To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

HITACHI SEMICONDUCTOR TECHNICAL UPDATE

DATE	9th April, 1999	No.	TN-SH7-142 A/E
THEME	FPU double precision instruction		
CLASSIFICATION	☐ Spec. change ☐ Supplement of Documents	☑ Limitation on Use	
PRODUCT NAME	HD6417091BP200, HD6417091RBP200, HD6417750BP200, HD6417750F167, HD6417750VF128		
REFERENCE DOCUMENTS	SH7091 Hardware Manual, SH7091 Programming Manual, SH7750 Hardware Manual, SH7750 Programming Manual	Effective Date	eternity
		From	

1. Abstract

Double precision denormalized numbers in the FDIV, FADD, FSUB or FMUL instructions give wrong results.

1.1 Target

Software designers who develop software in scientific and engineering field are the target. Especially, those designers who use double precision floating point instructions and treat the denormalized number as it is. But this concerning target is not for those designers who use double precision floating point instructions and treat zero-flushed denormalized number and also not for those designers who use single precision floating point instruction as well.

1.2 Bug

There are two bugs. First, when the inputs are denormalized numbers as described in (a) and (b), then wrong results are generated. And second one, when the inputs are denormalized numbers and qNaN as described in (C), then wrong results are generated :

(a)In double-precision FDIV, there are cases that the wrong results generates as 0 or infinity when inputs include denormalized number.

(b)In double-precision FMUL, there are cases that the wrong result generates as infinity when inputs include denormalized number.

(c)In double-precision FDIV,FADD,FSUB, or FMUL, there are cases that exception occurs by mistake and the wrong result generates as infinity when inputs are denormalized number and qNaN.

1.3 Bug Effect

The worst case is, when double precision FDIV or FMUL instruction with denormalized number's input is used the wrong result is written in register. Particularly, denormalized number / denormalized number=0, denormalized number / 0 =0 is not appropriate in mathematical sense.

2. Countermeasure

[1] is recommended. [2] is desirable when there are needs to calculate denormalized numbers in scientific and engineering fields.

[1]Double precision instruction is only used at the mode "FPSCR.DN=1" which means denormalized number is treated as zero. The performance does not decrease in this work around.

[2] The case that wrong results are generated when inputs are denormalized numbers(a,b) should be modified by software. Please refer to section 4. *Software modification* in detail.

(i) Source and destination register(DRn) should be saved.

(ii)When the result is 0 or infinity in double-precision FDIV, the function that calculate denormalized numbers is called. The function should be prepared by software designer.

The case that wrong results are generated when inputs are denormalized numbers and qNaN(c) should be modified by TRAP routine. Please refer to section 5. *TRAP routine modification* in detail.

(i)When one of the input is denormalized numbers and the other is qNaN in double-precision FDIV,FADD,FSUB,or FMUL, qNaN(64h'7ff7ffff_ffffffff) is written in destination register by the TRAP routine.

3. Details

3.1 Definition

Data pattern that cause bugs is defined. The data patterns from (A) to (D) in tables correspond to the following patterns.

(A)Double precision denormalized number
64h'00000000_XXXXXXXX or 64h'80000000_XXXXXXXX (X:0 or 1)
condition:32h'XXXXXXXX!=32h'00000000

(B)Double precision denormalized number
64h'000YYYYY_XXXXXXXX or 64h'800YYYYY_XXXXXXXX (X:0 or 1)
condition: 20h'YYYYY!=20h'00000

(C)Double precision qNaN
64h'7ff00000_XXXXXXXX or 64h'fff00000_XXXXXXXX (X:0 or 1)
condition: 32h'XXXXXXXX!=32h'00000000

(C)Double precision qNaN *unlimited qNaN
64h'7ffXXXXX_XXXXXXXX or 64h'fffXXXXX_XXXXXXXX (X:0 or 1)
condition: 52h'XXXXX_XXXXXXXX!=52h'00000_00000000

3.2 Bugs List

Table1 shows the combination of instruction and data that generate wrong results in case of FPSCR.DN=1'b0(denormalized numbers is treated as it is).

The inputs (A),(B), and (C) are the data patterns defined in section 3.1.

The NG types from (1) to (7) is classified from the wrong results category and these NG types is used in Table2,3,4.

In NG types (1), (2), (3) and (7), the output 0 or infinity is the wrong result.

In NG types (4), (5) and (6), if exception TRAP(FPU Error) occurs then qNaN is not written .

The relation between section 1.2 and Table1 is shown:

(a) corresponds to (1),(2),and (3)
(b) corresponds to (7)
(c) corresponds to (4),(5),and (6)

Table1 NG results

NG type	Instruction	Inputs		SH-4	Correct
		DRm	DRn		
(1)	FDIV	+0/-0	(A)DENORM	+0/-0	DZ
(2)	FDIV	(A)DENORM	+0/-0	+0/-0	FPU Error
		(A)DENORM	(A)DENORM		
(3)	FDIV	(A)DENORM	+INF/-INF	+INF/-INF	FPU Error
(4)	FDIV	(C)qNaN	(A)DENORM	FPU Error	qNaN*
		(C)qNaN	(B) DENORM		
		(B) DENORM	(C)qNaN		
(5)	FADD/FSUB	(C)qNaN	DENORM	FPU Error	qNaN*
		DENORM	(C)qNaN		
(6)	FMUL	(C)qNaN	(B) DENORM	FPU Error	qNaN*
		(B) DENORM	(C)qNaN		
(7)	FMUL	(A)DENORM	+INF/-INF	+INF/-INF	FPU Error
		+INF/-INF	(A)DENORM		

*qNaN:64h'7ff7ffff_ffffffff

No NG occurs in case FPSCR.DN=1'b1(denormalized numbers is flushed to zero) .

The summary of special cases in double FDIV, FADD, FSUB, and FMUL instructions are shown.

SH-4 outputs correct results in dotted box.
SH-4 outputs wrong results in white box.

Table2 FDIV DRm, DRn (DRn / DRm → DRn)

DRm \ DRn	NORM	+0	-0	+INF	-INF	(A) positive DENORM	(A) negative DENORM	(B) DENORM	(C) qNaN	(D) qNaN	sNaN
NORM	DIV	0		INF		Error					
+0	DZ	Invalid		+INF	-INF	+0 (1)	-0 (1)	DZ			
-0				-INF	+INF	-0 (1)	+0 (1)				
+INF	0	+0	-0	Invalid		Error			qNaN		Invalid
-INF		-0	+0								
(A) positive DENORM		+0(2)	-0(2)	(3) +INF	(3) -INF	+0 (2)	-0 (2)				
(A) negative DENORM		-0(2)	+0(2)	(3) -INF	(3) +INF	-0 (2)	+0 (2)				
(B) DENORM									Error (4)		
(C) qNaN						Error (4)					
(D) qNaN											
sNaN											

Table3 FADD DRm, DRn (DRn + DRm→ DRn) FSUB DRm, DRn (DRn - DRm→ DRn)

DRm \ DRn	NORM	+0	-0	+INF	-INF	(A) positive DENORM	(A) negative DENORM	(B) DENORM	(C) qNaN	(D) qNaN	sNaN
NORM	ADD			+INF	-INF	Error					
+0		+0									
-0			-0								
+INF					Invalid				qNaN		Invalid
-INF	-INF			Invalid	-INF						
(A) positive DENORM									Error (5)		
(A) negative DENORM											
(B) DENORM											
(C) qNaN						Error (5)					
(D) qNaN											
sNaN											

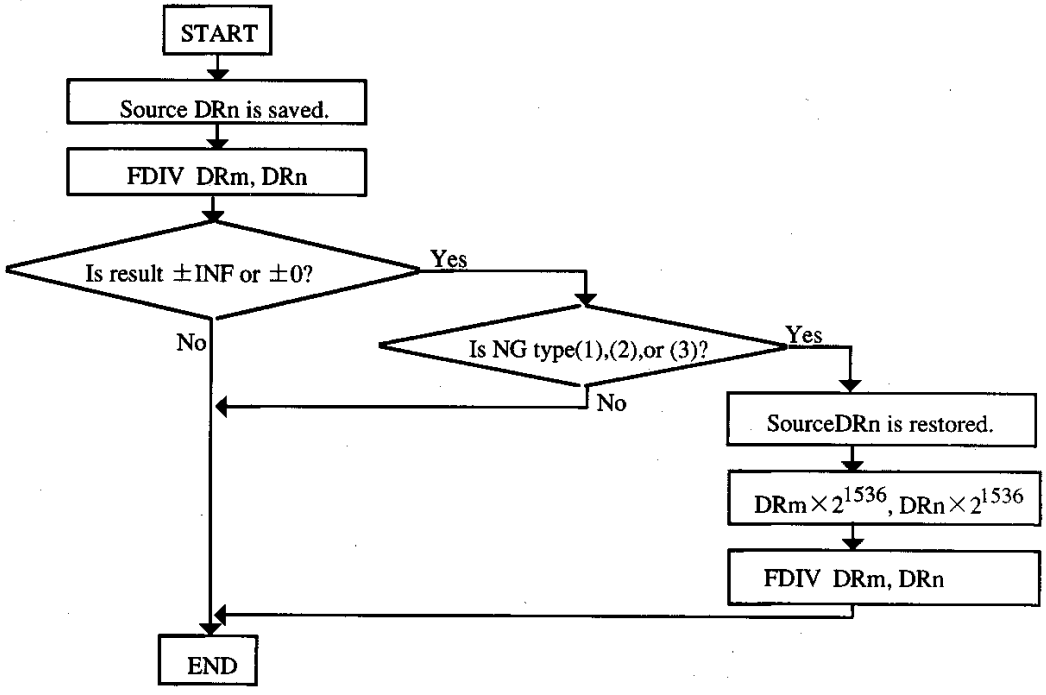
Table4 FMUL DRm, DRn (DRn*DRm→DRn)

DRm \ DRn	NORM	+0	-0	+INF	-INF	(A) positive DENORM	(A) negative DENORM	(B) DENORM	(C) qNaN	(D) qNaN	sNaN
NORM	MUL	0		INF		Error					
+0	0	+0	-0	Invalid							
-0		-0	+0								
+INF	INF	Invalid		+INF	-INF	+INF(7)	-INF(7)		qNaN		Invalid
-INF				-INF	+INF	-INF(7)	+INF(7)				
(A) positive DENORM				+INF(7)	-INF(7)						
(A) negative DENORM				-INF(7)	+INF(7)						
(B) DENORM									Error(6)		
(C) qNaN								Error(6)			
(D) qNaN											
sNaN											

4. Software modification

4.1 NG type: (1),(2),or (3)

In NG types (1),(2),or (3), software should be modified according to the following flows.
Source operand is adjusted by multiplying 2^{1536}, and should be calculated as normalized number.
If NG type is (1) and Divide by Zero exception is enabled, divide by zero exception occurs and the destination register is not modified.
If NG type is (1) and Divide by Zero exception is disabled, the destination register is set to infinity with sign based on source operands.



4.2 NG type: (7)

If NG type is (7), FPU Error exception does not occur. But results is correct, and thus modification by software is not needed.

5. Trap routine modification

In NG types (4),(5),or (6), instruction and input data should be checked and qNaN should be written in destination register as Table5 in TRAP routine.
qNaN is always 64h'7ff7ffff_ffffffff.

Table5 TRAP routine

NG type	instruction check	input check		result
		DRm	DRn	
(4)	FDIV	qNaN	DENORM	qNaN
	FDIV	qNaN	DENORM	qNaN
	FDIV	DENORM	qNaN	qNaN
(5)	FADD/FSUB	qNaN	DENORM	qNaN
	FADD/FSUB	DENORM	qNaN	qNaN
(6)	FMUL	qNaN	DENORM	qNaN
	FMUL	DENORM	qNaN	qNaN