Date: Dec. 15, 2015

RENESAS TECHNICAL UPDATE

TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061 Japan
Renesas Electronics Corporation
Inquiry http://japan.renesas.com/contact/
E-mail: csc@renesas.com

Product Category	MPU & MCU		Document No.	TN-RL*-A025C/E	Rev.	3.00
Title	RL78/G14 Restriction		Information Category	Technical Notification		
Applicable Products	RL78/G14 Family R5F104xx	Lot No All lot	Reference Document	RL78/G14 User's Manual: Hardware Rev.3.20 R01UH0186EJ0320 (Jan. 2015)		

List of Restrictions to be added in this notification

Item	Restrictions that are added in this notification.	Products.	Corresponding page
1.1	Restriction on Setting of P6x (x = 0 to 3)	All	p.2

List of Restrictions of notified

Item	Restrictions of notified.	Products.	Corresponding page
2.1	Restriction of the division instruction (DIVHU, DIVWU)	All	p.3-p.7
2.2	Restriction on Use of Watchdog Timer	Product of ROM capacity 96Kbyte ~ 256Kbyte	p.8-p.10

Revision History

Revision history of RL78/G14 restrictions

Document Number	Date issued	Description
TN-RL*-A025A/E	April 9, 2014	First edition issued List of usage restriction previously: No. 2.2
TN-RL*-A025B/E	October 6, 2014	Second edition issued List of usage restriction previously: No. 2.1
TN-RL*-A025C/E	December 15, 2015	Third edition issued List of usage restriction added: No. 1.1 (this document)

1. Restriction that are added in this notification

1.1. Restriction on Setting of P6x (x = 0 to 3)

1.1.1. Restriction

【Applicable Usage】

This restriction applies when the port pins P60, P61, P62, and P63 are in use.

1.1.2. Details of the Restriction

The actual block diagram for the circuits of port pins P60, P61, P62, and P63 is shown in figure 1 below but the diagram in the user's manual is as shown in figure 2.

As shown in figure 1, the input buffer of the actual circuit is enabled even if the pin is operating as an output (the corresponding bit in port mode register 6 (PM6) is set to 0). This may lead to a shoot-through current flowing through the port pins P60, P61, P62, and P63 when the voltage levels on these pins are intermediate.

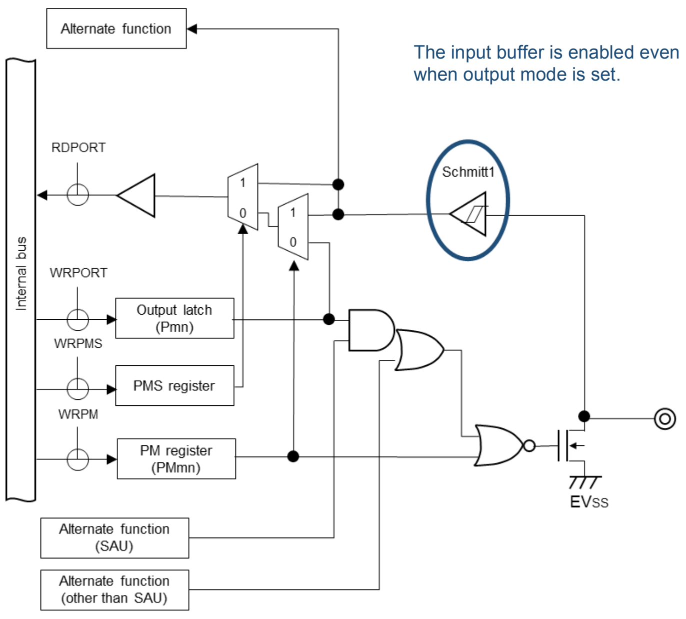

Figure 1 Correct Circuit Diagram

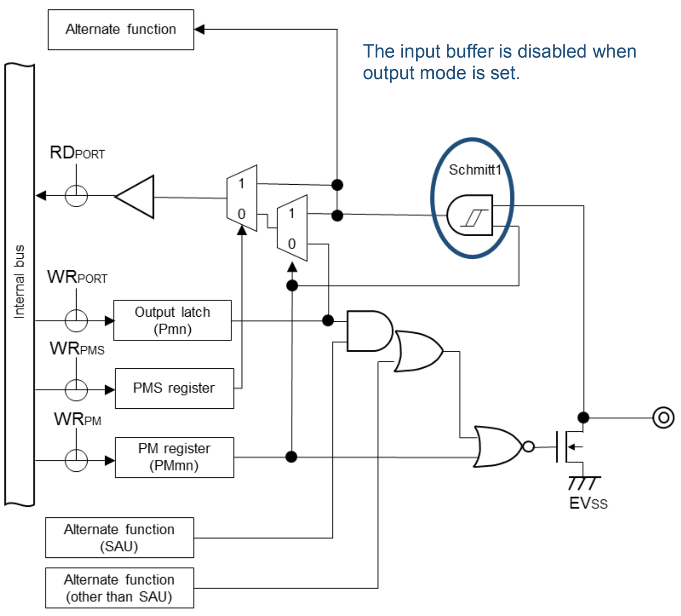

Figure 2 Diagram in the User's Manual

1.1.3. Workaround

Do not apply intermediate voltages to the port pins P60, P61, P62, and P63 regardless of the settings of the corresponding bits in register PM6.

1.1.4. Planned Action

We do not plan to modify the hardware; instead we add a restriction on usage of port pins P60, P61, P62, and P63.

We will correct the block diagram and add a cautionary note on the shoot-through current in the next revision of the user's manual.

2. Restriction notified

2.1. Restriction of the division instruction (DIVHU, DIVWU)

2.1.1. About the restriction

<Usage subject to the restriction>

If the software code applies to ALL of the four conditions below, the code is subject to the restriction.

1) A divide instruction (DIVHU or DIVWU) is executed in an interrupt service routine.
   A divide instruction (DIVHU or DIVWU) is defined as Group 1 instruction.
2) Multiple interrupts are enabled in the interrupt service routine in which the divide instruction (DIVHU or DIVWU) is executed.
3) More than one interrupts with different interrupt priorities occur during the process of the interrupt service routine mentioned in 2) above.
   Please refer to Table 2.1 for the detail of the priorities of the corresponding interrupts.
4) The divide instruction (DIVHU or DIVWU) is followed by a Group 2 instruction.
   Please refer to Item 5. "The List of Group 2 Instruction" for the details of Group 2 instructions. Please note, that any instruction is classified as "Group 2" if the preceding divide instruction is executed in RAM.

2.1.2. Detail of the restriction

There is a possibility of unintended operation when branching from Interrupt A to Interrupt C, or branching from Interrupt C to Interrupt A.

I. A "Group 1" instruction (DIVHU, DIVWU) and a "Group 2" instruction are consecutive in Interrupt A in which multiple interrupts are enabled.

II. Interrupt B, whose request occurs during the process of Interrupt A, is suspended.

III. Interrupt C is generated during the two clock cycles just before the MCU completes the execution of the divide instruction (8th or 9th cycle for DIVHU, 16th or 17th cycle for DIVWU).

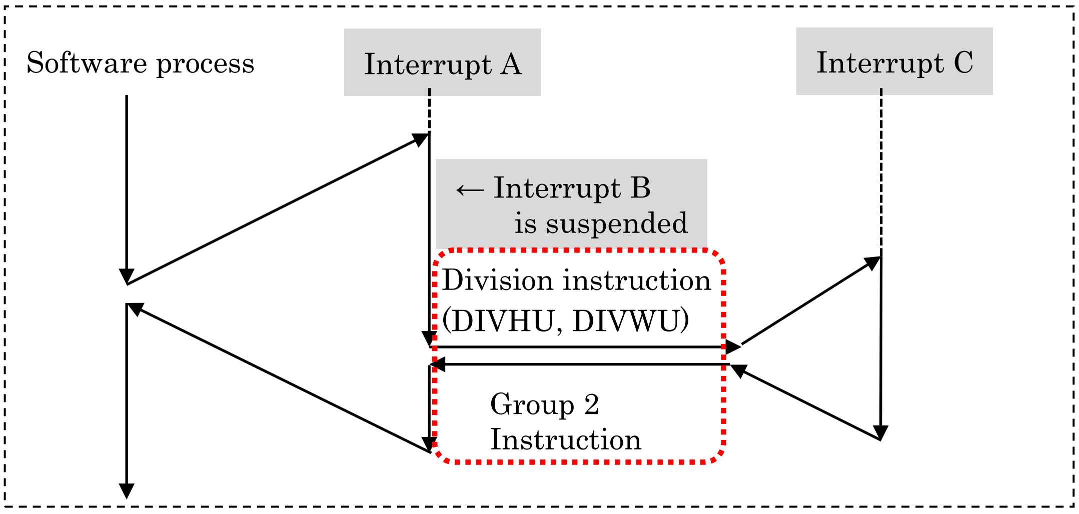
Software process
Interrupt A
Interrupt C
← Interrupt B is suspended
Division instruction (DIVHU, DIVWU)
Group 2 Instruction

Figure2.1. Behavior subject to the restrictions

Note 1: Please refer to Item 5. "The List of Group 2 Instruction" for the details of Group 2 instruction.

Note 2: Whether the MCU accepts an interrupt or not depends on the combination of the priority levels (0 to 3) of the interrupts. Table 2.1 shows the combinations subject to the restriction.

Table2.1. Combinations subject to the restriction

Priority level of Interrupt A	Priority level of Interrupt B	Priority level of Interrupt C	Restrictions
Level 0	Level 1/ Level 2/ Level 3	Level 0	Your code could be subject to the restrictions.
Level 1	Level 1/ Level 2/ Level 3	Level 0	
Level 2	Level 2/ Level 3	Level 0/ Level 1	
Level 3	Level 3	Level 0/ Level 1/ Level 2	
Other than above			Your code is not subject to the restrictions.

2.1.3. Software Workaround

Please implement one of the following software workaround.

（A）Disable interrupts during a divide or modulo operation.

Example:

```
    __asm("push PSW");
    DI();
**Divide or modulo operation in C**
    __asm("pop PSW");
```

（B）Insert a NOP instruction immediately after the divide instruction.
Also, if the divide instruction (DIVHU or DIVWU) is executed in RAM, move it to code flash.

Example:

```
DIVWU          ; Divide instruction
NOP            ; Insert a NOP instruction
RET            ; Group 2 instruction
```

In the case of using a High-level language including C, compilers may generate the instructions subject to this restriction. In this case, take the workaround (A).

Note: In the case of Renesas compiler CA78K0, “#pragma di” should be declared in the code to use DI();

2.1.4. Permanent Measure

We will implement the software workaround into Renesas compiler CA78K0R V1.7.1.

Detail of the implementation:

CA78K0R V1.7.1 always inserts a NOP instruction immediately after each DIVWU / DIVHU instruction when building. This Implementation eliminates the need for the software workaround mentioned in Item 3. Software Workaround".Note

V1.7.1 Release Schedule: November 18, 2014

Note: If a divide instruction (DIVHU or DIVWU) is executed in RAM, code modification is required.

2.1.5. List of Group2 instruction

In the case a divide instruction (DIVHU or DIVWU) is followed by an instruction of Group2, it is subject to the restriction mentioned in this report. Instructions meeting either of the following conditions (Condition 1 to 3) are belong to Group2.

Condition 1: Instruction whose execution cycles are 2 or more.

Instruction	Operand
XCH	A, saddr
	A, sfr
	A, !addr16
	A, [DE]
	A, [DE+byte]
	A, [HL]
	A, [HL+byte]
	A, [HL+B]
	A, [HL+C]
ADD	saddr, #byte
ADDC	saddr, #byte
SUB	saddr, #byte
SUBC	saddr, #byte
AND	saddr, #byte
OR	saddr, #byte
XOR	saddr, #byte

Instruction	Operand
INC	saddr
INC	!addr16
INC	[HL+byte]
DEC	saddr
DEC	!addr16
DEC	[HL+byte]
INCW	saddrp
INCW	!addr16
INCW	[HL+byte]
DECW	saddrp
DECW	!addr16
DECW	[HL+byte]
MOV1	saddr.bit, CY
MOV1	sfr.bit, CY
MOV1	[HL].bit, CY
SET1	saddr.bit
SET1	sfr.bit
SET1	!addr16.bit
SET1	[HL].bit
CLR1	saddr.bit
CLR1	sfr.bit
CLR1	!addr16.bit
CLR1	[HL].bit

Instruction	Operand
CALL	All
CALLT	All
BRK	-
RET	-
RETI	-
RETB	-
BR	All
BC	All
BNC	All
BZ	All
BNZ	All
BH	All
BNH	All
BT	All
BF	All
BTCLR	All
HALT	-
STOP	-

Condition 2: Instruction reading the code flash memory or the mirror area.

Instruction in the tables below reading the code flash memory or the mirror area belong to “Group 2”.

Instruction	Operand
MOV	A, !addr16
	A, [DE]
	A, [DE+byte]
	A, [HL]
	A, [HL+byte]
	A, [HL+B]
	A, [HL+C]
	A, word[B]
	A, word[C]
	A, word[BC]
	B, !addr16
	C, !addr16
	X, !addr16
	A, ES:!addr16
	A, ES:[DE]
	A, ES:[DE+byte]
	A, ES:[HL]
	A, ES:[HL+byte]
	A, ES:[HL+B]
	A, ES:[HL+C]
	A, ES:word[B]
	A, ES:word[C]
	A, ES:word[BC]
	B, ES:!addr16
	C, ES:!addr16
	X, ES:!addr16

Instruction	Operand
MOVW	AX, !addr16
	AX, [DE]
	AX, [DE+byte]
	AX, [HL]
	AX, [HL+byte]
	AX, word[B]
	AX, word[C]
	AX, word[BC]
	BC, !addr16
	DE, !addr16
	HL, !addr16
	AX, ES:!addr16
	AX, ES:[DE]
	AX, ES:[DE+byte]
	AX, ES:[HL]
	AX, ES:[HL+byte]
	AX, ES:word[B]
	AX, ES:word[C]
	AX, ES:word[BC]
	BC, ES:!addr16
	DE, ES:!addr16
	HL, ES:!addr16

Instruction	Operand
ADD ADDC SUB SUBC AND OR XOR	A, !addr16
	A, [HL]
	A, [HL+byte]
	A, [HL+B]
	A, [HL+C]
	A, ES:!addr16
	A, ES:[HL]
	A, ES:[HL+byte]
	A, ES:[HL+B]
	A, ES:[HL+C]
CMP	A, !addr16
	A, [HL]
	A, [HL+byte]
	A, [HL+B]
	A, [HL+C]
	!addr16, #byte
	A, ES:!addr16
	A, ES:[HL]
	A, ES:[HL+byte]
	A, ES:[HL+B]
	A, ES:[HL+C]
	ES:!addr16, #byte
CMP0	!addr16
	ES:!addr16
CMPS	X, [HL+byte]
	X, ES:[HL+byte]

Instruction	Operand
ADDW	AX, !addr16
	AX, [HL+byte]
	AX, ES:!addr16
	AX, ES: [HL+byte]
SUBW	AX, !addr16
	AX, [HL+byte]
	AX, ES:!addr16
	AX, ES: [HL+byte]
CMPW	AX, !addr16
	AX, [HL+byte]
	AX, ES:!addr16
	AX, ES: [HL+byte]
MOV1	CY, [HL].bit
	CY, ES:[HL].bit
AND1	CY, [HL].bit
	CY, ES:[HL].bit
OR1	CY, [HL].bit
	CY, ES:[HL].bit
XOR1	CY, [HL].bit
	CY, ES:[HL].bit
BT	ES:[HL].bit, $addr20
BF	ES:[HL].bit, $addr20

Condition 3 : Instruction suspending interrupt requests.

Instruction listed in the table below, that suspend interrupt requests, belong to Group2.

Instruction	Operand
MOV	PSW, #byte
MOV	PSW, A
MOV1	PSW.bit, CY
SET1	PSW.bit
CLR1	PSW.bit
RETB	-
RETI	-
POP	PSW
BTCLR	PSW.bit,$addr20
EI	-
DI	-
SKC	-
SKNC	-
SKZ	-
SKNZ	-
SKH	-
SKNH	-

Instruction writing to the registers below belong to Group2 since they suspend interrupt requests.

Writing to the registers below by register addressing also subjects to the condition3.

- Interrupt request flag register

  IF0L, IF0H, IF1L, IF1H, IF2L, IF2H
- Interrupt mask flag register

  MK0L, MK0H, MK1L, MK1H, MK2L, MK2H
- Priority specification flag register

  PR00L, PR00H, PR01L,PR01H, PR02L, PR02H

  PR10L, PR10H, PR11L, PR11H, PR12L, PR12H

The table below shows the instructions writing to the registers listed above.

Instruction	Operand
MOV	sfr, #byte
MOV	!addr16, #byte
MOV	sfr, A
MOV	!addr16, A
MOV	[DE], A
MOV	[DE+byte], #byte
MOV	[DE+byte], A
MOV	[HL], A
MOV	[HL+byte], #byte
MOV	[HL+byte], A
MOV	[HL+B], A
MOV	[HL+C], A
MOV	word[B], #byte
MOV	word[B], A
MOV	word[C], #byte
MOV	word[C], A
MOV	word[BC], #byte
MOV	word[BC], A

Instruction	Operand
XCH	A, sfr
XCH	A, !addr16
XCH	A, [DE]
XCH	A, [DE+byte]
XCH	A, [HL]
XCH	A, [HL+byte]
XCH	A, [HL+B]
XCH	A, [HL+C]
ONEB	!addr16
CLRB	!addr16
MOVS	[HL+byte], X
MOVW	sfrp, #word
MOVW	sfrp, AX
MOVW	!addr16, AX
MOVW	[DE], AX
MOVW	[DE+byte], AX
MOVW	[HL], AX
MOVW	[HL+byte], AX
MOVW	word[B], AX
MOVW	word[C], AX
MOVW	word[BC], AX

Instruction	Operand
INC	!addr16
INC	[HL+byte]
DEC	!addr16
DEC	[HL+byte]
INCW	!addr16
INCW	[HL+byte]
DECW	!addr16
DECW	[HL+byte]
MOV1	sfr.bit, CY
MOV1	[HL].bit, CY
SET1	sfr.bit
SET1	!addr16.bit
SET1	[HL].bit
CLR1	sfr.bit
CLR1	!addr16.bit
CLR1	[HL].bit
BTCLR	sfr.bit, $addr20
BTCLR	[HL].bit, $addr20

2.2. Reatriction on Use of the Watchdog Timer

2.2.1. About the restriction

Condition:

The restriction on applies to the following settings:

- The overflow time of the watchdog timer is set to $2^{13}/f_{IL}$, $2^{14}/f_{IL}$, or $2^{16}/f_{IL}$
- The watchdog timer interval interrupt is used.

· User option byte (000C0H) settings

7	6	5	4	3	2	1	0
WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

WDTINT	Use of interval interrupt of watchdog timer
1	Interval interrupt is generated when 75% + 1/2 f_{IL} of the overflow time is reached.

WDTON	Operation control of watchdog timer counter
1	Counter operation enabled (counting started after reset)

WDCS2	WDCS1	WDCS0	Watchdog timer overflow time
1	0	1	$2^{13}/f_{IL}$ (474.90 ms)
1	1	0	$2^{14}/f_{IL}$ (949.80 ms)
1	1	1	$2^{16}/f_{IL}$ (3799.19 ms)

Settings other than the above do not apply to the restriction.

Description:

After clearing the watchdog timer, the watchdog timer interval interrupt (INTWDTI) is unintentionally generated after one clock of the watchdog timer elapses.

2.2.2. Software Workaround

Perform the procedure below to reset the watchdog timer counter.

(1) The WDTIMK flag [*1] is set to 1 before the watchdog timer counter is reset.
(2) Reset the watchdog timer counter.
(3) Wait at least 80 μs.
(4) Clear the WDTIIF flag [*2] to 0 and clear the WDTIMK flag [*1] to 0.

[*1] : Bit 0 of interrupt mask flag register 0 (MK0L register)
[*2] : Bit 0 of interrupt request flag register 0 (IF0L register)

2.2.3. Modification Schedule

This matter is considered to be a restriction. The above mentioned workaround will be added to Chapter 13 Watchdog Timer when User's Manual: Hardware Rev.3.10 is published.

2.2.4. Products Affected by This Restriction

No.	Description	Products[Note]	
		RL78/G14 (Code Flash Memory 16 KB to 64 KB) R5F104xA, R5F104xC, R5F104xD, R5F104xE	RL78/G14 (Code Flash Memory 96 KB to 256 KB) R5F104xF, R5F104xG, R5F104xH, R5F104xJ
1	Restriction on Use of Watchdog Timer	Restriction not applicable	Restriction applicable

Note. This restriction is not intended for R5F104xK and R5F104xL.
R5F104xK and R5F104xL has 384 KB to 512 KB ROM capacity.
This Note was added to the second edition.

Appendix 2-1

List of Products Affected by This Restriction

RL78/G14 (Code Flash Memory 96 KB to 256 KB)

(Product names are blank product names. The ROM number is omitted.)

30-pin LSSOP 7.62 mm (300)	R5F104AFASP, R5F104AGASP R5F104AFDSP, R5F104AGDSP R5F104AFGSP, R5F104AGGSP
32-pin HWQFN 5x5 mm	R5F104BFANA, R5F104BGANA R5F104BFDNA, R5F104BGDNA R5F104BFGNA, R5F104BGGNA
32-pin LQFP 7x7 mm	R5F104BFAFP, R5F104BGAFP R5F104BFDFP, R5F104BGDFP R5F104BFGFP, R5F104BGGFP
36-pin WFLGA 4x4 mm	R5F104CFALA, R5F104CGALA R5F104CFGLA, R5F104CGGLA
40-pin HWQFN 6x6 mm	R5F104EFANA, R5F104EGANA, R5F104EHANA R5F104EFDNA, R5F104EGDNA, R5F104EHDNA R5F104EFGNA, R5F104EGGNA, R5F104EHGNA
44-pin LQFP 10x10 mm	R5F104FFAFP, R5F104FGAFP, R5F104FHAFP, R5F104FJAFP, R5F104FFDFP, R5F104FGDFP, R5F104FHDFP, R5F104FJDFP, R5F104FFGFP, R5F104FGGFP, R5F104FHGFP, R5F104FJGFP,
48-pin LFQFP 7x7 mm	R5F104GFAFB, R5F104GGAFB, R5F104GHAFB, R5F104GJAFB R5F104GFDFB, R5F104GGDFB, R5F104GHDFB, R5F104GJDFB R5F104GFGFB, R5F104GGGFB, R5F104GHGFB, R5F104GJGFB
48-pin HWQFN 7x7 mm	R5F104GFANA, R5F104GGANA, R5F104GHANA, R5F104GJANA R5F104GFDNA, R5F104GGDNA, R5F104GHDNA, R5F104GJDNA R5F104GFGNA, R5F104GGGNA, R5F104GHGNA, R5F104GJGNA
52-pin LQFP 10x10 mm	R5F104JFAFA, R5F104JGAFA, R5F104JHAFA, R5F104JJAFA R5F104JFDFA, R5F104JGDFA, R5F104JHDFA, R5F104JJDFA R5F104JFGFA, R5F104JGGFA, R5F104JHGFA, R5F104JJGFA
64-pin LQFP 12x12 mm	R5F104LFAFA, R5F104LGAFA, R5F104LHAFA, R5F104LJAFA R5F104LFDFA, R5F104LGDFA, R5F104LHDFA, R5F104LJDFA R5F104LFGFA, R5F104LGGFA, R5F104LHGFA, R5F104LJGFA
64-pin LFQFP 10x10 mm	R5F104LFAFB, R5F104LGAFB, R5F104LHAFB, R5F104LJAFB R5F104LFDFB, R5F104LGDFB, R5F104LHDFB, R5F104LJDFB R5F104LFGFB, R5F104LGGFB, R5F104LHGFB, R5F104LJGFB
64-pin FLGA 5x5 mm	R5F104LFALA, R5F104LGALA, R5F104LHALA, R5F104LJALA R5F104LFGLA, R5F104LGGLA, R5F104LHGLA, R5F104LJGLA
64-pin LQFP 14x14 mm	R5F104LFAFP, R5F104LGAFP, R5F104LHAFP, R5F104LJAFP R5F104LFDFP, R5F104LGDFP, R5F104LHDFP, R5F104LJDFP R5F104LFGFP, R5F104LGGFP, R5F104LHGFP, R5F104LJGFP
80-pin LFQFP 12x12 mm	R5F104MFAFB, R5F104MGAFB, R5F104MHAFB, R5F104MJAFB R5F104MFDFB, R5F104MGDFB, R5F104MHDFB, R5F104MJDFB R5F104MFGFB, R5F104MGGFB, R5F104MHGFB, R5F104MJGFB
80-pin LQFP 14x14 mm	R5F104MFAFA, R5F104MGAFA, R5F104MHAFA, R5F104MJAFA R5F104MFDFA, R5F104MGDFA, R5F104MHDFA, R5F104MJDFA R5F104MFGFA, R5F104MGGFA, R5F104MHGFA, R5F104MJGFA
100-pin LFQFP 14x14 mm	R5F104PFAFB, R5F104PGAFB, R5F104PHAFB, R5F104PJAFB R5F104PFDFB, R5F104PGDFB, R5F104PHDFB, R5F104PJDFB R5F104PFGFB, R5F104PGGFB, R5F104PHGFB, R5F104PJGFB
100-pin LQFP 14x20 mm	R5F104PFAFA, R5F104PGAFA, R5F104PHAFA, R5F104PJAFA R5F104PFDFA, R5F104PGDFA, R5F104PHDFA, R5F104PJDFA R5F104PFGFA, R5F104PGGFA, R5F104PHGFA, R5F104PJGFA