </table>	Bit Data			Process Selections	bit 22	bit 21	bit 20	0	0	0	0 DATA RAM0 → [PL]	0	0	1	1 DATA RAM1 → [PL]	0	1	0	0 DATA RAM2 → [PL]	0	1	1	1 DATA RAM3 → [PL]	1	0	0	0 DATA RAM0 → [PL] ,CT0++	1	0	1	1 DATA RAM1 → [PL] ,CT1++	1	1	0	0 DATA RAM2 → [PL] ,CT2++	1	1	1	1 DATA RAM3 → [PL] ,CT3++
Bit Data			Process Selections																																					
bit 22	bit 21	bit 20																																						
0	0	0	0 DATA RAM0 → [PL]																																					
0	0	1	1 DATA RAM1 → [PL]																																					
0	1	0	0 DATA RAM2 → [PL]																																					
0	1	1	1 DATA RAM3 → [PL]																																					
1	0	0	0 DATA RAM0 → [PL] ,CT0++																																					
1	0	1	1 DATA RAM1 → [PL] ,CT1++																																					
1	1	0	0 DATA RAM2 → [PL] ,CT2++																																					
1	1	1	1 DATA RAM3 → [PL] ,CT3++																																					
<p>Flag</p>	<p>PL ; becomes data selected by multiple choice. PH ; changed by [PL] sign extension. CTx(x=0~3) ; incremented when b22 = 1. No change when b22 = 0.</p>																																							
<p>Comments</p>	<p>* [Mx(x=0 ~ 3)] designates DATA RAMx(x=0~3). [MCx(x=0 ~ 3)] designates DATA RAMx(x=0~3) and after transfer increments [CTx(x=0~3)].</p>																																							

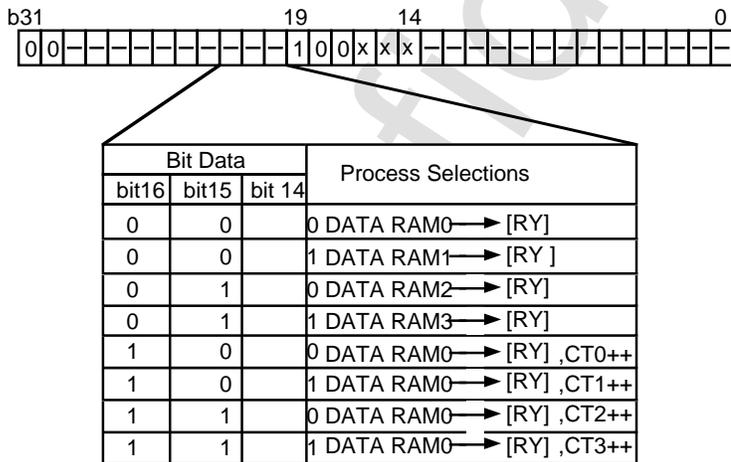
- Y-Bus Control Commands

Y-Bus control commands transfer data using the Y-Bus to the RY register and ACH, ACL registers. The following pages shows more about Y-Bus control commands.

SEGA Confidential



NOP		Y-Bus No Operation
Operation Description	No Y-Bus control process	
Label	NOP	
Instruction Code	<pre> b31 19 17 0 0 0 - - - - - 0 0 0 - - - - - </pre>	
Flag	No change	
Comments		

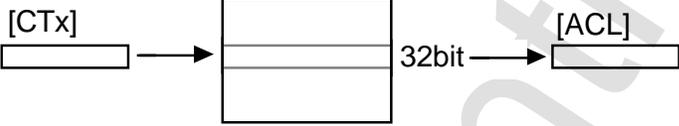
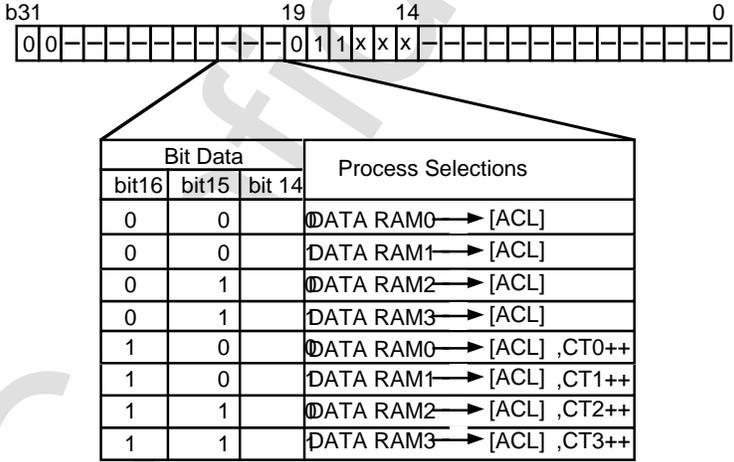
MOV [s],Y	Transfer (Memory →[RY])																																							
Operation Description	<p>Data is transferred to [RY] from the data RAM address displayed by [CTx(x=0~3)].</p>  <p>The diagram shows a box labeled [CTx] on the left with an arrow pointing to a larger box representing a 32-bit bus. From the right side of this bus, another arrow points to a box labeled [RY].</p>																																							
Label	<p>MOV [Source RAM],Y Source RAM = M0 ~ M3,MC0 ~ MC3 *</p>																																							
Instruction Code	 <p>The instruction code is a 32-bit field. Bit 31 is 0, bit 30 is 0, bits 29-20 are dashes, bit 19 is 1, bit 18 is 0, bits 17-15 are x's, bit 14 is x, and bits 13-0 are dashes.</p> <table border="1" data-bbox="584 861 1120 1186"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Process Selections</th> </tr> <tr> <th>bit16</th> <th>bit15</th> <th>bit 14</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td></td> <td>0 DATA RAM0 → [RY]</td> </tr> <tr> <td>0</td> <td>0</td> <td></td> <td>1 DATA RAM1 → [RY]</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> <td>0 DATA RAM2 → [RY]</td> </tr> <tr> <td>0</td> <td>1</td> <td></td> <td>1 DATA RAM3 → [RY]</td> </tr> <tr> <td>1</td> <td>0</td> <td></td> <td>0 DATA RAM0 → [RY], CT0++</td> </tr> <tr> <td>1</td> <td>0</td> <td></td> <td>1 DATA RAM0 → [RY], CT1++</td> </tr> <tr> <td>1</td> <td>1</td> <td></td> <td>0 DATA RAM0 → [RY], CT2++</td> </tr> <tr> <td>1</td> <td>1</td> <td></td> <td>1 DATA RAM0 → [RY], CT3++</td> </tr> </tbody> </table>	Bit Data			Process Selections	bit16	bit15	bit 14	0	0		0 DATA RAM0 → [RY]	0	0		1 DATA RAM1 → [RY]	0	1		0 DATA RAM2 → [RY]	0	1		1 DATA RAM3 → [RY]	1	0		0 DATA RAM0 → [RY], CT0++	1	0		1 DATA RAM0 → [RY], CT1++	1	1		0 DATA RAM0 → [RY], CT2++	1	1		1 DATA RAM0 → [RY], CT3++
Bit Data			Process Selections																																					
bit16	bit15	bit 14																																						
0	0		0 DATA RAM0 → [RY]																																					
0	0		1 DATA RAM1 → [RY]																																					
0	1		0 DATA RAM2 → [RY]																																					
0	1		1 DATA RAM3 → [RY]																																					
1	0		0 DATA RAM0 → [RY], CT0++																																					
1	0		1 DATA RAM0 → [RY], CT1++																																					
1	1		0 DATA RAM0 → [RY], CT2++																																					
1	1		1 DATA RAM0 → [RY], CT3++																																					
Flag	<p>RY ; becomes data selected by multiple choice. CTx(x=0~3) ; incremented when b16 = 1. No change when b16 = 0.</p>																																							
Comments	<p>* [Mx(x=0 ~ 3)] designates DATA RAMx(x=0~3). [MCx(x=0 ~ 3)] designates DATA RAMx(x=0~3) and after transfer increments [CTx(x=0~3)].</p>																																							



CLR A		0 Clear
Operation Description	0 clears the [ACH] and [ACL] values.	
Label	CLR A	
Instruction Code	<pre> b31 19 14 0 0 0 --- --- --- --- --- --- 0 0 1 --- --- --- --- --- </pre>	
Flag	ACH ; becomes 0 ACL ; becomes 0	
Comments		

MOV ALU,A	Transfer ([ALU] →[ACH][ACL])
Operation Description	<p>Transfers the value of the [ALU] high order 16 bit to [ACH] and the value of the [ALU] low order 32 bit to [ACL].</p> <div data-bbox="487 346 1128 525" data-label="Diagram"> <pre> graph TD ALU[ALU] --- H[16bit] ALU --- L[32bit] H --> ACH[ACH] L --> ACL[ACL] </pre> </div>
Label	MOV ALU,A
Instruction Code	<div data-bbox="487 682 1226 766" data-label="Diagram"> <pre> b31 19 14 0 0 0 - - - - - - - - - - 0 1 0 - 0 </pre> </div>
Flag	<p>ACH ; becomes ALU high order 16 bit data ACL ; becomes ALU low order 32 bit data</p>
Comments	



MOV [s],A	Transfer (Memory →[ACL])																																							
<p>Operation Description</p>	<p>Data is transferred to [ACL] from the data RAM address displayed by [CTx(x=0~3)]. The value of [ACH] is changed by the sign extension of [ACL].</p>  <p>The diagram shows a box labeled [CTx] on the left with an arrow pointing to a larger box representing a 32-bit bus. From the right side of this bus, another arrow points to a box labeled [ACL]. The text '32bit' is placed between the bus and the [ACL] box.</p>																																							
<p>Label</p>	<p>MOV [Source RAM],A Source RAM = MO ~ M3,MC0 ~ MC3 *</p>																																							
<p>Instruction Code</p>	 <p>The instruction code is shown as a 32-bit field: <code>0 0 [bits 17-19] 0 1 1 x x x [bits 1-13]</code>. A callout box provides the following bit data table:</p> <table border="1" data-bbox="737 856 1273 1184"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Process Selections</th> </tr> <tr> <th>bit16</th> <th>bit15</th> <th>bit 14</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>DATA RAM0 → [ACL]</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>DATA RAM1 → [ACL]</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>DATA RAM2 → [ACL]</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>DATA RAM3 → [ACL]</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>DATA RAM0 → [ACL] ,CT0++</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>DATA RAM1 → [ACL] ,CT1++</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>DATA RAM2 → [ACL] ,CT2++</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>DATA RAM3 → [ACL] ,CT3++</td> </tr> </tbody> </table>	Bit Data			Process Selections	bit16	bit15	bit 14	0	0	0	DATA RAM0 → [ACL]	0	0	1	DATA RAM1 → [ACL]	0	1	0	DATA RAM2 → [ACL]	0	1	1	DATA RAM3 → [ACL]	1	0	0	DATA RAM0 → [ACL] ,CT0++	1	0	1	DATA RAM1 → [ACL] ,CT1++	1	1	0	DATA RAM2 → [ACL] ,CT2++	1	1	1	DATA RAM3 → [ACL] ,CT3++
Bit Data			Process Selections																																					
bit16	bit15	bit 14																																						
0	0	0	DATA RAM0 → [ACL]																																					
0	0	1	DATA RAM1 → [ACL]																																					
0	1	0	DATA RAM2 → [ACL]																																					
0	1	1	DATA RAM3 → [ACL]																																					
1	0	0	DATA RAM0 → [ACL] ,CT0++																																					
1	0	1	DATA RAM1 → [ACL] ,CT1++																																					
1	1	0	DATA RAM2 → [ACL] ,CT2++																																					
1	1	1	DATA RAM3 → [ACL] ,CT3++																																					
<p>Flag</p>	<p>ACL ; becomes data selected by multiple choice. ACH ; is changed by the sign extension of [ACL] CTx(x=0~3) ; incremented when b16 = 1. No change when b16 = 0.</p>																																							
<p>Comments</p>	<p>* [Mx(x=0 ~ 3)] designates DATA RAMx(x=0~3). [MCx(x=0 ~ 3)] designates DATA RAMx(x=0~3) and after transfer increments [CTx(x=0~3)].</p>																																							

- D1-Bus Control Commands

D1-Bus control commands control the exchange of data between memory connected to the D1-Bus. The following pages shows more about D1-Bus control commands.

SEGA Confidential



NOP		D1-Bus No Operation
Operation Description	No D1-Bus control process	
Label	NOP	
Instruction Code	<div style="text-align: center;"> b31 13 0 </div>	
Flag	No change	
Comments		

MOV SImm,[d]	Transfer (SImm →[destination])																																																																																									
Operation Description	<p>SImm data is transferred to the RAM or register designated by [destination]. SImm data is signed 8 bit data.</p> <p>Short Immediate Data → [destination]</p> <p style="margin-left: 150px;">D31 - 7 ← b7 D6-0 ← b6-0</p>																																																																																									
Label	<p>MOV SImm,[Destination] Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,TOP,CT0 ~ CT3</p>																																																																																									
Instruction Code	<div style="text-align: center;"> </div> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit11</th> <th>bit10</th> <th>bit 9</th> <th>bit 8</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>[TOP]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[CT0]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>[CT1]</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>[CT2]</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>[CT3]</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit11	bit10	bit 9	bit 8	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	[TOP]	1	1	0	0	[CT0]	1	1	0	1	[CT1]	1	1	1	0	[CT2]	1	1	1	1	[CT3]
Bit Data				[d] Selections																																																																																						
bit11	bit10	bit 9	bit 8																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	[TOP]																																																																																						
1	1	0	0	[CT0]																																																																																						
1	1	0	1	[CT1]																																																																																						
1	1	1	0	[CT2]																																																																																						
1	1	1	1	[CT3]																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAMx(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									



MOV [s],[d]	Transfer ([source]→[destination])																																																																																																																																																																																				
Operation Description	RAM data or register data designated by [source] is transferred to the RAM or register designated by [destination].																																																																																																																																																																																				
Label	MOV [Source], [Destination] Source = M0 ~ M3 *,MC0 ~ MC3 *,ALH,ALL Destination = MC0 ~ MC3,RX,PL,RA0,WA0,LOP, TOP,CT0 ~ CT3																																																																																																																																																																																				
Instruction Code	<div style="text-align: center;"> </div> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="4">Bit Data</th> <th>[d] Selections</th> <th colspan="4">Bit Data</th> <th>[s] Selections</th> </tr> <tr> <th>bit11</th> <th>bit10</th> <th>bit 9</th> <th>bit 8</th> <th></th> <th>bit 3</th> <th>bit 2</th> <th>bit 1</th> <th>bit 0</th> <th></th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0,CT0++</td><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1,CT1++</td><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2,CT2++</td><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3,CT3++</td><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td><td>0</td><td>1</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td><td>0</td><td>1</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td><td>0</td><td>1</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td><td>0</td><td>1</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td><td>1</td><td>0</td><td>0</td><td>0</td><td>no field</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td><td>1</td><td>0</td><td>0</td><td>1</td><td>[ALU LOW]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td><td>1</td><td>0</td><td>1</td><td>0</td><td>[ALU HIGH]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>[TOP]</td><td></td><td></td><td></td><td></td><td></td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[CT0]</td><td></td><td></td><td></td><td></td><td></td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>[CT1]</td><td></td><td></td><td></td><td></td><td></td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>[CT2]</td><td></td><td></td><td></td><td></td><td></td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>[CT3]</td><td></td><td></td><td></td><td></td><td></td></tr> </tbody> </table>	Bit Data				[d] Selections	Bit Data				[s] Selections	bit11	bit10	bit 9	bit 8		bit 3	bit 2	bit 1	bit 0		0	0	0	0	DATA RAM0,CT0++	0	0	0	0	DATA RAM0	0	0	0	1	DATA RAM1,CT1++	0	0	0	1	DATA RAM1	0	0	1	0	DATA RAM2,CT2++	0	0	1	0	DATA RAM2	0	0	1	1	DATA RAM3,CT3++	0	0	1	1	DATA RAM3	0	1	0	0	[RX]	0	1	0	0	DATA RAM0 ,CT0++	0	1	0	1	[PL]	0	1	0	1	DATA RAM1 ,CT1++	0	1	1	0	[RA0]	0	1	1	0	DATA RAM2 ,CT2++	0	1	1	1	[WA0]	0	1	1	1	DATA RAM3 ,CT3++	1	0	0	0	unused	1	0	0	0	no field	1	0	0	1	unused	1	0	0	1	[ALU LOW]	1	0	1	0	[LOP]	1	0	1	0	[ALU HIGH]	1	0	1	1	[TOP]						1	1	0	0	[CT0]						1	1	0	1	[CT1]						1	1	1	0	[CT2]						1	1	1	1	[CT3]					
Bit Data				[d] Selections	Bit Data				[s] Selections																																																																																																																																																																												
bit11	bit10	bit 9	bit 8		bit 3	bit 2	bit 1	bit 0																																																																																																																																																																													
0	0	0	0	DATA RAM0,CT0++	0	0	0	0	DATA RAM0																																																																																																																																																																												
0	0	0	1	DATA RAM1,CT1++	0	0	0	1	DATA RAM1																																																																																																																																																																												
0	0	1	0	DATA RAM2,CT2++	0	0	1	0	DATA RAM2																																																																																																																																																																												
0	0	1	1	DATA RAM3,CT3++	0	0	1	1	DATA RAM3																																																																																																																																																																												
0	1	0	0	[RX]	0	1	0	0	DATA RAM0 ,CT0++																																																																																																																																																																												
0	1	0	1	[PL]	0	1	0	1	DATA RAM1 ,CT1++																																																																																																																																																																												
0	1	1	0	[RA0]	0	1	1	0	DATA RAM2 ,CT2++																																																																																																																																																																												
0	1	1	1	[WA0]	0	1	1	1	DATA RAM3 ,CT3++																																																																																																																																																																												
1	0	0	0	unused	1	0	0	0	no field																																																																																																																																																																												
1	0	0	1	unused	1	0	0	1	[ALU LOW]																																																																																																																																																																												
1	0	1	0	[LOP]	1	0	1	0	[ALU HIGH]																																																																																																																																																																												
1	0	1	1	[TOP]																																																																																																																																																																																	
1	1	0	0	[CT0]																																																																																																																																																																																	
1	1	0	1	[CT1]																																																																																																																																																																																	
1	1	1	0	[CT2]																																																																																																																																																																																	
1	1	1	1	[CT3]																																																																																																																																																																																	
Flag	Area selected by [d] selection is data of an area selected by [s] selection																																																																																																																																																																																				
Comments	* [Mx(x=0 ~ 3)] designates DATA RAM x(x=0~3) [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																																																																																																																				

Load Immediate Command

The load immediate command transfers immediate data to the storage destination. Unconditional transfer follows the format in Figure 4.6. Conditional transfer follows the format in Figure 4.7. Details are on the following pages.

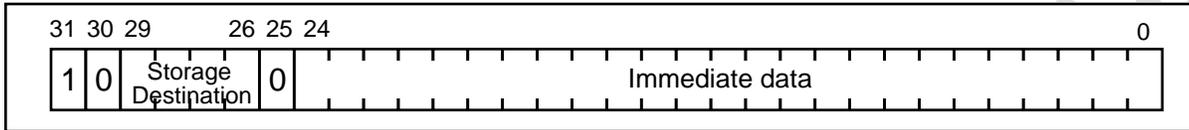


Figure 4.6 Load Immediate Command Format 1 (Unconditional Transfer)

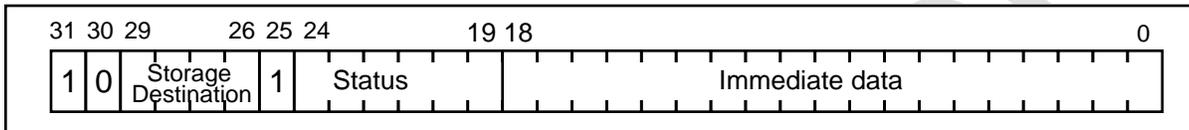
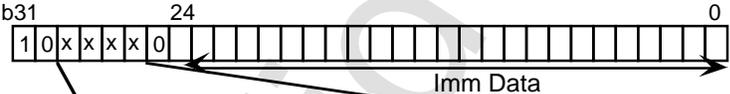
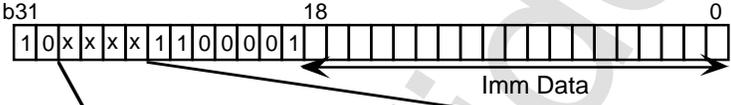


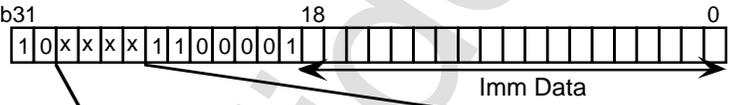
Figure 4.7 Load Immediate Command Format 2 (Conditional Transfer)



MVI Imm,[d]	Unconditional Transfer (Imm →[destination])																																																																																									
Operation Description	Imm data is unconditional and is transferred to the RAM or register designated by [destination]. Imm data is signed 25 bit data.																																																																																									
Label	MVI Imm,[Destination] Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="706 829 1250 1417" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit11</th> <th>bit10</th> <th>bit 9</th> <th>bit 8</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP],[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit11	bit10	bit 9	bit 8	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP],[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit11	bit10	bit 9	bit 8																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP],[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] multiple choice ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									

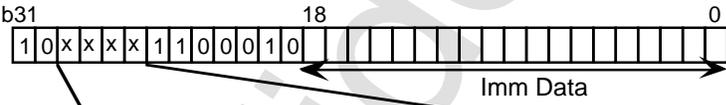
MVI Imm,[d]Z	Conditional Transfer (Z=1 then Imm →[destination])																																																																																									
Operation Description	<p>When the Z flag is 1, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],Z Destination RAM = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="560 819 1104 1396" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									

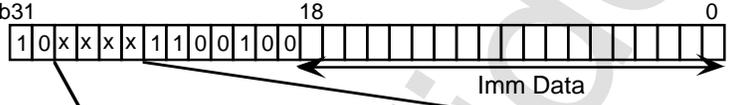


MVI =Imm,[d]NZ	Conditional Transfer (Z=0 then Imm →[destination])																																																																																									
Operation Description	<p>When the Z flag is 0, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],NZ Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="711 819 1258 1402" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer increments [CTx(x=0~3)].																																																																																									

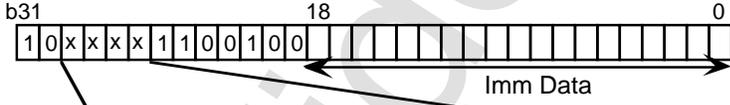
MVI Imm,[d]S	Conditional Transfer (S=1 then Imm →[destination])																																																																																									
Operation Description	<p>When the S flag is 1, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. The address next after this command will be executed twice, once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],S Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;"> </div> <table border="1" style="margin: 10px auto; border-collapse: collapse; text-align: center;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									

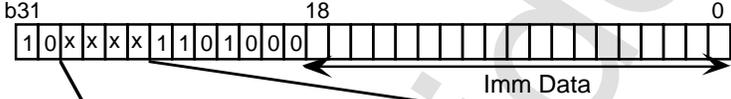


MVI Imm,[d]NS	Conditional Transfer (S=0 then Imm →[destination])																																																																																									
Operation Description	<p>When the S flag is 0, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used asexecution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],NS Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	 <table border="1" data-bbox="706 819 1258 1402"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									

MVI Imm,[d]C	Conditional Transfer (C=1 then Imm →[destination])																																																																																									
Operation Description	<p>When the C flag is 1, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used asexecution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],C Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="552 819 1096 1396" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									

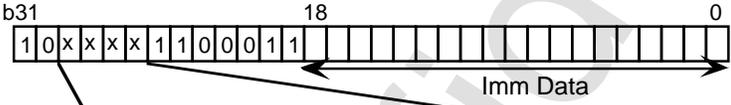


MVI Imm,[d]NC	Conditional Transfer (C=0 then Imm →[destination])																																																																																									
Operation Description	<p>When the C flag is 0, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],NC Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	 <table border="1" data-bbox="706 819 1258 1396"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									

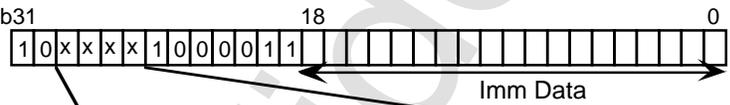
MVI Imm,[d],T0	Conditional Transfer (T0=1 then Imm →[destination])																																																																																									
Operation Description	<p>When the T0 flag is 1, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],T0 Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	<p>Area selected by [d] selection ; becomes Imm data</p>																																																																																									
Comments	<p>* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].</p>																																																																																									



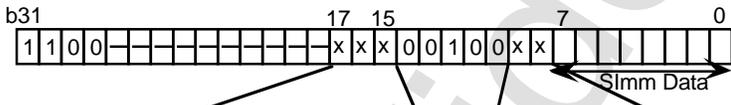
MVI Imm,[d]NT0	Conditional Transfer (T0=0 then Imm →[destination])																																																																																									
Operation Description	<p>When the T0 flag is 0, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data.</p> <p>Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],NT0 Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="706 829 1250 1407" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	<p>Area selected by [d] selection ; becomes Imm data</p>																																																																																									
Comments	<p>* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].</p>																																																																																									

MVI Imm,[d]ZS	Conditional Transfer (Z=1 or S=1 then Imm [destination]) →																																																																																									
Operation Description	<p>When the Z flag or S flag is 1, Imm data is transferred to the RAM or register designated by [destination]. Imm data is signed 19 bit data. Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],ZS Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="560 840 1104 1417" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	Area selected by [d] selection ; becomes Imm data																																																																																									
Comments	* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].																																																																																									



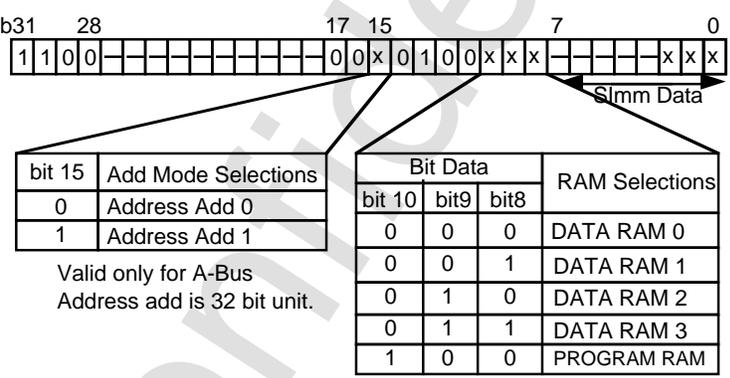
MVI Imm,[d]NZS	Conditional Transfer (Z=0=S then Imm → [destination])																																																																																									
Operation Description	<p>When the Z flag or S flag are both 0, Imm data is transferred to the RA or register designated by [destination]. Imm data is signed 19 bit data. Can be used as execution of the subroutine program (see instruction code**) by sending Imm data (subroutine begin dress) to the PC and saving the PC (jump address after subroutine ends) value to TOP. Be aware that the address next after this command will be executed twice once before the subroutine and once after.</p>																																																																																									
Label	<p>MVI Imm,[Destination],NZS Destination = MC0 ~ MC3 *,RX,PL,RA0,WA0,LOP,PC</p>																																																																																									
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="711 821 1256 1409" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="4">Bit Data</th> <th rowspan="2">[d] Selections</th> </tr> <tr> <th>bit29</th> <th>bit28</th> <th>bit27</th> <th>bit26</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>DATA RAM0 ,CT0++</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>DATA RAM1 ,CT1++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>DATA RAM2 ,CT2++</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>DATA RAM3 ,CT3++</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>[RX]</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>[PL]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>[RA0]</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>[WA0]</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>[LOP]</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>[PC] → [TOP] ,[PC]</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>unused</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>unused</td></tr> </tbody> </table>	Bit Data				[d] Selections	bit29	bit28	bit27	bit26	0	0	0	0	DATA RAM0 ,CT0++	0	0	0	1	DATA RAM1 ,CT1++	0	0	1	0	DATA RAM2 ,CT2++	0	0	1	1	DATA RAM3 ,CT3++	0	1	0	0	[RX]	0	1	0	1	[PL]	0	1	1	0	[RA0]	0	1	1	1	[WA0]	1	0	0	0	unused	1	0	0	1	unused	1	0	1	0	[LOP]	1	0	1	1	unused	1	1	0	0	[PC] → [TOP] ,[PC]	1	1	0	1	unused	1	1	1	0	unused	1	1	1	1	unused
Bit Data				[d] Selections																																																																																						
bit29	bit28	bit27	bit26																																																																																							
0	0	0	0	DATA RAM0 ,CT0++																																																																																						
0	0	0	1	DATA RAM1 ,CT1++																																																																																						
0	0	1	0	DATA RAM2 ,CT2++																																																																																						
0	0	1	1	DATA RAM3 ,CT3++																																																																																						
0	1	0	0	[RX]																																																																																						
0	1	0	1	[PL]																																																																																						
0	1	1	0	[RA0]																																																																																						
0	1	1	1	[WA0]																																																																																						
1	0	0	0	unused																																																																																						
1	0	0	1	unused																																																																																						
1	0	1	0	[LOP]																																																																																						
1	0	1	1	unused																																																																																						
1	1	0	0	[PC] → [TOP] ,[PC]																																																																																						
1	1	0	1	unused																																																																																						
1	1	1	0	unused																																																																																						
1	1	1	1	unused																																																																																						
Flag	<p>Area selected by [d] selection ; becomes Imm data</p>																																																																																									
Comments	<p>* [MCx(x=0 ~ 3)] designates DATA RAM x(x=0~3) and, after transfer, increments [CTx(x=0~3)].</p>																																																																																									

DMA D0,[RAM],Slmm	DMA Transfer (D0[31-0] → RAM)																																																								
Operation Description	D0[31-0] data is transferred to RAM . The external address register and transfer word number register are updated (added) according to the address add number. The transfer word number register is a register for storing the transfer word number in long word units. This word number is either 0 or transfer ends when forced to end.																																																								
Label	DMA D0,[Destination],Counter Destination = M0 ~ M3 *																																																								
Instruction Code	<div style="text-align: center;"> </div> <table border="1" style="margin: 10px auto; width: 60%;"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Add Mode Selections</th> </tr> <tr> <th>bit17</th> <th>bit16</th> <th>bit15</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>Address Add 0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>Address Add 1</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>Address Add 2</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>Address Add 4</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>Address Add 8</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>Address Add 16</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>Address Add 32</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>Address Add 64</td></tr> </tbody> </table> <table border="1" style="margin: 10px auto; width: 30%;"> <thead> <tr> <th colspan="2">Bit Data</th> <th rowspan="2">Selections</th> </tr> <tr> <th>bit9</th> <th>bit8</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>DATA RAM 0</td></tr> <tr><td>0</td><td>1</td><td>DATA RAM 1</td></tr> <tr><td>1</td><td>0</td><td>DATA RAM 2</td></tr> <tr><td>1</td><td>1</td><td>DATA RAM 3</td></tr> </tbody> </table>	Bit Data			Add Mode Selections	bit17	bit16	bit15	0	0	0	Address Add 0	0	0	1	Address Add 1	0	1	0	Address Add 2	0	1	1	Address Add 4	1	0	0	Address Add 8	1	0	1	Address Add 16	1	1	0	Address Add 32	1	1	1	Address Add 64	Bit Data		Selections	bit9	bit8	0	0	DATA RAM 0	0	1	DATA RAM 1	1	0	DATA RAM 2	1	1	DATA RAM 3
Bit Data			Add Mode Selections																																																						
bit17	bit16	bit15																																																							
0	0	0	Address Add 0																																																						
0	0	1	Address Add 1																																																						
0	1	0	Address Add 2																																																						
0	1	1	Address Add 4																																																						
1	0	0	Address Add 8																																																						
1	0	1	Address Add 16																																																						
1	1	0	Address Add 32																																																						
1	1	1	Address Add 64																																																						
Bit Data		Selections																																																							
bit9	bit8																																																								
0	0	DATA RAM 0																																																							
0	1	DATA RAM 1																																																							
1	0	DATA RAM 2																																																							
1	1	DATA RAM 3																																																							
Flag	T0 ; becomes 1.																																																								
Comments	<p>* [MCx(x=0 ~ 3)] selects DATA RAM x(x=0~3).</p> <p>**When the END signal informing you that transfer end from outside has been entered, T0; becomes 0.</p> <p>Designating address-add adds an add number after the command and becomes DMA0~DMA64.</p> <p>Add number is1 when address add number designation is omitted.</p> <p>The transfer source address is set in advance to RA0 and the transfer destination RAM address is set in advance to CTx.</p>																																																								

DMA [RAM],D0,SImm	DMA Transfer (RAM → D0[31-0])																																																								
Operation Description	RAM data is transferred to D0[31-0]. The external address register and transfer word number register are updated (added) according to the address add number. Only add numbers 0 and 1 are valid for the A Bus and the write unit is 32bit. All add numbers (0 - 64) are valid for the B-Bus. Write unit is 16bit; 32bit data is divided in half and written at intervals of 16X (0-64). The transfer word number register is a register for storing the transfer word number in long word units. This word number is either 0 or transfer ends when forced to end.																																																								
Label	DMA [Source],D0,Counter Source = Mo ~ M3 *																																																								
Instruction Code	<div style="text-align: center;">  </div> <table border="1" data-bbox="479 850 901 1176" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Add Mode Selections</th> </tr> <tr> <th>bit17</th> <th>bit16</th> <th>bit15</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>Address Add 0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>Address Add 1</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>Address Add 2</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>Address Add 4</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>Address Add 8</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>Address Add 16</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>Address Add 32</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>Address Add 64</td></tr> </tbody> </table> <table border="1" data-bbox="941 850 1226 1050" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th colspan="2">Bit Data</th> <th rowspan="2">Selections</th> </tr> <tr> <th>bit9</th> <th>bit8</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>DATA RAM 0</td></tr> <tr><td>0</td><td>1</td><td>DATA RAM 1</td></tr> <tr><td>1</td><td>0</td><td>DATA RAM 2</td></tr> <tr><td>1</td><td>1</td><td>DATA RAM 3</td></tr> </tbody> </table>	Bit Data			Add Mode Selections	bit17	bit16	bit15	0	0	0	Address Add 0	0	0	1	Address Add 1	0	1	0	Address Add 2	0	1	1	Address Add 4	1	0	0	Address Add 8	1	0	1	Address Add 16	1	1	0	Address Add 32	1	1	1	Address Add 64	Bit Data		Selections	bit9	bit8	0	0	DATA RAM 0	0	1	DATA RAM 1	1	0	DATA RAM 2	1	1	DATA RAM 3
Bit Data			Add Mode Selections																																																						
bit17	bit16	bit15																																																							
0	0	0	Address Add 0																																																						
0	0	1	Address Add 1																																																						
0	1	0	Address Add 2																																																						
0	1	1	Address Add 4																																																						
1	0	0	Address Add 8																																																						
1	0	1	Address Add 16																																																						
1	1	0	Address Add 32																																																						
1	1	1	Address Add 64																																																						
Bit Data		Selections																																																							
bit9	bit8																																																								
0	0	DATA RAM 0																																																							
0	1	DATA RAM 1																																																							
1	0	DATA RAM 2																																																							
1	1	DATA RAM 3																																																							
Flag	T0 ; becomes 1.																																																								
Comments	<p>* [MCx(x=0 ~ 3)] selects DATA RAM x(x=0~3).</p> <p>**When the END signal informing you that transfer end from outside has been entered, T0; becomes 0.</p> <p>Designating address-add adds an add number after the command and becomes DMA0~DMA64.</p> <p>Add number is 1 when address add number designation is omitted.</p> <p>The transfer source RAM address is set in advance to CTx and the transfer destination address is set in advance to WA0.</p>																																																								



DMA D0,[RAM],[s]		DMA Transfer (D0[31-0] → RAM)																																																																								
Operation Description	[s] data designated by bit0~2 is treated as a transfer counter, and only numbers displayed transfer D0[31-0] data to the RAM. External address register and transfer word number register are updated (added) according to the address add number. The transfer word number register stores transfer word numbers in long word units. This word number becomes 0 or transfer ends when forced to end.																																																																									
Label	DMA D0,[Destination],[Counter] Counter = M0 ~ M3 *,MC0~MC3* Destination = M0~M3 *,PRG *																																																																									
Instruction Code	<p>Valid only for A-Bus Address add is 32bit units.</p> <table border="1"> <tr> <td>bit 15</td> <td>Add Mode Selections</td> </tr> <tr> <td>0</td> <td>Address Add 0</td> </tr> <tr> <td>1</td> <td>Address Add 1</td> </tr> </table> <table border="1"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">RAM Selections</th> </tr> <tr> <th>bit 10</th> <th>bit9</th> <th>bit8</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>DATA RAM 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>DATA RAM 1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>DATA RAM 2</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>DATA RAM 3</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>PROGRAM RAM</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">[s] Selections</th> </tr> <tr> <th>bit 2</th> <th>bit1</th> <th>bit 0</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>DATA RAM 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>DATA RAM 1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>DATA RAM 2</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>DATA RAM 3</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>DATA RAM 0,CT0++</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>DATA RAM 1,CT1++</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>DATA RAM 2,CT2++</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>DATA RAM 3,CT3++</td> </tr> </tbody> </table>		bit 15	Add Mode Selections	0	Address Add 0	1	Address Add 1	Bit Data			RAM Selections	bit 10	bit9	bit8	0	0	0	DATA RAM 0	0	0	1	DATA RAM 1	0	1	0	DATA RAM 2	0	1	1	DATA RAM 3	1	0	0	PROGRAM RAM	Bit Data			[s] Selections	bit 2	bit1	bit 0	0	0	0	DATA RAM 0	0	0	1	DATA RAM 1	0	1	0	DATA RAM 2	0	1	1	DATA RAM 3	1	0	0	DATA RAM 0,CT0++	1	0	1	DATA RAM 1,CT1++	1	1	0	DATA RAM 2,CT2++	1	1	1	DATA RAM 3,CT3++
bit 15	Add Mode Selections																																																																									
0	Address Add 0																																																																									
1	Address Add 1																																																																									
Bit Data			RAM Selections																																																																							
bit 10	bit9	bit8																																																																								
0	0	0	DATA RAM 0																																																																							
0	0	1	DATA RAM 1																																																																							
0	1	0	DATA RAM 2																																																																							
0	1	1	DATA RAM 3																																																																							
1	0	0	PROGRAM RAM																																																																							
Bit Data			[s] Selections																																																																							
bit 2	bit1	bit 0																																																																								
0	0	0	DATA RAM 0																																																																							
0	0	1	DATA RAM 1																																																																							
0	1	0	DATA RAM 2																																																																							
0	1	1	DATA RAM 3																																																																							
1	0	0	DATA RAM 0,CT0++																																																																							
1	0	1	DATA RAM 1,CT1++																																																																							
1	1	0	DATA RAM 2,CT2++																																																																							
1	1	1	DATA RAM 3,CT3++																																																																							
Flag	T0 ; becomes 1. ** CTx(x0~3) ; incremented when b2=1. When b2=0, there is no change																																																																									
Comments	<p>* [MCx(x=0 ~ 3)] selects DATA RAM x(x=0~3). MCx(x=0~3) selects DATA RAM x(x=0~3), and after transfer increments CTx(x0~3). PRG selects program RAM.</p> <p>**When the END signal informing you that transfer end from outside has been entered, T0; becomes 0.</p> <p>Designating address-add adds an add number after the command and becomes DMA0~DMA1.</p> <p>Add number is 1 when address add number designation is omitted.</p> <p>The transfer source address is set in advance to RA0 and the transfer destination RAM address is set in advance to CTx.</p>																																																																									

DMAH D0,[RAM],SImm	DMA Transfer (D0[31-0] → RAM) by HOLD Status																																		
Operation Description	D0[31-0] data is transferred to the RAM. External address register and transfer word number register save the value at the time transfer begins. The transfer word number register stores the transfer word number in long word units. This word number becomes 0 or becomes the transfer end when forced to end.																																		
Label	DMAH D0,[Destination],[Counter] Destination = M0~M3 *,PR*																																		
Instruction Code	 <p>The diagram shows a 32-bit instruction code with bit positions b31 to 0. Bits 31-28 are fixed as 1100. Bits 17-15 are 00x. Bits 14-10 are 010. Bits 9-7 are xxx. Bits 6-3 are fixed as 0000. Bits 2-0 are xxx. A double-headed arrow labeled 'SImm Data' spans bits 2-0.</p> <table border="1" data-bbox="630 793 941 892"> <thead> <tr> <th>bit 15</th> <th>Add Mode Selections</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Address Add 0</td> </tr> <tr> <td>1</td> <td>Address Add 1</td> </tr> </tbody> </table> <p>Valid only for A-Bus Address add is 32 bit unit.</p> <table border="1" data-bbox="971 793 1334 1012"> <thead> <tr> <th colspan="3">Bit Data</th> <th>RAM Selections</th> </tr> <tr> <th>bit 10</th> <th>bit9</th> <th>bit8</th> <th></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>DATA RAM 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>DATA RAM 1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>DATA RAM 2</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>DATA RAM 3</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>PROGRAM RAM</td> </tr> </tbody> </table>	bit 15	Add Mode Selections	0	Address Add 0	1	Address Add 1	Bit Data			RAM Selections	bit 10	bit9	bit8		0	0	0	DATA RAM 0	0	0	1	DATA RAM 1	0	1	0	DATA RAM 2	0	1	1	DATA RAM 3	1	0	0	PROGRAM RAM
bit 15	Add Mode Selections																																		
0	Address Add 0																																		
1	Address Add 1																																		
Bit Data			RAM Selections																																
bit 10	bit9	bit8																																	
0	0	0	DATA RAM 0																																
0	0	1	DATA RAM 1																																
0	1	0	DATA RAM 2																																
0	1	1	DATA RAM 3																																
1	0	0	PROGRAM RAM																																
Flag	T0 ; becomes 1.**																																		
Comments	<p>* [MCx(x=0 ~ 3)] selects DATA RAM x(x=0~3). PR selects PROGRAM RAM</p> <p>**When the END signal informing you that transfer end from outside has been entered, T0; becomes 0.</p> <p>Designating address-add adds an add number after the command and becomes DMAH0~DMAH1.</p> <p>Add number is 1 when address add number designation is omitted.</p> <p>The transfer source address is set in advance to RA0 and the transfer destination RAM address is set in advance to CTx.</p>																																		

DMAH [RAM],D0,Simm		DMA Transfer (RAM → D0[31-0]) by HOLD Status																																																									
Operation Description	RAM data is transferred to D0[31-0]. The external address register and transfer word number register save the value when transfer starts. The transfer word number register is a register for storing the transfer word number in long word units. This word number is either 0 or transfer ends when forced to end.																																																										
Label	DMA H [Source],D0,Counter Source = Mo ~ M3 *																																																										
Instruction Code	<div style="text-align: center;"> </div> <table border="1" style="display: inline-table; margin-right: 20px;"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Add Mode Selections</th> </tr> <tr> <th>bit17</th> <th>bit16</th> <th>bit15</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>Address Add 0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>Address Add 1</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>Address Add 2</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>Address Add 4</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>Address Add 8</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>Address Add 16</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>Address Add 32</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>Address Add 64</td></tr> </tbody> </table> <table border="1" style="display: inline-table;"> <thead> <tr> <th colspan="2">Bit Data</th> <th rowspan="2">Selections</th> </tr> <tr> <th>bit9</th> <th>bit8</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>DATA RAM 0</td></tr> <tr><td>0</td><td>1</td><td>DATA RAM 1</td></tr> <tr><td>1</td><td>0</td><td>DATA RAM 2</td></tr> <tr><td>1</td><td>1</td><td>DATA RAM 3</td></tr> </tbody> </table>			Bit Data			Add Mode Selections	bit17	bit16	bit15	0	0	0	Address Add 0	0	0	1	Address Add 1	0	1	0	Address Add 2	0	1	1	Address Add 4	1	0	0	Address Add 8	1	0	1	Address Add 16	1	1	0	Address Add 32	1	1	1	Address Add 64	Bit Data		Selections	bit9	bit8	0	0	DATA RAM 0	0	1	DATA RAM 1	1	0	DATA RAM 2	1	1	DATA RAM 3
Bit Data			Add Mode Selections																																																								
bit17	bit16	bit15																																																									
0	0	0	Address Add 0																																																								
0	0	1	Address Add 1																																																								
0	1	0	Address Add 2																																																								
0	1	1	Address Add 4																																																								
1	0	0	Address Add 8																																																								
1	0	1	Address Add 16																																																								
1	1	0	Address Add 32																																																								
1	1	1	Address Add 64																																																								
Bit Data		Selections																																																									
bit9	bit8																																																										
0	0	DATA RAM 0																																																									
0	1	DATA RAM 1																																																									
1	0	DATA RAM 2																																																									
1	1	DATA RAM 3																																																									
Flag	T0 ; becomes 1.**																																																										
Comments	<p>* [MCx(x=0 ~ 3)] selects DATA RAM x(x=0~3).</p> <p>**When the END signal informing you that transfer end from outside has been entered, T0; becomes 0.</p> <p>Designating address-add adds an add number after the command and becomes DMAH0~DMAH64.</p> <p>Add number is 1 when address add number designation is omitted.</p> <p>The transfer source RAM address is set in advance to CTx and the transfer destination address is set in advance to WA0.</p>																																																										



DMA [RAM],D0,[s]		DMA Transfer (RAM → D0[31-0]) by HOLD Status																																																																																																																																																															
Operation Description	[s] data designated by bit0~2 is treated as transfer counter, and only numbers displayed transfer RAM data to D0[31-0] data. External address register and transfer word number register save the value when starting transfer. The transfer word number register stores transfer words in long word units. This word number becomes 0 or transfer ends when forced to end.																																																																																																																																																																
Label	DMAH [Source],DO,[Counter] Counter = M0 ~ M3 *,MC0~MC3* Source = M0~M3 *,PR*																																																																																																																																																																
Instruction Code	<table border="1"> <thead> <tr> <th colspan="3">Bit Data</th> <th rowspan="2">Add Mode Selections</th> <th colspan="3">Bit Data</th> <th rowspan="2">[s] Selections</th> </tr> <tr> <th>bit17</th> <th>bit16</th> <th>bit15</th> <th>bit 2</th> <th>bit1</th> <th>bit 0</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Address Add 0</td> <td>0</td> <td>0</td> <td>0</td> <td>Data RAM 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Address Add 1</td> <td>0</td> <td>0</td> <td>1</td> <td>Data RAM 1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Address Add 2</td> <td>0</td> <td>1</td> <td>0</td> <td>Data RAM 2</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Address Add 4</td> <td>0</td> <td>1</td> <td>1</td> <td>Data RAM 3</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>Address Add 8</td> <td>1</td> <td>0</td> <td>0</td> <td>Data RAM 0,CT0++</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>Address Add 16</td> <td>1</td> <td>0</td> <td>1</td> <td>Data RAM 1,CT1++</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>Address Add 32</td> <td>1</td> <td>1</td> <td>0</td> <td>Data RAM 2,CT2++</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>Address Add 64</td> <td>1</td> <td>1</td> <td>1</td> <td>Data RAM 3,CT3++</td> </tr> </tbody> </table> <table border="1"> <tr> <td colspan="17">b31</td> <td colspan="3">17</td> <td colspan="3">7</td> <td colspan="3">0</td> </tr> <tr> <td>1</td><td>1</td><td>0</td><td>0</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>x</td><td>x</td><td>x</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>x</td><td>x</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>x</td><td>x</td><td>x</td> </tr> </table> <table border="1"> <thead> <tr> <th colspan="2">Bit Data</th> <th rowspan="2">RAM Selections</th> </tr> <tr> <th>bit9</th> <th>bit8</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>DATA RAM 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>DATA RAM 1</td> </tr> <tr> <td>1</td> <td>0</td> <td>DATA RAM 2</td> </tr> <tr> <td>1</td> <td>1</td> <td>DATA RAM 3</td> </tr> </tbody> </table>			Bit Data			Add Mode Selections	Bit Data			[s] Selections	bit17	bit16	bit15	bit 2	bit1	bit 0	0	0	0	Address Add 0	0	0	0	Data RAM 0	0	0	1	Address Add 1	0	0	1	Data RAM 1	0	1	0	Address Add 2	0	1	0	Data RAM 2	0	1	1	Address Add 4	0	1	1	Data RAM 3	1	0	0	Address Add 8	1	0	0	Data RAM 0,CT0++	1	0	1	Address Add 16	1	0	1	Data RAM 1,CT1++	1	1	0	Address Add 32	1	1	0	Data RAM 2,CT2++	1	1	1	Address Add 64	1	1	1	Data RAM 3,CT3++	b31																	17			7			0			1	1	0	0	-	-	-	-	-	-	-	-	-	-	-	-	x	x	x	1	1	1	0	0	x	x	-	-	-	-	-	-	-	-	x	x	x	Bit Data		RAM Selections	bit9	bit8	0	0	DATA RAM 0	0	1	DATA RAM 1	1	0	DATA RAM 2	1	1	DATA RAM 3
Bit Data			Add Mode Selections	Bit Data				[s] Selections																																																																																																																																																									
bit17	bit16	bit15		bit 2	bit1	bit 0																																																																																																																																																											
0	0	0	Address Add 0	0	0	0	Data RAM 0																																																																																																																																																										
0	0	1	Address Add 1	0	0	1	Data RAM 1																																																																																																																																																										
0	1	0	Address Add 2	0	1	0	Data RAM 2																																																																																																																																																										
0	1	1	Address Add 4	0	1	1	Data RAM 3																																																																																																																																																										
1	0	0	Address Add 8	1	0	0	Data RAM 0,CT0++																																																																																																																																																										
1	0	1	Address Add 16	1	0	1	Data RAM 1,CT1++																																																																																																																																																										
1	1	0	Address Add 32	1	1	0	Data RAM 2,CT2++																																																																																																																																																										
1	1	1	Address Add 64	1	1	1	Data RAM 3,CT3++																																																																																																																																																										
b31																	17			7			0																																																																																																																																										
1	1	0	0	-	-	-	-	-	-	-	-	-	-	-	-	x	x	x	1	1	1	0	0	x	x	-	-	-	-	-	-	-	-	x	x	x																																																																																																																													
Bit Data		RAM Selections																																																																																																																																																															
bit9	bit8																																																																																																																																																																
0	0	DATA RAM 0																																																																																																																																																															
0	1	DATA RAM 1																																																																																																																																																															
1	0	DATA RAM 2																																																																																																																																																															
1	1	DATA RAM 3																																																																																																																																																															
Flag	T0 ; becomes 1. ** CTx(x=0~3) ; incremented when b2=1. No changes when b2=0.																																																																																																																																																																
Comments	* [MCx(x=0 ~ 3)] selects DATA RAM x(x=0~3). MCx(x=0~3) selects DATA RAM x(x=0~3), and after transfer increments CTx(x0~3). **When the END signal informing you that transfer end from outside has been entered, T0; becomes 0. Designating address-add adds an add number after the command and becomes DMAH0~DMAH64. Add number is 1 when address add number designation is omitted. The transfer source RAM address is set in advance to CTx and the transfer destination address is set in advance to WA0.																																																																																																																																																																



JMP NZ,Imm	Conditional Jump (Z=0)
Operation Description	When the Z flag is 0, jump is in accordance with address data (Imm).
Label	JMP NZ,[address]
Instruction Code	<pre> b31 25 19 7 0 1 1 0 1 -- -- 1 0 0 0 0 1 -- -- -- -- -- -- -- -- -- -- Imm Data </pre>
Flag	No change
Comments	



JMP NC,Imm		Conditional Jump (C=0)
Operation Description	When the C flag is 0, jump is in accordance with address data (Imm)	
Label	JMP NC,[address]	
Instruction Code	<div style="text-align: center;"> <p style="margin: 0;"> b31 25 19 7 0 </p> <p style="margin: 0; text-align: right;">Imm Data</p> </div>	
Flag	No change	
Comments		



JMP T0,Imm		Conditional Jump (T0=1)
Operation Description	When the T0 flag is 1, jump is in accordance with address data (Imm).	
Label	JMP T0,[address]	
Instruction Code	<p style="text-align: center;"> b31 25 19 7 0 </p> <p style="text-align: right; margin-right: 50px;">Imm Data</p>	
Flag	No change	
Comments		

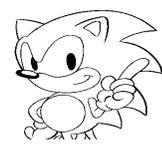
APPENDIX

This appendix contains a list of SCU register address maps.

SEGA Confidential

Register names are shown in parenthesis.

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
25FE0000(D09F)																																
25FE0004(D00W)																																
25FE0008(D00C)																																
25FE000C(D00A0)																																
25FE0010(D00EN)																																
25FE0014(D00M0)																																
25FE0018(-)																																
25FE001C(-)																																
25FE0020(O1R)																																
25FE0024(O1W)																																
25FE0028(O1C)																																
25FE002C(O1A0)																																
25FE0030(D1EN)																																
25FE0034(D1M0)																																
25FE0038(-)																																
25FE003C(-)																																
25FE0040(O2R)																																
25FE0044(D2W)																																
25FE0048(O2C)																																
25FE004C(O2A0)																																
25FE0050(O2EN)																																
25FE0054(D2M0)																																
25FE0058(-)																																
25FE005C(-)																																
25FE0060(DS TP)																																
25FE0064(-)																																
25FE0068(-)																																
25FE006C(-)																																
25FE0070(-)																																
25FE0074(-)																																
25FE0078(-)																																
25FE007C(DS TA)																																



SECRET

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
25FE0080(PPAF)																																		
25FE0084(PPD)	PD31	PD30	PD29	PD28	PD27	PD26	PD25	PD24	PD23	PD22	PD21	PD20	PD19	PD18	PD17	PD16	PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0		
25FE0088(PDA)																									RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0		
25FE008C(PDD)	RD31	RD30	RD29	RD28	RD27	RD26	RD25	RD24	RD23	RD22	RD21	RD20	RD19	RD18	RD17	RD16	RD15	RD14	RD13	RD12	RD11	RD10	RD9	RD8	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0		
25FE0090(TOC)																							TOC9	TOC8	TOC7	TOC6	TOC5	TOC4	TOC3	TOC2	TOC1	TOC0		
25FE0094(T1S)																								T198	T197	T196	T195	T194	T193	T192	T191	T190		
25FE0098(T1MD)																								T1MD									TENB	
25FE009C(-)																																		
25FE00A0(IMS)																																		
25FE00A4(IST)	IST31	IST30	IST29	IST28	IST27	IST26	IST25	IST24	IST23	IST22	IST21	IST20	IST19	IST18	IST17	IST16							IMS9	IMS8	IMS7	IMS6	IMS5	IMS4	IMS3	IMS2	IMS1	IMSO		
25FE00A8(-)																																		
25FE00AC(-)																																		
25FE00B0(ASR0)	A0PRO	A0WPC	A0RPC	A0EWT	A0BWS	A0BW2	A0BW1	A0BW0	A0NW3	A0NW2	A0NW1	A0NW0	A0LN1	A0LN0	A0SZ	A1PRO	A1WPC	A1RPC	A1EWT	A1BW3	A1BW2	A1BW1	A1BW0	A1NW3	A1NW2	A1NW1	A1NW0	A1LM1	A1LM0	A1SZ				
25FE00B4(ASR1)	A2PRO	A2WPC	A2RPC	A2EWT													A2SZ	A3WPC	A3RPC	A3EWT	A3BW3	A3BW2	A3BW1	A3BW0	A3NW3	A3NW2	A3NW1	A3NW0	A3LM1	A3LM0	A3SZ			
25FE00B8(AREF)																																		
25FE00BC(AIAK)																																		
25FE00C0(-)																																		
25FE00C4(RSEL)																																		
25FE00C8(VER)																																		
25FE00CC(-)																																		

Acronym	Address	Bit	Description
AIACK	25FE00A8h	0	A-Bus interrupt acknowledge output valid bit (=0: invalid / =1: valid)
ARFEN	25FE00B8h	4	A-Bus refresh output valid bit (=0: invalid/=1: valid)
ARWT3-0	25FE00B8h	3 - 0	A-Bus refresh wait number
A0BW3-0	25FE00B0h	27 - 24	CS0 space, burst cycle wait number set bit
A0EWT	25FE00B0h	28	CS0 space, external wait effective bit (=0: invalid/=1: valid)
A0LN1-0	25FE00B0h	19 - 18	CS0 space, burst length set bit
A0PRD	25FE00B0h	31	CS0 space, previous read effective bit (=0: invalid/=1: valid)
A0RPC	25FE00B0h	29	CS0 space, pre-charge insert bit after read
A0NW3-0	25FE00B0h	23 - 20	CS0 space, normal cycle wait number set bit
A0SZ	25FE00B0h	16	CS0 space, bus size set bit
A0WPC	25FE00B0h	30	CS0 space, pre-charge insert bit after write
A1BW3-0	25FE00B0h	11 - 8	CS1 space, burst cycle wait number set bit
A1EWT	25FE00B0h	12	CS1 space, external wait effective bit (=0: invalid/=1: valid)
A1LN1-0	25FE00B0h	3 - 2	CS1 space, burst length set bit
A1NW3-0	25FE00B0h	7 - 4	CS1 space, normal cycle wait number set bit
A1PRD	25FE00B0h	15	CS1 space, previous read effective bit (=0: invalid/=1: valid)
A1RPC	25FE00B0h	13	CS1 space, pre-charge insert bit after read
A1SZ	25FE00B0h	0	CS1 space, bus size set bit
A1WPC	25FE00B0h	14	CS1 space, pre-charge insert bit after write
A2EWT	25FE00B4h	28	CS2 space, external wait effective bit (=0: invalid/=1: valid)
A2LN1-0	25FE00B4h	19 - 18	CS2 space, burst length set bit
A2PRD	25FE00B4h	31	CS2 space, previous read effective bit (=0: invalid/=1: valid)
A2RPC	25FE00B4h	29	CS2 space, pre-charge insert bit after read
A2SZ	25FE00B4h	16	CS2 space, bus size set bit
A2WPC	25FE00B4h	30	CS2 space, pre-charge insert bit after write
A3BW3-0	25FE00B4h	11 - 8	Dummy space, burst cycle wait number set bit
A3EWT	25FE00B4h	12	Dummy space, external wait effective bit (=0: invalid/=1: valid)
A3LN1-0	25FE00B4h	3 - 2	Dummy space, burst length set bit
A3NW3-0	25FE00B4h	7 - 4	Dummy space, normal cycle wait number set bit
A3PRD	25FE00B4h	15	Dummy space, previous read effective bit (=0: invalid/=1: valid)
A3RPC	25FE00B4h	13	Dummy space, pre-charge insert bit after read
A3SZ	25FE00B4h	0	Dummy space, bus size set bit
A3WPC	25FE00B4h	14	Dummy space, pre-charge insert bit after write
C	25FE0080h	20	DSP program control port, Carry flag
DACSA	25FE007Ch	20	DMA A-Bus Access Flag (=0: no access/=1: access)
DACSB	25FE007Ch	21	DMA B-Bus Access Flag (=0: no access/=1: access)
DACSD	25FE007Ch	22	DMA DSP-Bus Access Flag (=0: no access/=1: access)
DDMV	25FE007Ch	0	DSP side DMA operate flag (=0: stop/=1: operate)
DDWT	25FE007Ch	1	DSP side DMA standby flag (=0: stop/=1: standby)
DSTOP	25FE0060h	0	DMA force-stop bit (=0: DMA operable/=1: DMA force stop)
D0BK	25FE007Ch	16	DMA level 0 interrupt flag (=0: stop/=1: interrupt)
D0C19-0	25FE0008h	19 - 0	DMA level 0 transfer byte number
D0EN	25FE0010h	8	DMA level 0 enable bit (=0: Disable/=1: Enable)



Acronym	Address	Bit	Description
D0FT2-0	25FE0014h	2 - 0	DMA level 0 starting factor selection bit =000B: V-Blank-IN receive and enable bit set =001B: V-Blank-OUT receive and enable bit set =010B: H-Blank-IN receive and enable bit set =011B: Timer 0 receive and enable bit set =100B: Timer 1 receive and enable bit set =101B: Sound Req receive and enable bit set =110B: Sprite draw end and enable bit set =111B: DMA start bit set and enable bit set
D0GO	25FE0010h	0	DMA level 0 start bit (=0: stop/=1: start)
D0MOD	25FE0014h	24	DMA level 0 mode bit (=0: direct mode/=1: indirect mode)
D0MV	25FE007Ch	4	DMA level 0 operating flag (=0: stop/=1: start)
D0RA	25FE000Ch	8	DMA level 0 read address add value (=0: no add/=1: adds 4 byte)
D0RUP	25FE0014h	16	DMA level 0 read address update bit
D0R26-0	25FE0000h	26 - 0	DMA level 0 read address
D0WA2-0	25FE000Ch	2 - 0	DMA level 0 write address add value =000b: no addition =001b: adds 2 bytes =010b: adds 4 bytes =011b: adds 8 bytes =100b: adds 16 bytes =101b: adds 32 bytes =110b: adds 64 bytes =111b: adds 128 bytes
D0WT	25FE007Ch	5	DMA level 0 standby flag (=0: stop/=1: standby)
D0WUP	25FE0014h	8	DMA level 0 write address update bit
D0W26-0	25FE0004h	26 - 0	DMA level 0 write address
D1BK	25FE007Ch	17	DMA level 1 interrupt flag (=0: stop/=1: interrupt)
D1C11-0	25FE0028h	11 - 0	DMA level 1 transfer byte number
D1EN	25FE0030h	8	DMA level 1 enable bit (=0: Disable/=1: Enable)
D1FT2-0	25FE0034h	2 - 0	DMA level 1 starting factor selection bit =000B: V-Blank-IN receive and enable bit set =001B: V-Blank-OUT receive and enable bit set =010B: H-Blank-IN receive and enable bit set =011B: Timer 0 receive and enable bit set =100B: Timer 1 receive and enable bit set =101B: Sound Req receive and enable bit set =110B: Sprite draw end and enable bit set =111B: DMA start bit set and enable bit set
D1GO	25FE0030h	0	DMA level 1 start bit (=0: stop/=1: start)
D1MOD	25FE0034h	24	DMA level 1 mode bit (=0: direct mode/=1: indirect mode)
D1MV	25FE007Ch	8	DMA level 1 operating flag (=0: stop/=1: start)
D1RA	25FE002Ch	8	DMA level 1 read address add value (=0: no add/=1: adds 4 bytes)
D1RUP	25FE0034h	16	DMA level 1 read address update bit
D1R26-0	25FE0020h	26 - 0	DMA level 1 read address

Acronym	Address	Bit	Description
D1WA2-0	25FE002Ch	2 - 0	DMA level 1 write address add value =000 _B : does not add =001 _B : adds 2 bytes =010 _B : adds 4 bytes =011 _B : adds 8 bytes =100 _B : adds 16 bytes =101 _B : adds 32 bytes =110 _B : adds 64 bytes =111 _B : adds 128 bytes
D1WT	25FE007Ch	9	DMA level 1 standby flag (=0: stop/=1: standby)
D1WUP	25FE0034h	8	DMA level 1 write address update bit
D1W26-0	25FE0024h	26 - 0	DMA level 1 write address
D2C11-0	25FE0048h	11 - 0	DMA level 2 transfer byte number
D2EN	25FE0050h	8	DMA level 2 enable bit (=0: Disable/=1: Enable)
D2FT2-0	25FE0054h	2 - 0	DMA level 2 starting factor selection bits =000 _B : V-Blank-IN receive and enable bit set =001 _B : V-Blank-OUT receive and enable bit set =010 _B : H-Blank-IN receive and enable bit set =011 _B : Timer 0 receive and enable bit set =100 _B : Timer 1 receive and enable bit set =101 _B : Sound Req receive and enable bit set =110 _B : Sprite draw end and enable bit set =111 _B : DMA start bit set and enable bit set
D2G0	25FE0050h	0	DMA level 2 start bit (=0: stop/=1: operation)
D2MOD	25FE0054h	24	DMA level 2 mode bit (=0: direct mode/=1: indirect mode)
D2MV	25FE007Ch	12	DMA level 2 operation flag (=0: stop/=1: operation)
D2RA	25FE004Ch	8	DMA level 2 read address add value (=0: no add/=1: adds 4 bytes)
D2RUP	25FE0054h	16	DMA level 2 read address update bit
D2R26-0	25FE0040h	26 - 0	DMA level 2 read address
D2WA2-0	25FE004Ch	2 - 0	DMA level 2 write address add value =000 _B : no addition =001 _B : adds 2 bytes =010 _B : adds 4 bytes =011 _B : adds 8 bytes =100 _B : adds 16 bytes =101 _B : adds 32 bytes =110 _B : adds 64 bytes =111 _B : adds 128 bytes
D2WT	25FE007Ch	13	DMA level 2 standby flag (=0: stop/=1: standby)
D2WUP	25FE0054h	8	DMA level 2 write address update bit
D2W26-0	25FE0044h	26 - 0	DMA level 2 write address
E	25FE0080h	18	DSP Program control port, Program end interrupt flag
EP	25FE0080h	25	DSP Program control port, Temporary stop execution flag during program execution (=0: don't execute / =1: execute)



Acronym	Address	Bit	Description
ES	25FE0080H	17	DSP Program Control Port, Program Step Execution Control Bit (=0: don't execute / =1: execute)
EX	25FE0080H	16	DSP Program Control Port, Program Execution Control Bit (=0: don't execute / =1: execute)
IMS0	25FE00A0H	0	V-Blank-IN Interrupt Mask Bit
IMS1	25FE00A0H	1	V-Blank-OUT Interrupt Mask Bit
IMS2	25FE00A0H	2	H-Blank-IN Interrupt Mask Bit
IMS3	25FE00A0H	3	Timer 0 Interrupt Mask Bit
IMS4	25FE00A0H	4	Timer 1 Interrupt Mask Bit
IMS5	25FE00A0H	5	DSP End Interrupt Mask Bit
IMS6	25FE00A0H	6	Sound Request Interrupt Mask Bit
IMS7	25FE00A0H	7	SMPC Interrupt Mask Bit
IMS8	25FE00A0H	8	PAD Interrupt Mask Bit
IMS9	25FE00A0H	9	Level 2-DMA End Interrupt Mask Bit
IMS10	25FE00A0H	10	Level 1-DMA End Interrupt Mask Bit
IMS11	25FE00A0H	11	Level 0-DMA End Interrupt Mask Bit
IMS12	25FE00A0H	12	DMA Illegal Interrupt Mask Bit
IMS13	25FE00A0H	13	Sprite Draw End Interrupt Mask Bit
IMS15	25FE00A0H	15	A-Bus Interrupt Mask Bit
IST0	25FE00A4H	0	V-Blank-IN Interrupt Status Bit
IST1	25FE00A4H	1	V-Blank-OUT Interrupt Status Bit
IST2	25FE00A4H	2	H-Blank-IN Interrupt Status Bit
IST3	25FE00A4H	3	Timer 0 Interrupt Status Bit
IST4	25FE00A4H	4	Timer 1 Interrupt Status Bit
IST5	25FE00A4H	5	DSP End Interrupt Status Bit
IST6	25FE00A4H	6	Sound request Interrupt Status Bit
IST7	25FE00A4H	7	SMPC Interrupt Status Bit
IST8	25FE00A4H	8	PAD Interrupt Status Bit
IST9	25FE00A4H	9	Level 2-DMA End Interrupt Status Bit
IST10	25FE00A4H	10	Level 1-DMA End Interrupt Status Bit
IST11	25FE00A4H	11	Level 0-DMA End Interrupt Status Bit
IST12	25FE00A4H	12	DMA Illegal Interrupt Status Bit
IST13	25FE00A4H	13	Sprite Draw End Interrupt Status Bit
IST31-16	25FE00A4H	31-16	Outside Interrupt 15-0 Status Bit
LE	25FE0080H	15	DSP Program Control Port, Program Counter Load Enable Bit (=0: no execute/=1: execute)
PD31-0	25FE0084H	31 - 0	DSP Program RAM Data Port
PR	25FE0080H	26	DSP Program Control Port, Pause Cancel Flag while program is executing (=0: no execute/=1: execute)
P7-0	25FE0080H	7 - 0	DSP Program RAM Address
RA7-0	25FE0088H	7 - 0	DSP Data RAM Address
RD31-0	25FE0080H	31 - 0	DSP Data RAM Data Port
RSEL	25FE00C4H	0	SDRAM Selection Bit (=0: 2 Mbit x 2 / =1: 4 Mbit x 2)
S	25FE0080H	22	DSP Program Control Port, Sine Flag

TENB	25FE0098h	0	Timer Enable Bit (=0: OFF / =1: ON)
T0	25FE0080h	23	DSP Program Control Port, D0 Bus Use DMA Execute Flag
T0C9-0	25FE0090h	9 - 0	Timer 0 Compare Data
T1MD	25FE0098h	8	Timer 1 ModeBit =0: occurs at each line =1: occurs only at lines indicated by Timer 0
T1S8-0	25FE0094h	8 - 0	Timer 1 Set Data
V	25FE0080h	19	DSP Program Control Port, Overflow Flag
VER3-0	25FE00C8h	3 - 0	SCU Chip Version Number
Z	25FE0080h	21	DSP Program Control Port, Zero Flag

SEGA Confidential



INDEX

Numbers within () shows the page of the “First” heading.

Numeric

1 command Repeat Execution.....	89
---------------------------------	----

Alphabetic

A-Bus	ii
A-Bus Control Register	61
A-Bus Interrupt Acknowledge	61
A-Bus Interrupt Acknowledge Register	61
A-Bus Interrupt Acknowledge Map	14
A-Bus Refresh Register	13, 71
A-Bus Refresh Register Map	13
A-Bus Refresh Wait Number	71
A-Bus Set Register (CS0, 1 spaces)	62
A-Bus Set Register (CS2 and dummy spaces)	62
A-Bus Set Register Map	13
Access, Interrupt, Standby, Operation Registers	47
B-Bus	(ii)
Blanking Interrupt	29
Block Diagram	3
Commands.....	91
Commands (1), List of.....	80
Commands (2), List of	81
Commands (3), List of.....	82
Commands (4), List of.....	83
Constants, Description of	90
CS0 Space Burst Cycle Set Value	65
CS0 Space Burst Length Set Value	65
CS0 Space Bus Size Set Value	66
CS0 Space Single Cycle Set Value	65
CS0, 1 Space A-Bus Set Set Register	62
CS1 Space Burst Cycle Set Value	67
CS1 Space Burst Length Set Value	68
CS1 Space Bus Size Set Value	68
CS1 Space Single Cycle Set Value	67
CS2 Space Burst Cycle Value	68
CS2 Space Bus Size Set Value	70

Data	ii
Data Write Example (Indirect Mode).....	23
Difference in DMA operation by Address Renewal Bit	22
Difference in Timing by Setting External Wait Effective Bit	64
Direct Mode DMA Transfer Operation	18
DMA Enable Register	45
DMA Command Execution	87
DMA Command Format 1	132
DMA Command Format 2	132
DMA Control Register	41
DMA End Interrupt	33
DMA Force-Stop Register	47
DMA Force-Stop Register Map	8
DMA Illegal Interrupt	33
DMA Mode	18
DMA Mode, Address Renewal, Start Factor Select Register	46
DMA Status Register	47, 48
DMA Status Register Map	9
DMA Transfer (Basic Operation)	16
DMA Transfer Execution by Address Add Value Set	26
DMA Transferable Area when Started from DSP	17
DMA Transferable Area when Started from Main CPU	17
DMA Write Address while Stopped	46
DSP	34
DSP Control Port.....	51
DSP Data RAM Address Port	10, 53
DSP Data RAM Address Port Map	10
DSP Data RAM Data Port	54
DSP Data RAM Data Port Map	10
DSP End Interrupt	33
DSP Program Control Port	9
DSP Program Load Step 1	34
DSP Program Load Step 2	35
DSP Program Load Step 3	35
DSP Program RAM Data Port	10, 53
DSP Program RAM Data Port Map	10
Dummy Space Burst Cycle Set Value	71
Dummy Space Burst Length Set Value	71
Dummy Space Bus Size Set Value	72
Dummy Space Single Cycle Set Value	71



Example of transfer between SCU and Processor	44
Features of Data Transfer to DSP from D0 Bus	87, 88
High/Low Level DMA Operation	48
Indirect Mode DMA Transfer	20
Indirect Mode DMA Transfer Flow	19
Interrupt Control Register	57
Interrupt Factor	27
Interrupt Factor, General Names	28
Interrupt Mask Register	57
Interrupt Mask Register Map	12
Interrupt Status Register	58
Interrupt Status Register Contents	59
Interrupt Status Register Map	12
Jump Command Execution	85
Jump Command Format	141
Level 0 Transfer Byte Number	42
Level 2-0 Address Add Value	42
Level 2-0 DMA Authorization Bit	45
Level 2-0 DMA Mode, Address Renewal, Start Factor Select Register	46
Level 2-0 DMA Set Register Map	8
Level 2-0 Read Address	41
Level 2-0 Write Address	41
Level 2-1 Transfer Byte Number	42
Load Immediate Command Format 1 (unconditional transfer)	120
Load Immediate Command Format 2 (conditional transfer)	120
Loop Bottom Command Format	153
Loop Program Execution	86
Main CPU	i
Operand Execution Method	85
Operation Command Format	91
Operation when Cache Hit	5
PAD Interrupt	33
RAM Page Select	53
Read Address Add Value	43
Registers, List of	40
Results of Previous Read Process	63

SCSP	i
SCU	i
SCU Control Register	73
SCU Mapping (Cache_address).....	4
SCU Mapping (Cache_through_address)	6
SCU Overview	2
SCU SDRAM Select Register Map	14
SCU SDRAM Select Bit	73
SCU Version Register Map	14
SCU Version Register	73
SMPC	ii
SMPC Interrupt	33
Sound Request Interrupt	33
Special Process Execution	89
Sprite Draw End Interrupt	33
Start Factor	46
Subroutine Program Execution	90
System Configuration	2
Timer 0 Compare Register	11, 55
Timer 0 Compare Register Map	11
Timer 0 Interrupt Degree of Occurrence	30
Timer 1 Interrupt Degree of Occurrence	31
Timer 1 Mode Register	11, 56
Timer 1 Mode Register Map	11
Timer 1 Occurrence Select Content	56
Timer 1 Set Data Register	11, 55
Timer 1 Set Data Register Map	11
Timer Operation Contents	56
Timer Register	55
Timing when setting pre-charge insert bit after Read	63
Timing when setting pre-charge insert bit after Write	63
Timing when Writing CS2 Burst Cycle	65
VDP1	i
VDP2	i
Work RAM Area Contents	24
Write Address Add Value	43
Write Address Add Value Indication	45



(This page is blank in the original Japanese document.)

SEGA Confidential

Commands

NOP (ALU Operation)	93
AND	94
OR	95
XOR	96
ADD	97
SUB	98
AD2	99
SR	100
RR	101
SL	102
RL	103
RL8	104
NOP (X-Bus Operation)	106
MOV [s], X	107
MOV MUL, p	108
MOV [s], P	109
NOP (Y-Bus Control)	111
MOV [s], Y	112
CLR A	113
MOV ALU, A	114
MOV [s], A	115
NOP (D1-Bus No Operation)	117
MOV SImm, [d]	118
MOV [s], [d]	119
MVI Imm, [d]	121
MVI [d], Imm, Z	122
MVI Imm, [d], NZ	123
MVI Imm, [d], S	124
MVI Imm, [d], NS	125
MVI Imm, [d], C	126
MVI Imm, [d], NC	127
MVI Imm, [d], T0	128
MVI Imm, [d], NT0	129
MVI Imm, [d], ZS	130
MVI Imm, [d], NZS	131



DMA D0 , [RAM] , SImm	133
DMA [RAM] , D0 , SImm	134
DMA , D0 , [RAM] , [s]	135
DMA [RAM] , D0 , [s]	136
DMAH , D0 , [RAM] , SImm	137
DMAH [RAM] , D0 , SImm	138
DMAH D0 , [RAM] , [s]	139
DMAH [RAM] , D0 , [s]	140
JMP Imm	142
JMP Z , Imm	143
JMP NZ , Imm	144
JMP S , Imm	145
JMP NS , Imm	146
JMP C , Imm	147
JMP NC , mm	148
JMP T0 , Imm	149
JMP NT0 , Imm	150
JMP ZS , Imm	151
JMP NZS , Imm	152
BTM	154
LPS	155
END	157
ENDI	33, 158