date: 2002/11/20

HITACHI SEMICONDUCTOR TECHNICAL UPDATE

Classification of Production	Development Environment	No	TN-OS*-065A/E	Rev	1

THEME	HI7750 Notes on using the FPU	Classification of Information	1. Spec change 2. Supplement of Documents [✓ 3.] Limitation of Use 4. Change of Mask 5. Change of Production Line

PRODUCT NAME	HS0775ITCE1SME, HS0775ITCE1SMB, HS0775ITCE1SMS, HS0775ITHE1STE, HS0775ITHE1STB, HS0775ITHE1STS, HS0775ITIE1SFE, HS0775ITIE1SFB, HS0775ITIE1SFS, HS0775ITIE1SFE-E, HS0775ITIE1SFB-E, HS0775ITIE1SFS-E, HS0775ITIE1SFU, HS0775ITIE1SFV, HS0775ITIE1SFW, HS0775ITIE1SFX, HS0775ITIE1SFY, HS0775ITIE1SFZ	Lot No. V1.0A, V1.0B, V1.0C	Reference Documents	HI7750 User's Manual (HS0775ITCE1SE) ADE-702-180 Rev.1.0	Effective Date Forever

When using the FPU on the HI7750, please note the contents on the following document.

[Attached document]
" Notes on using the FPU on the HI7750" (HI7750-NOTE-FPU-021105(E))"

HI7750-NOTE-FPU-021105(E)

Notes on Using the FPU on the HI7750

Read this document carefully before using the FPU, and take note of the points on usage described herein.

System Software Development Dept.
Semiconductor & Integrated Circuits
Hitachi, Ltd.

1. Task, Extended SVC Handler, and I/O Handler

1.1 Attributes TA_COP1, TA_COP2

The SH-4 has two FPU register banks, 0 and 1. Attributes TA_COP1 and TA_COP2 select use of FPU bank 0 and FPU bank 1, respectively.

Specify attributes TA_COP1 and TA_COP2 as described in Table 1. The FPU register bank at initiation is 0 (FPSCR.FR = 0).

Table 1 Specifying Attributes TA_COP1, TA_COP2

Case	Attribute	Remarks
Matrix calculation (both FPU register banks used)	TA_COP1|TA_COP2	
Floating-point calculation	TA_COP1	The normal floating-point calculation uses a single FPU register bank. Specification of TA_COP2 is not recommended because FPSCR.FR = 1 must be set at the top of the entry function of the task and task exception processing routine.
No floating-point calculation	TA_COP1	TA_COP1 needs not be set only when FPU = single or FPU = double is specified by the compiler option.

1.2 Compiler Options

The FPSCR at initiation is shown in 3.1, States on the Initiation of Tasks and Handlers.

When a floating-point calculation is executed, the FPU will not operate normally with these initial values and some compiler options. When any of the options in Table 2 is selected, the corresponding bit of FPSCR must be set to the value shown. This setting is made at the top of the entry function of the task and of the extended SVC handler and of the I/O handler.

Table 2 Value to be Set in FPSCR

Bits in FPSCR	Compiler Option		Value to be Set
Precision mode (FPSCR.PR)	FPU option	Double	1
		Other than double	(Not required)
Denormalization mode (FPSCR.DN)	Denormalize option	OFF	(Not required)
		ON	0
Rounding mode (FPSCR.RM)	Round option	Zero	(Not required)
		Nearest	B'00

The following shows an example for setting FPSCR in the conditions below:

[Compiler Options]

- FPU =Double
- Denormalize = ON
- Round = Nearest

```
#include <machine.h>   /* Included to use built-in function set_fpscr(). */
#define INI_FPSCR 0x00080000   /* PR=1, DN=0, SZ=0, RM=B’00 */
#pragma noregsave(Task)
void Task(INT stacd)
{
    set_fpscr(INI_FPSCR);    /* Sets FPR at the top of function.  */
    /* Task processing */
    ext_tsk();
}
```

2. Interrupt Handler, Exception Processing Routine, TRAP Routine, Cyclic Handler, Alarm Handler, System Initialization Handler, and I/O Initialization Handler

2.1 Execution of Floating-Point Calculation

When a floating-point calculation is executed by these handlers, make the required initial settings for the FPSCR at the start of these handlers, with reference to section 4.3.2 of the HI7750 User's Manual (HS0775ITCE1SE). When leaving the handler, the FPSCR must be restored.

Table 3 shows values to be set in the bits of FPSCR.

Table 3 Value to be Set in FPSCR

Bits in FPSCR	Compiler Option		Value to be Set
Precision mode (FPSCR.PR)	FPU option	Double	1
		Other than double	0
Denormalization mode (FPSCR.DN)	Denormalize option	OFF	1
		ON	0
Rounding mode (FPSCR.RM)	Round option	Zero	B'01
		Nearest	B'00
Transfer size mode (FPSCR.SZ)			0
FPU register bank (FPSCR.FR)			0
Other bits of FPSCR			0

2.2 No Execution of Floating-Point Calculation

When the compiler options satisfy the FPU conditions in table 4, the FPSCR register must be saved and restored even if the floating-point calculation is not executed.

Table 4 Conditions of Compiler Options

Compiler Option	Specification
FPU	Not specified
FPSCR	Aggressive

```
#include <machine.h>   /* Included to use built-in function set_fpscr(). */
void InterruptHandler(void)
{
    UW     old_fpscr;
    old_fpscr = get_fpscr(); /* Saves FPSCR. */
    /* Processing of the handler */
    set_fpscr(old_fpscr);    /* Restores FPSCR. */
}
```

3. Information for Reference

3.1 States on the Initiation of Tasks and Handlers

Table 5 shows the FPSCR states on the initiation of tasks and handlers

Table 5 States on the Initiation of Tasks and Handlers

State at Initiation	Task, Extended SVC Handler, and I/O Handler	Interrupt Handler, Cyclic Handler, Alarm Handler System Initialization Handler, and I/O Initialization Handler	Exception Processing Routine and TRAP Routine
Value of FPSCR	H'00040001 (undefined if TA_COP1 nor TA_COP2 is specified)	Undefined	Same as before the exception occurred
Precision mode (FPSCR.PR)	Single precision (0)		
Denormalization mode (FPSCR.DN)	Handled as 0 (1)		
Rounding mode (FPSCR.RM)	Rounded to 0 (B'01)		
Transfer size mode (FPSCR.SZ)	32 bits (0)		
FPU regsiter bank(FPSCR.FR)	Bank 0 (0)		
Other bits of FPSCR	0		

3.2 FPSCR Structure of SH-4

31 22	21	20	19	18	17 12	11 7	6 2	1 0
Reserved	FR	SZ	PR	DN	Cause	Enable	Flag	RM

bit		Meaning		
21	FR	FPU register bank	0	Bank 0
			1	Bank 1
20	SZ	Transfer size mode	0	The data size of the FMOV instruction is 32 bits.
			1	The data size of the FMOV instruction is a 32-bit register pair (64 bits).
19	PR	Precision mode	0	Single precision
			1	Double precision
18	DN	Denormalization mode	0	A denormalized number is treated as such.
			1	A denormalized number is treated as zero.
17-12	Cause	FPU exception cause field		
11-7	Enable	FPU exception enable field		
6-2	Flag	FPU exception flag field		
1,0	RM	Rounding mode	B'00	Round to Nearest
			B'01	Round to Zero

3.3 Handling by the Compiler

This section explains handling of the FPU by V5.1, V6, and V7.1 of the compiler. The compiler never generates any object code to change the FPSCR when "Single" or "Double" has been specified as the FPU option.

(1) FPSCR.PR (Precision mode)

Table 6 Handling of the FPSCR.PR Bit by the Compiler

Compiler Option		Precision Mode Assumed by the Compiler on Entry to Functions (FPSCR.PR Bit)*1	Precision Mode at the End of the Function*2	Remarks
FPU option	FPSCR option *3			
Single	(Specification disabled)	Single precision (0)	Single precision (0)	The compiler does not generate any object code to change the PR bit.
Double	(Specification disabled)	Double precision (1)	Double precision (1)	
No specification (Mix)	Safe	Single precision (0)	Single precision (0)	
	Aggressive	Single precision (0)	Undefined	

Note: *1 The compiler assumes this precision mode in generating code at the top of the function.
*2 The compiler generates code to select this precision mode at the end of the function.
*3 Compiler V5.1 does not support this FPSCR option; treatment is the same as ‘aggressive’.

(2) FPSCR.DN (Denormalization mode)

Table 7 Handling of the FPSCR.DN Bit by the Compiler

Compiler Option	Denormalization Mode Assumed by the Compiler (FPSCR.DN Bit)*	Remarks
Denormalize Option		
OFF	A denormalized number is treated as zero. (1)	The compiler does not generate any object code to change the DN bit.
ON	A denormalized number is treated as such. (0)	

Note: *1 The compiler assumes this denormalization mode in generating code at the top of the function.

(3) FPSCR.RM (Rounding mode)

Table 8 Handling of the FPSCR.RM Bits by the Compiler

Compiler Option	Rounding Mode Assumed by the Compiler (FPSCR.RM Bit)*	Remarks
Round Option		
Zero	Round to Zero (B’01)	The compiler does not generate any object code to change the RM bits.
Nearest	Round to Nearest (B’00)	

Note: *1 The compiler assumes this rounding mode in generating code at the top of the function.

(4) FPSCR.SZ (transfer size mode)

The compiler always assumes SZ = 0 (the data size of the FMOV instruction is 32 bits.) and does not generate any object code to change the SZ bit.

(5) FPSCR.FR (FPU register bank)

The compiler does not generate any object code to change the FR bit.

However, in the built-in functions st_ext() and ld_ext(), the FR bit is temporarily changed within these function. The value of the FR bit on return from these function is the same as the value when the function was called.