SH7266/SH7267 Groups

Renesas Quad Serial Peripheral Interface

High Speed Read Processing on Serial Flash Memory

R01AN0647EJ0100
Rev. 1.00
Sep. 28, 2011

Abstract

SH7266/SH7267 Renesas Quad Serial Peripheral Interface (RQSPI) can access the serial flash memory that supports multiple I/Os, and performs a read processing in high speed clock of maximum 72MHz. This application explains about the procedure to speed up the read processing in the RQSPI.

Target Device

SH7267

Contents

1. Introduction 2
2. Applications 3
3. Sample Program List 7
4. References 31

1. Introduction

1.1 Specification

This application note consists of an explanation of an application program for high speed performance of a read processing in the Renesas Quad Serial Peripheral Interface (hereinafter called RQSPI), and an application program. The details on the basic usage of RQSPI is referred to the Application Note “SH7266/SH7267 Group Interfacing Serial Flash Memory Using the Renesas Quad Serial Peripheral Interface”. Therefore, this application note is limited to explain the revised points for a high speed read processing.

1.2 Modules Used

- Renesas Quad Serial Peripheral Interface (RQSPI)
- Renesas Serial Peripheral Interface (RSPI)
- Boot Mode (serial flash memory boot)
- General-purpose I/O port

1.3 Operation Confirmation Conditions

Item	Contents
MCU used	SH7266/SH7267
Operating frequency	Internal clock: 144MHz
	Bus clock: 72MHz
	Peripheral clock: 36MHz
Integrated development environment	High-performance Embedded Workshop Ver.4.07.00 (by Renesas Electronics)
C compiler	SuperH RISC engine family C/C++ Compiler Package Ver.9.03 Release02 (by Renesas Electronics)
Compile option	High-performance Embedded Workshop in default setting (-cpu=sh2afpu -fpu=single -object="$(CONFIGDIR)\$(FILELEAF).obj" -debug -gbr=auto -chgincpath -errorpath -global_volatile=0 - opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 - nologo)
Serial flash memory	S25FL032P (by Spansion)

1.4 Related Application Notes

For additional information associated with this application note, refer to the following application notes.

- SH7266/SH7267 Group Interfacing Serial Flash Memory Using the Renesas Quad Serial Peripheral Interface
- SH7266/SH7267 Group Boot From the Serial Flash Memory
- SH7266/SH7267 Groups Booting from the serial flash memory (with multiple I/Os)

1.5 About Active-low Pins (Signal)

The symbol “#” suffixed to the pin (or signal) names indicates that the pins (or signals) are active-low.

2. Applications

In this application, the Renesas Quad Serial Peripheral Interface (RQSPI) connects to a serial flash memory to read data in high speed QSPCLK at 72MHz . This performance is under the condition; Bφ = 72MHz. (1)

For details on the read/write procedure in the QSPCLK at 36MHz, refer to the application note, “SH7266/SH7267 Group Interfacing Serial Flash Memory Using the Renesas Quad Serial Peripheral Interface”. This application note explains the procedure for extending the set up period to read in the QSPCLK at 72MHz and revised sections in the reference program.

Note: 1. When in write processing, setting the QSPCLK to 72MHz is not allowed.

2.1 Extending the Setup Time

The access speed is proportional to the frequency of the QSPCLK. The SH7267 enables to set the QSPCLK to 72MHz limited for the receive operation. Therefore in this application, the QSCLK frequency for a read processing is set to 72MHz. At the same time, the data latch timing is set to be delayed half a cycle to fulfill the data input set up time (t_{SU}).

Figure 1 shows the read processing timing in this application.

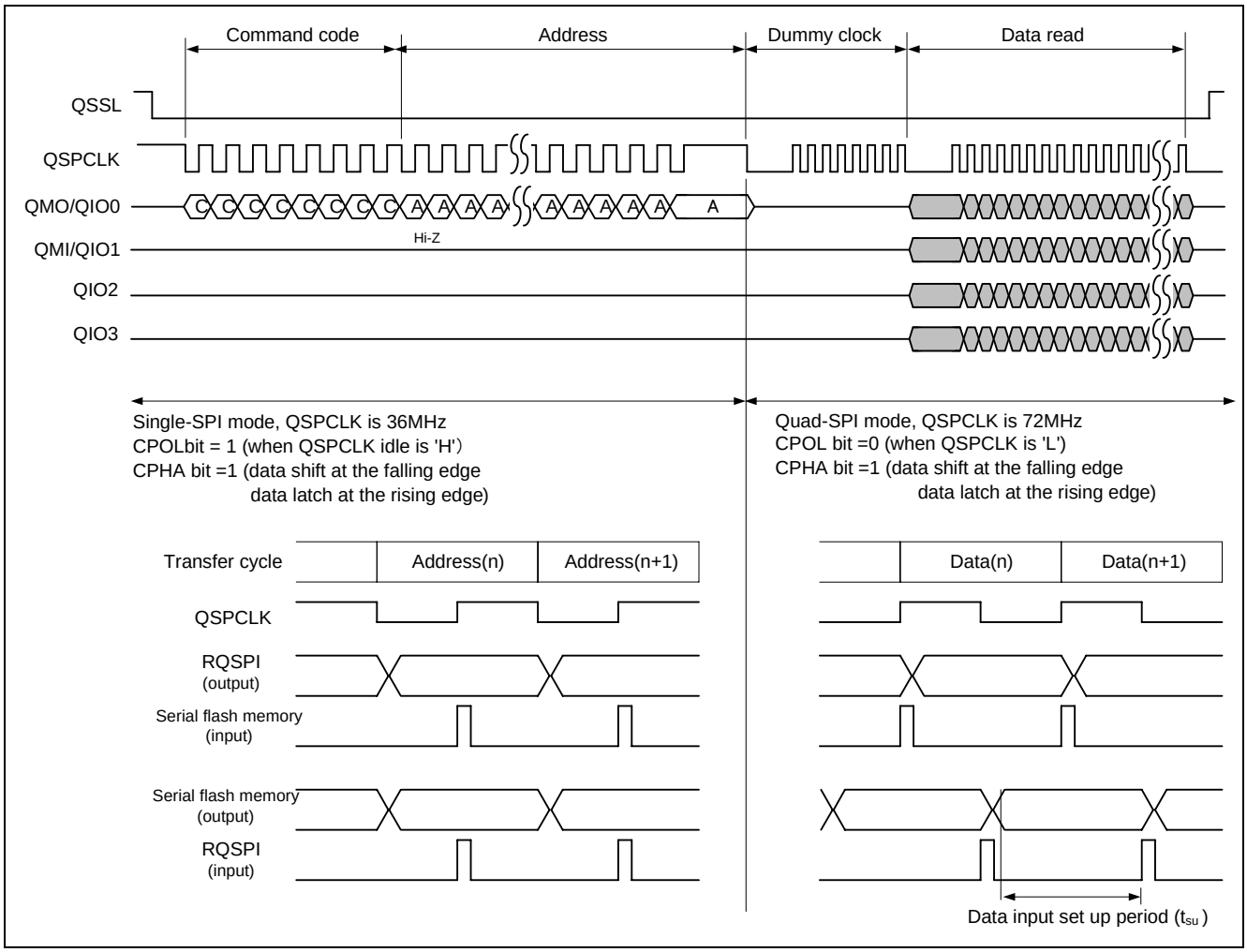

Figure 1 Read Processing Timing in This Application

2.2 Interface Timing

Figure 2 shows the interface timing when expanding the set up period. Table 1 and Table 2 list the timing condition for the serial flash memory and the SH7267. Set the program to fulfill these conditions.

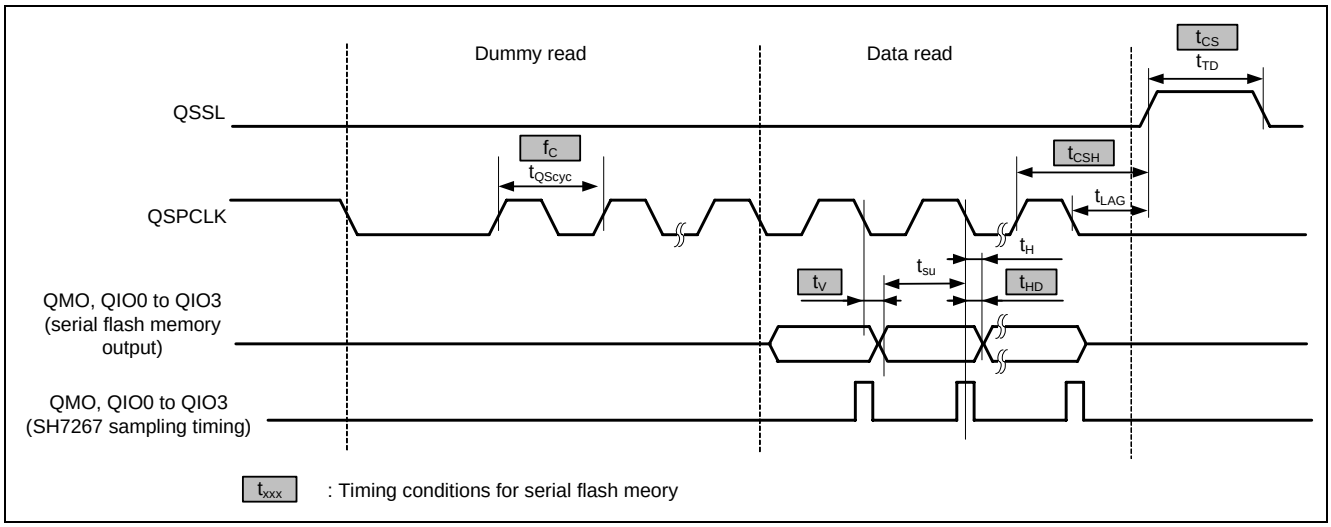

Figure 2 Interface Timing when Expanding Set-up Period

Table 1 Timing Conditions for Serial Flash Memory when Expanding Set-up Period

Symbol	Item	Description	Related Register
t_{CS}	Chip select 'H' time	Time required for QSSL negation t_{TD}(= next access delay) x t_{QScyc} × x t_{cyc} ≧ t_{CS} (min)	SPND register SPCMD register
f_{SCK}	Serial clock frequency	Maximum operation frequency that a slave can respond to The following formula must be fulfilled: f_C(max) ≧ 1/ (t_{QScyc} x t_{cyc})	SPBR register SPCMD register
t_{CSH}	Chip select 'L' hold time	Hold time required between the last QSPCLK rising and the QSSL negattion The following formula must be fulfilled: t_{LAG}(=QSSL negate delay x t_{QScyc} x t_{cyc} ≧ t_{CSH} (min)	SSLND register SPCMD register

Note: t_{cyc} represents one-cycle time of the bus clock (Bφ).

Table 2 Timing Conditions for SH7267 when Expanding Set-up Period

Symbol	Item	Description	Related Register
t_{SU}	Data input set-up time	Time required by the slave between data output and data sampling The following formula must be fulfilled: tQScyc x tcyc - tV(max) ≧ tSU(min)	SPBR register SPCMD register
t_H	Data input hold time	Time required by the slave to hold the data output The following formula must be fulfilled: tHO(min) ≧ tH(min)	

Note: t_{cyc} represents one-cycle time of the bus clock (Bφ).

2.3 Revision in the Sample Program

This section explains about the revised points in this sample program described in this application note. The revision is made based on the application note “SH7266/SH7267 Group Serial Flash Memory Using the Renesas Quad Serial Peripheral Interface”.

Figure 3 shows the sequence controls used in this sample program.

Revisions are made to the setting values for the command register 1 (SPCMD1) and the command register 2 (SPCMD2) to execute a read processing in the Quad Output Read Command (H'6B) in the 72MHz clock. Table 3 lists the revised values. A data register read (SPDR) is not carried by the CPU, but by the 16byte burst transfer using the DMAC which setting values are shown in Table 4.

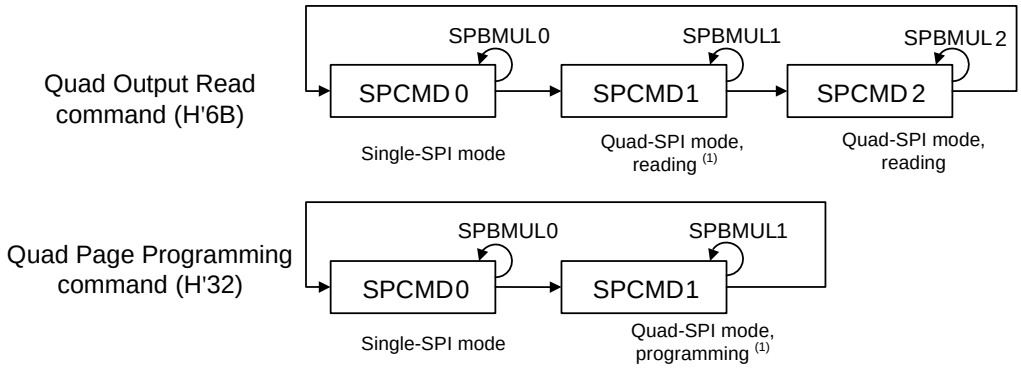

Figure 3 Sequence Control in the Sample Program

Table 3 SPCMDn Register Setting Values when at High Speed Read

Register Name	Setting Value	Description
SPCMD0	H'E287	• SCKDEN="B'1" : the clock delay time is the SPCKD setting value • SLNDEN="B'1" : the QSSL negation delay time is the SSLND setting value • SPNDEN="B'1" : the next access delay time is SPND setting value • LSBF="B'0" : MSB first • SPB[3:0]="B'0010" : transfer data is 32-bit length • SSLKP="B'1" : hold the QSSL signal after transferring • SPIMOD[1:0]="B'00" : Single-SPI • SPRW="B'0" : Programming operation • BRDV[1:0]="B'01" : half of the basic bit rate (36MHz) • CPOL="B'1" : QSCLK is 1 when idling • CPHA="B'1" : Data shift at falling edge, dadta latch at rising edge
SPCMD1	H'E2D1	• SPIMOD[1:0]="B'10" : Quad-SPI • SPRW="B'1" : reading operation • BRDV[1:0]="B'00" : basic bit rate (72MHz) • CPOL="B'0": QSPCLK is 0 when idling (other bit setting value is the same as those in the SPCMD0 register)
SPCMD2	H'E2D1	(same setting value as in the SPCMD1 register)
SPSCR	H’02	• SPSC[1:0]="B'10" : SPCMD number to be referred to (0 → 1 → 2 → - - -)

Table 4 DMAC Register Setting Value

Register Name	Setting Value	Description
CHCR4	H'02004818	• TC=B'0 : perform one transfer for one transfer request • SAF=B'1 : perform 4 transfers by 4byte from the SAR register • erDM[1:0]=B'01 : destination address increases • SM[1:0]=B'00 : source address is fixed • RS[3:0]=B'1000 : Transfer request source is DMA expansion resource • selector • TB=B'0 : cycle steal mode • TS[1:0]=B'11 : transfer unit is 16byte (long word x 4) • IE=B'0 : disable interrupt request • DE=B'1 : enable DMA transfer
SAR4	H'FFFFE004	SPDR register in the RQSPI
DAR4	-	Read buffer address
DMARS2	H'A2	• Transfer request source is the RQSPI reception
DMAOR	H'0001	• Enable DMA transfer in all channels

3. Sample Program List

3.1 Supplemental Information on the sample Programs

When using the boot mode 0 (the boot from the memory connected to the CS0 space), the pins QIO2 and QIO3 cannot be set as the RQSPI function. Therefore the application program should be started in the boot mode 1 or boot mode 3 (serial flash boot).

For the details on the boot procedure when using the serial flash boot, and on writing the program in the serial flash memory, refer to the application note “SH7266/SH7267 Groups Boot Example (with multiple I/Os) from the Serial Flash Memory”.

3.2 Sample Program List “main.c” (1)

```
1   /***********************************************************************
2   *   DISCLAIMER
3   *
4   *   This software is supplied by Renesas Electronics Corporation and is only
5   *   intended for use with Renesas products. No other uses are authorized.
6   *
7   *   This software is owned by Renesas Electronics Corporation and is protected under
8   *   all applicable laws, including copyright laws.
9   *
10  *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11  *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12  *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13  *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14  *   DISCLAIMED.
15  *
16  *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17  *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18  *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19  *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20  *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21  *
22  *   Renesas reserves the right, without notice, to make changes to this
23  *   software and to discontinue the availability of this software.
24  *   By using this software, you agree to the additional terms and
25  *   conditions found by accessing the following link:
26  *   http://www.renesas.com/disclaimer
27  ************************************************************************
28  * Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
29  *""FILE COMMENT""*********** Technical reference data *************************
30  *   System Name : SH7266/SH7267 Sample Program
31  *   File Name   : main.c
32  *   Abstract    : RQSPI quad serial flash memory high speed reading
33  *   Version     : 1.00.00
34  *   Device      : SH7266/SH7267
35  *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36  *               : C/C++ compiler package for the SuperH RISC engine family
37  *               :                               (Ver.9.03 Release02).
38  *   OS          : None
39  *   H/W Platform: R0K57267(CPU board), Spansion[S25FL032P](serial flash)
40  *   Description :
41  ************************************************************************
42  *   History     : Aug.20,2010 Ver.1.00.00
43  *""FILE COMMENT END""*****************************************************/
44  #include <stdio.h>
45  #include "qserial_flash.h"
46
```

3.3 Sample Program List “main.c” (2)

```
47    /* ==== Macro definition ==== */
48    #define TOP_ADDRESS    0              /* Start address on the serial flash memory */
49
50    /* ==== Function prototype declaration ==== */
51    void main(void);
52
53    /* ==== Variables definition ==== */
54    #pragma section DEBUG_BUFFER
55    static unsigned char data[SF_SECTOR_SIZE];
56    static unsigned char rbuf[SF_SECTOR_SIZE];
57    #pragma section
58
59    /*""FUNC COMMENT""*****************************************************************
60     * ID           :
61     * Outline      : Main processing for serial flash memory access
62     *-------------------------------------------------------------------------------
63     * Include      :
64     *-------------------------------------------------------------------------------
65     * Declaration  : void main(void);
66     *-------------------------------------------------------------------------------
67     * Description  : Erasing, programming and reading in the serial flash memory.
68     *              : After initializing the RSPI channel 0, erases all the areas
69     *              : except sector 0, then writes data. Read and check the result.
70     *-------------------------------------------------------------------------------
71     * Argument     : void
72     *-------------------------------------------------------------------------------
73     * Return Value : void
74     *-------------------------------------------------------------------------------
75     * Note         : None
76     *""FUNC COMMENT END""*************************************************************/
77    void main(void)
78    {
79      int i, j;
80      static unsigned long addr;
81
82      /* ==== Initializes the serial flash memory  ==== */
83      sf_init_serial_flash();
84
85      /* ==== Cancels the serial flash memory protect ==== */
86      sf_protect_ctrl( SF_REQ_UNPROTECT );
87
88      /* ==== Erases all the sectors except sector 0 ==== */
89      for(i = 1; i < SF_NUM_OF_SECTOR; i++){
90        sf_sector_erase( i );
91      }
```

3.4 Sample Program List “main.c” (3)

```
92      /* ==== Programs data in all the sectors except sector 0 ==== */
93      addr = TOP_ADDRESS + SF_SECTOR_SIZE;        /* In sector 1 */
94      for(i = 1; i < SF_NUM_OF_SECTOR; i++){      /* In sector 1 and later sectors */
95
96        /* ---- Initializes data for one sector ---- */
97        for(j = 0; j < SF_SECTOR_SIZE; j++){
98            data[j] = (i + j) % 100;
99        }
100       /* ---- Writes the sector size ---- */
101       for(j = 0; j < ( SF_SECTOR_SIZE / SF_PAGE_SIZE ); j++){
102
103           /* ---- Write the page size ---- */
104           sf_byte_program( addr, data+(j*SF_PAGE_SIZE), SF_PAGE_SIZE );
105           addr += SF_PAGE_SIZE;             /* Renew the address to write to */
106       }
107     }
108     /* ==== Reads data in all the sectors except sector 0 ==== */
109     addr = TOP_ADDRESS + SF_SECTOR_SIZE;    /* In sector 1 */
110     for(i = 1; i < SF_NUM_OF_SECTOR; i++){  /* In sector 1 and later sectors */
111
112       /* ---- Reads the sector size ---- */
113       sf_byte_read( addr, rbuf, SF_SECTOR_SIZE );
114       addr += SF_SECTOR_SIZE;               /* Renews the address to read from  */
115
116       /* ---- Verify check ---- */
117       for(j = 0; j < SF_SECTOR_SIZE; j++){
118           data[j] = (i + j) % 100;          /* Regenerates the written data */
119           if( data[j] != rbuf[j] ){
120               puts("Error: verify error\n");
121               fflush(stdout);
122               while(1){
123                   /* error */
124               }
125           }
126       }
127     }
128     /* ==== Serial flash memory protect ==== */
129     sf_protect_ctrl( SF_REQ_PROTECT );
130
131     while(1){
132          /* loop */
133       }
134   }
135
136   /* End of File */
```

3.5 Sample Program List “qserial_flash.c” (1)

```
/*********************************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corporation and is only
*   intended for use with Renesas products.  No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corporation and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
*********************************************************************************
*   Copyright (C) 2010 Renesas Electronics Corporation. All Rights Reserved.
*""FILE COMMENT""*********** Technical reference data *************************
*   System Name : SH7267 Sample Program
*   File Name   : qserial_flash.c
*   Abstract    : RQSPI quad serial flash memory high-speed read
*   Version     : 1.02.00
*   Device      : SH7266/SH7267
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                               (Ver.9.03 Release02).
*   OS          : None
*   H/W Platform: R0K57267(CPU board), Spansion[S25FL032P](serial flash)
*   Description :
*********************************************************************************
*   History     : Aug.20,2010 Ver.1.00.00
*               : Oct.13,2010 Ver.1.01.00 modified the sequence control method
*               : Apr.14,2010 Ver.1.02.00 added a buffer to streo dummy read data
*               :                         tentatively
*""FILE COMMENT END""*************************************************************/
#include <stdio.h>
#include <machine.h>
#include "iodefine.h"
#include "qserial_flash.h"
#include "rqspi.h"
```

3.6 Sample Program List “qserial_flash.c” (2)

```
52    /* ==== Macro definition ==== */
53    /* ---- Serial flash memory command[S25FL032P(Spansion)] ---- */
54    #define SFLASHCMD_CHIP_ERASE      0xc7
55    #define SFLASHCMD_SECTOR_ERASE    0xd8
56    #define SFLASHCMD_BYTE_PROGRAM    0x02
57    #define SFLASHCMD_BYTE_READ       0x0B
58    #define SFLASHCMD_QUAD_PROGRAM    0x32
59    #define SFLASHCMD_QUAD_READ       0x6B
60    #define SFLASHCMD_WRITE_ENABLE    0x06
61    #define SFLASHCMD_WRITE_DISABLE   0x04

      (omitted)

288   /*""FUNC COMMENT""*****************************************************************
289    * ID           :
290    * Outline      : Data read
291    *-------------------------------------------------------------------------------
292    * Include      :
293    *-------------------------------------------------------------------------------
294    * Declaration  : void sf_byte_read(unsigned long addr, unsigned char *buf, int size);
295    *-------------------------------------------------------------------------------
296    * Description  : Reads the serial flash memory by the specified byte count
297    *-------------------------------------------------------------------------------
298    * Argument     : unsigned long addr ; I : Serial flash memory address to read
299    *              : unsigned char *buf ; I : Buffer address to store read data
300    *              : int size           ; I : Byte count to read
301    *-------------------------------------------------------------------------------
302    * Return Value : void
303    *-------------------------------------------------------------------------------
304    * Note         : None
305    *""FUNC COMMENT END""*************************************************************/
306   void sf_byte_read(unsigned long addr, unsigned char *buf, int size)
307   {
308     unsigned char cmd[4];
309     unsigned long dummy[2];
310
311     cmd[0] = SFLASHCMD_QUAD_READ;
312     cmd[1] = (unsigned char)((addr >> 16) & 0xff);
313     cmd[2] = (unsigned char)((addr >>  8) & 0xff);
314     cmd[3] = (unsigned char)( addr        & 0xff);
315
316     io_rqspi_set_cmd( 0, SPI_SINGLE,  cmd, NULL, sizeof(cmd));
317     io_rqspi_set_cmd( 1, SPI_QUAD_RD, NULL, dummy, 4); /* (2clk/byte)×4 = 8clk */
318     io_rqspi_set_cmd( 2, SPI_QUAD_RD, NULL, buf, size);
319     io_rqspi_transfer(2);
320   }

      (The rest is omitted.)

483   /* End of File */
```

3.7 Sample Program List “qserial_flash.h” (1)

```
1   /***********************************************************************
2   *   DISCLAIMER

    (omitted)

27  ***********************************************************************
28  *   Copyright (C) 2010 Renesas Electronics Corporation. All Rights Reserved.
29  *""FILE COMMENT""*********** Technical reference data *********************
30  *   System Name : SH7267 Sample Program
31  *   File Name   : qserial_flash.h
32  *   Abstract    : RQSPI quad serial flash memory high-speed read
33  *   Version     : 1.00.00
34  *   Device      : SH7266/SH7267
35  *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36  *               : C/C++ compiler package for the SuperH RISC engine family
37  *               :                              (Ver.9.03 Release02).
38  *   OS          : None
39  *   H/W Platform: R0K57267(CPU board), Spansion[S25FL032P](serial flash)
40  *   Description :
41  ***********************************************************************
42  *   History     : Aug.20,2010 Ver.1.00.00
43  *""FILE COMMENT END""*****************************************************/
44  #ifndef _QSERIAL_FLASH_H_
45  #define _QSERIAL_FLASH_H_
46
47  /* ==== Macro definition ==== */
48  #define SF_PAGE_SIZE      256        /* Page size */
49  #define SF_SECTOR_SIZE    0x10000    /* Sector size as 64KB   */
50  #define SF_NUM_OF_SECTOR  64         /* Sector count as 64   */
51
52  enum sf_req{
53    SF_REQ_PROTECT = 0,                /* Protect request */
54    SF_REQ_UNPROTECT,                  /* Protect cancel request */
55    SF_REQ_SERIALMODE,                 /* Serial/dual mode request */
56    SF_REQ_QUADMODE,                   /* Quad mode request */
57  };
58
59  /* ==== Function prototype declaration ==== */
60  void sf_init_serial_flash(void);
61  void sf_protect_ctrl(enum sf_req req);
62  void sf_set_mode(enum sf_req req);
63  void sf_chip_erase(void);
64  void sf_sector_erase(int sector_no);
65  void sf_byte_program(unsigned long addr, unsigned char *buf, int size);
66  void sf_byte_read(unsigned long addr, unsigned char *buf, int size);
67
68  #endif /* _QSERIAL_FLASH_H_ */
69  /* End of File */
70
```

3.8 Sample Program List “rqspi.c” (1)

```
1   /***********************************************************************
2   *   DISCLAIMER
3   *
4   *   This software is supplied by Renesas Electronics Corporation and is only
5   *   intended for use with Renesas products.  No other uses are authorized.
6   *
7   *   This software is owned by Renesas Electronics Corporation and is protected under
8   *   all applicable laws, including copyright laws.
9   *
10  *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11  *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12  *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13  *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14  *   DISCLAIMED.
15  *
16  *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17  *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18  *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19  *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20  *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21  *
22  *   Renesas reserves the right, without notice, to make changes to this
23  *   software and to discontinue the availability of this software.
24  *   By using this software, you agree to the additional terms and
25  *   conditions found by accessing the following link:
26  *   http://www.renesas.com/disclaimer
27  ***********************************************************************
28  *   Copyright (C) 2010(2011) Renesas Electronics Corporation. All Rights Reserved.
29  *""FILE COMMENT""*********** Technical reference data *************************
30  *   System Name : SH7267 Sample Program
31  *   File Name   : rqspi.c
32  *   Abstract    : RQSPI quad serial flash memory high-speed read
33  *   Version     : 1.01.00
34  *   Device      : SH7266/SH7267
35  *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36  *               : C/C++ compiler package for the SuperH RISC engine family
37  *               :                              (Ver.9.03 Release02).
38  *   OS          : None
39  *   H/W Platform: R0K57267(CPU board), Spansion[S25FL032P](serial flash)
40  *   Description :
41  ***********************************************************************
42  *   History     : Aug.20,2010 Ver.1.00.00
43  *               : Apr.14,2011 Ver.1.01.00 Changed the clock frequency to 72MHz when
44  *               :                         Dual/Quad read
45  *               :                         Added the processing to check the data
46  *               :                         count before SPDR read
47  *               :                         Uses DMAC for processing in multiples of 16byte
48  *""FILE COMMENT END""*****************************************************/
```

3.9 Sample Program List “rqspi.c” (2)

```
49    #include <stdio.h>
50    #include <machine.h>
51    #include "iodefine.h"
52    #include "rqspi.h"
53
54    /* ==== Macro definition ==== */
55    #define SPCR_SPI_ENABLE  0x48        /* Always write 1 to bit 3 */
56    #define SPCR_SPI_DISABLE 0x08        /* Always write 1 to bit 3 */
57    #define SPCMD_SPIRW_BIT  0x0070      /* SPI mode and read/program target bit write */
58    #define SPCMD_SPB_BIT    0x0f00      /* Transfer data length (access size) target bit */
59    #define SPCMD_SPB_8BITS  0x0000
60    #define SPCMD_SPB_16BITS 0x0100
61    #define SPCMD_SPB_32BITS 0x0200
62    #define SPCMD_DEFAULT_SET 0xe087
63                  /* bit 15  : clock delay             : SPCKD( 1.5 QSPCLK)    */
64                  /* bit 14  : QSSL negation delay     : SSLND( 2 QSPCLK)      */
65                  /* bit 13  : succeeding access delay : SPND( 2 QSPCLK)       */
66                  /* bit 12  : format                  : MSB first         */
67                  /* bit 11-8: transfer data length    : 8 bits            */
68                  /* bit 7   : QSSL signal hold: Retained after transfer is completed */
69                  /* bit 6-5 : SPI operation mode      : Single-SPI        */
70                  /* bit 4   : read-write              : Program invalid       */
71                  /* bit 3-2 : bit rate                : SPBR divided by 2(36Mbps) */
72                  /* bit 1   : CPOL(QSPCLK polarity)   : 1 when idling     */
73                  /* bit 0   : CPHA(QSPCLK phase)      : Shifts data at the odd edge*/
74                  /*                                   : Latches data at the even edge*/
75    #define SPCMD_FAST_SET 0xe081
76                  /* bit 15  : clock delay             : SPCKD( 1.5 QSPCLK)    */
77                  /* bit 14  : QSSL negation delay     : SSLND( 2 QSPCLK)      */
78                  /* bit 13  : succeeding access delay : SPND( 2 QSPCLK)       */
79                  /* bit 12  : format                  : MSB first         */
80                  /* bit 11-8: transfer data length    : 8 bits            */
81                  /* bit 7   : QSSL signal hold: Retained after transfer is completed */
82                  /* bit 6-5 : SPI operation mode      : Single-SPI        */
83                  /* bit 4   : read-write              : Program invalid       */
84                  /* bit 3-2 : bit rate                : same as SPBR(72Mbps)  */
85                  /* bit 1   : CPOL(QSPCLK polarity)   : 0 when idling     */
86                  /* bit 0   : CPHA(QSPCLK 位相)        : Shifts data at the odd edge*/
87                  /*                                   : Latches data at the even edg*/
88
89    /* ---- Structure to define the RQSPI sequence control information ---- */
90    typedef struct{
91      unsigned short spcmd;           /* SPCMDn register setting */
92      unsigned long  spbmul;          /* SPBMULn register setting */
93      void    *wr_ptr;            /* Address storing the transit data */
94      void    *rd_ptr;            /* Address storing the received data */
95    }RQSPI_CMD_ST;
96
```

3.10 Sample Program List “rqspi.c” (3)

```
97    /* ==== Function prototype declaration ==== */
98    extern void io_dmac_read_SPDR_reg_16burst( void *rdp, int cnt );
99
100   static int io_rqspi_update_SPCMD_reg( int seq );
101   static void io_rqspi_write_SPDR_reg( void *wrp, int cnt, unsigned short bitsz );
102   static void io_rqspi_read_SPDR_reg( void *rdp, int cnt , unsigned short bitsz );
103   static void io_rqspi_rdwr_SPDR_reg( void *wrp, void *rdp, int cnt, unsigned short bitsz );
104
105   /* ==== Variable definition ==== */
106   RQSPI_CMD_ST rqspi_cmd_set[4];
107
108   /*""FUNC COMMENT""*****************************************************************
109    * ID          :
110    * Outline     :  Initialize the RQSPI
111    *-------------------------------------------------------------------------------
112    * Include     : iodefine.h
113    *-------------------------------------------------------------------------------
114    * Declaration : static void io_init_rqspi(void);
115    *-------------------------------------------------------------------------------
116    * Description : Configures the Renesas Quad Serial Peripheral Interface.
117    *             : Configures the RQSPI in master mode, and executes the transfer
118    *             : setting according to the specification of the serial flash memory.
119    *-------------------------------------------------------------------------------
120    * Argument    : void
121    *-------------------------------------------------------------------------------
122    * Return Value: void
123    *-------------------------------------------------------------------------------
124    * Note         : None
125    *""FUNC COMMENT END""*************************************************************/
126   void io_rqspi_initialize(void)
127   {
128     /* ==== Clock supply ==== */
129     CPG.STBCR8.BIT.MSTP82 = 0;
130
131     /* ==== Port ==== */
132     PORT.PFCR3.BIT.PF12MD = 0x06u;  /* QMI/QIO1 */
133     PORT.PFCR2.BIT.PF11MD = 0x06u;  /* QMO/QIO0 */
134     PORT.PFCR2.BIT.PF10MD = 0x06u;  /* QSSL */
135     PORT.PFCR2.BIT.PF9MD  = 0x06u;  /* QSPCLK */
136     PORT.PDCR3.BIT.PD15MD = 0x03u;  /* QIO3 */
137     PORT.PDCR3.BIT.PD14MD = 0x03u;  /* QIO2 */
138
139     /* ==== Disables the SPI function, and initializes the inner state ==== */
140     RQSPI.SPCR.BYTE = SPCR_SPI_DISABLE;
141
142     /* ==== Registers the slave select polarity (SSLP) ==== */
143     RQSPI.SSLP.BIT.SSLP = 0;    /* QSSL signal is Low-active */
144
```

3.11 Sample Program List “rqspi.c” (4)

```
145     /* ==== Registers the pin control (SPPCR) ==== */
146     RQSPI.SPPCR.BYTE = 0x26;
147              /* Sets the output pin idle value to 0 */
148              /* QIO3 is fixed to 1 in single-SPI and dual-SPI operation */
149              /* QIO2 is fixed to 1 in single-SPI and dual-SPI operation */
150              /* Normal operating mode (disables the loop-back) */
151     /* ==== Registers the buffer control (SPBFCR) ==== */
152     RQSPI.SPBFCR.BYTE = 0xC0;/* Transmit buffer empty with 4-byte empty space */
153     RQSPI.SPBFCR.BYTE = 0x25;
154              /* Clear bufffer reset */
155              /* QIO2 is fixed to 1 in single-SPI and dual-SPI operation */
156              /* Receive the receive buffer data count trigger by 16byte */
157              /* (reset the reception trigger count accordingly before transferring) */
158
159     /* ==== Registers the bit rate register (SPBR) ==== */
160     RQSPI.SPBR.BYTE = 0;        /* Base bit rate is 72Mbps (B-clock is 72MHz) */
161
162     /* ==== Registers the clock delay (SPCKD) ==== */
163     RQSPI.SPCKD.BYTE = 0x00;    /* SSL set up time is 1.5 QSPCLK */
164
165     /* ==== Registers the slave select negation delay (SSLND) ==== */
166     RQSPI.SSLND.BYTE = 0x01;    /* SSL hold time is 2 QSPCLK */
167
168     /* ==== Registers the succeeding access delay (SPND) ==== */
169     RQSPI.SPND.BYTE = 0x01;         /* Continuous transfer delay time is 2 QSPCLK */
170
171     /* ==== Command register n (SPCMDn) ==== */
172     RQSPI.SPCMD0.WORD = SPCMD_DEFAULT_SET;     /* (Reset before transferring) */
173     RQSPI.SPCMD1.WORD = SPCMD_DEFAULT_SET;     /* (Reset before transferring) */
174     RQSPI.SPCMD2.WORD = SPCMD_DEFAULT_SET;     /* (Reset before transferring) */
175     RQSPI.SPCMD3.WORD = SPCMD_DEFAULT_SET;     /* (Reset before transferring) */
176   }
```

3.12 Sample Program List “rqspi.c” (5)

```
177  /*""FUNC COMMENT""*****************************************************************
178   * ID           :
179   * Outline      : Update the sequence control information
180   *-------------------------------------------------------------------------------
181   * Include      :
182   *-------------------------------------------------------------------------------
183   * Declaration  : void io_rqspi_set_cmd( int idx, unsigned short mode, void *wrp,
184   *              :                                    void *rdp, unsigned long sz)
185   *-------------------------------------------------------------------------------
186   * Description  : Updates the structure, rqspi_cmd_set, to define the sequence
187   *                control information.
188   *-------------------------------------------------------------------------------
189   * Argument     : int            idx  ; I : Target command register number(0 to 3)
190   *              : unsigned short mode ; I : SPI mode and PW setting to specify
191   *              :                           in the SPCMD
192   *              : void          *wrp  ; I : Address storing the transmit data
193   *              : void          *rdp  ; O : Address storing the receive data
194   *              : unsigned long  sz   ; I : Number of transmit/receive data (bytes)
195   *-------------------------------------------------------------------------------
196   * Return Value : void
197   *-------------------------------------------------------------------------------
198   * Note         :
199   *""FUNC COMMENT END""*************************************************************/
200  void io_rqspi_set_cmd( int idx, unsigned short mode, void *wrp, void *rdp,
201                         unsigned long sz)
202  {
203    RQSPI_CMD_ST *cmd = rqspi_cmd_set;
204    unsigned short bitsz = SPCMD_SPB_8BITS;
205    unsigned long  trncnt = sz;
206
207    /* ---- Set the most appropriate data size ---- */
208    if( ((sz&0x3)==0) && (((int)wrp&0x3)==0) && (((int)rdp&0x3)==0) ){
209      bitsz = SPCMD_SPB_32BITS;
210      trncnt = sz >> 2;
211    }
212    else if( ((sz&0x1)==0) && (((int)wrp&0x1)==0) && (((int)rdp&0x1)==0) ){
213      bitsz = SPCMD_SPB_16BITS;
214      trncnt = sz >> 1;
215    }
216
217    /* ---- Uses the high-speed clock for half-duplex --- */
218    if( mode==SPI_QUAD_RD || mode==SPI_DUAL_RD ){
219      cmd[idx].spcmd  = (mode | SPCMD_FAST_SET | bitsz);
220    }
221    else{
222      cmd[idx].spcmd  = (mode | SPCMD_DEFAULT_SET | bitsz);
223    }
224    cmd[idx].spbmul = trncnt;
225    cmd[idx].wr_ptr = wrp;
226    cmd[idx].rd_ptr = rdp;
227  }
```

3.13 Sample Program List “rqspi.c” (6)

```
/*""FUNC COMMENT""*****************************************************************
 * ID           :
 * Outline      : RQSPI transfer processing
 *-----------------------------------------------------------------------------
 * Include      : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : int io_rqspi_transfer( int seq );
 *-----------------------------------------------------------------------------
 * Description  : Transfers the RQSPI.
 *              : Transfers the RQSPI using the structure, rqspi_cmd_set, that
 *              : defines the sequence control information. Transfers RQSPI.
                  transmits and
 *              : receives the data register by reading/writing for necessary times.
 *-----------------------------------------------------------------------------
 * Argument     : int seq ; I : number of command register for use -1
                                (only for SPCMD0: 0)
 *-----------------------------------------------------------------------------
 * Return Value : 0 : Normal end
 *-----------------------------------------------------------------------------
 * Note         : None
 *""FUNC COMMENT END""*************************************************************/
int io_rqspi_transfer( int seq )
{
  int i, cnt;
  unsigned short spirw, bitsz;

  /* ==== Sets the command register ==== */
  rqspi_cmd_set[seq].spcmd &= ~(0x0080);     /* Add negate settings in the QSSL */
  io_rqspi_update_SPCMD_reg(seq);

  /* ==== Enables the SPI transfer ==== */
  RQSPI.SPCR.BYTE = SPCR_SPI_ENABLE;

  /* ==== Reads and programs the data register ==== */
  for( i=0; i<=seq; i++){
    bitsz = rqspi_cmd_set[i].spcmd & SPCMD_SPB_BIT;
    spirw = rqspi_cmd_set[i].spcmd & SPCMD_SPIRW_BIT;
    cnt   = rqspi_cmd_set[i].spbmul;

    /* ---- Programming in half-duplex --- */
    if( spirw==SPI_QUAD_WR || spirw==SPI_DUAL_WR ){
        io_rqspi_write_SPDR_reg( rqspi_cmd_set[i].wr_ptr, cnt, bitsz);
    }
    /* ---- Reading in half-duplex --- */
    else if( spirw==SPI_QUAD_RD || spirw==SPI_DUAL_RD ){
        if( (bitsz == SPCMD_SPB_32BITS) && ((cnt & 0x3) == 0) ){
            /* Use DMA transfer for reading processingy by maltiple 16bytes */
            io_dmac_read_SPDR_reg_16burst( rqspi_cmd_set[i].rd_ptr, (cnt/4) );
        }
        else{
            io_rqspi_read_SPDR_reg( rqspi_cmd_set[i].rd_ptr, cnt, bitsz);
        }
    }
```

3.14 Sample Program List “rqspi.c” (7)

```
281       /* ---- Full-duplex --- */
282       else{
283           io_rqspi_rdwr_SPDR_reg( rqspi_cmd_set[i].wr_ptr, rqspi_cmd_set[i].rd_ptr,
284                                   cnt, bitsz);
285       }
286     }
287     /* ==== Waits for transfer ending till the QSSL is negated ==== */
288     while( PORT.PFPR0.BIT.PF10PR == 0 ){
289       /* wait */
290     }
291     /* ==== SPI transfer is completed ==== */
292     RQSPI.SPCR.BYTE = SPCR_SPI_DISABLE;
293   }
294   /*""FUNC COMMENT""*****************************************************************
295    * ID           :
296    * Outline      : Update the command register
297    *-------------------------------------------------------------------------------
298    * Include      : iodefine.h
299    *-------------------------------------------------------------------------------
300    * Declaration  : static int io_rqspi_update_SPCMD_reg( int seq );
301    *-------------------------------------------------------------------------------
302    * Description  : Propagates the sequence control information to the sequence control
303    *              : register and command register.
304    *-------------------------------------------------------------------------------
305    * Argument     : int seq ; I : number of command register for use -1 (with only SPCMD0:0)
306    *-------------------------------------------------------------------------------
307    * Return Value : 0 : Normal end
308    *-------------------------------------------------------------------------------
309    * Note         : None
310    *""FUNC COMMENT END""*************************************************************/
311   static int io_rqspi_update_SPCMD_reg( int seq )
312   {
313     RQSPI_CMD_ST *cmd = rqspi_cmd_set;
314
315     /* ==== Sets the sequence control register ==== */
316     RQSPI.SPSCR.BYTE = seq;
317
318     /* ==== Sets the sequence control register ==== */
319     /* ---- CMD3 ---- */
320     RQSPI.SPCMD3.WORD  = cmd[3].spcmd;
321     RQSPI.SPBMUL3.LONG = cmd[3].spbmul;
322     /* ---- CMD2 ---- */
323     RQSPI.SPCMD2.WORD  = cmd[2].spcmd;
324     RQSPI.SPBMUL2.LONG = cmd[2].spbmul;
325     /* ---- CMD1 ---- */
326     RQSPI.SPCMD1.WORD  = cmd[1].spcmd;
327     RQSPI.SPBMUL1.LONG = cmd[1].spbmul;
328     /* ---- CMD0 ---- */
329     RQSPI.SPCMD0.WORD  = cmd[0].spcmd;
330     RQSPI.SPBMUL0.LONG = cmd[0].spbmul;
331
332     return 0;
333   }
```

3.15 Sample Program List “rqspi.c” (8)

```
/*""FUNC COMMENT""*****************************************************************
 * ID           :
 * Outline      : Program the data register (half-duplex)
 *-----------------------------------------------------------------------------
 * Include      : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : static void io_rqspi_write_SPDR_reg( void *wrp, int cnt,
 *              :                                      unsigned short bitsz );
 *-----------------------------------------------------------------------------
 * Description  : Programs the data stored in the argument wrp in the data
 *              : register in the access size specified in the argument bitsz.
                : Repeats the processing for the times specified by the argument cnt.
 *-----------------------------------------------------------------------------
 * Argument     : void          *wrp   ; I : Ponter to the program data
 *              : int            cnt   ; I : Number of times to program
 *              : unsigned short bitsz ; I : Access size to the data register
 *-----------------------------------------------------------------------------
 * Return Value : void
 *-----------------------------------------------------------------------------
 * Note         : None
 *""FUNC COMMENT END""*************************************************************/
static void io_rqspi_write_SPDR_reg( void *wrp, int cnt, unsigned short bitsz )
{
  if( bitsz == SPCMD_SPB_32BITS ){
    unsigned long *wrp_l = (unsigned long *)wrp;
    while( cnt-- ){
        while(RQSPI.SPSR.BIT.SPTEF == 0){
            /* Waits until the transmit buffer becomes empty */
        }
        RQSPI.SPDR.LONG = *wrp_l++;
    }
  }
  else if( bitsz == SPCMD_SPB_16BITS ){
    unsigned short *wrp_w = (unsigned short *)wrp;
    while( cnt-- ){
        while(RQSPI.SPSR.BIT.SPTEF == 0){
            /* Waits until the transmit buffer becomes empty */
        }
        RQSPI.SPDR.WORD = *wrp_w++;
    }
  }
  else{
    unsigned char *wrp_c = (unsigned char *)wrp;
    while( cnt-- ){
        while(RQSPI.SPSR.BIT.SPTEF == 0){
            /* Waits until the transmit buffer becomes empty */
        }
        RQSPI.SPDR.BYTE = *wrp_c++;
    }
  }
}
```

3.16 Sample Program List “rqspi.c” (9)

```
385  /*""FUNC COMMENT""*****************************************************************
386   * ID           :
387   * Outline      : Read the data register (half-duplex)
388   *-------------------------------------------------------------------------------
389   * Include      : iodefine.h
390   *-------------------------------------------------------------------------------
391   * Declaration  : static void io_rqspi_read_SPDR_reg( void *rdp, int cnt,
392   *              :                                      unsigned short bitsz );
393   *-------------------------------------------------------------------------------
394   * Description  : Reads the data register in size specified by the argument bitsz.
395   *              : Stores the data in the buffer area specified by the argument rdp.
396   *              : Repeats the processing for the number of times specified
397                  : by the argument cnt.
398   *-------------------------------------------------------------------------------
399   * Argument     : void          *rdp   ; I : Buffer address to store the read data
400   *              : int            cnt   ; I : Number of times to read
401   *              : unsigned short bitsz ; I : Access size to the data register
402   *-------------------------------------------------------------------------------
403   * Return Value : void
404   *-------------------------------------------------------------------------------
405   * Note         : None
406   *""FUNC COMMENT END""*************************************************************/
407  static void io_rqspi_read_SPDR_reg( void *rdp, int cnt , unsigned short bitsz )
408  {
409    if( bitsz == SPCMD_SPB_32BITS ){
410      unsigned long *rdp_l = (unsigned long *)rdp;
411      while( cnt-- ){
412          while( RQSPI.SPBDCR.BIT.RXBC < 4 ){
413              /* Waits a data reception */
414          }
415          *rdp_l++ = RQSPI.SPDR.LONG;
416      }
417    }
418    else if( bitsz == SPCMD_SPB_16BITS ){
419      unsigned short *rdp_w = (unsigned short *)rdp;
420      while( cnt-- ){
421          while( RQSPI.SPBDCR.BIT.RXBC < 2 ){
422              /* Waits a data reception */
423          }
424          *rdp_w++ = RQSPI.SPDR.WORD;
425      }
426    }
427    else{
428      unsigned char *rdp_c = (unsigned char *)rdp;
429      while( cnt-- ){
430          while( RQSPI.SPBDCR.BIT.RXBC < 1 ){
431              /* Waits a data reception */
432          }
433          *rdp_c++ = RQSPI.SPDR.BYTE;
434      }
435    }
436  }
```

3.17 Sample Program List “rqspi.c” (10)

```
437  /*""FUNC COMMENT""*****************************************************************
438   * ID           :
439   * Outline      : Read/program the data register (full-duplex)
440   *-------------------------------------------------------------------------------
441   * Include      : iodefine.h
442   *-------------------------------------------------------------------------------
443   * Declaration  : static void io_rqspi_rdwr_SPDR_reg( void *wrp, void *rdp,
444   *              :                                    int cnt, unsigned short bitsz );
445   *-------------------------------------------------------------------------------
446   * Description  : Reads the data register in the size specified by the argument
447   *              : bitsz, and stores it in the buffer area assigned by the argument rdp.
448   *              : Repeats the processing for the number of times specified by the
449   *              : argument cont.
450   *-------------------------------------------------------------------------------
451   * Argument     : void          *wrp   ; I : Pointer to write data
452   *              : void          *rdp   ; I : Buffer address to store the read data
453   *              : int            cnt   ; I : Number of times to write/program
454   *              : unsigned short bitsz ; I : Access size to the data register
455   *-------------------------------------------------------------------------------
456   * Return Value : void
457   *-------------------------------------------------------------------------------
458   * Note         : None
459   *""FUNC COMMENT END""*************************************************************/
460  static void io_rqspi_rdwr_SPDR_reg( void *wrp, void *rdp, int cnt, unsigned short bitsz )
461  {
462    unsigned long tmp = 0;
463
464    /* ==== When the access size is 32-bit ==== */
465    if( bitsz == SPCMD_SPB_32BITS ){
466      unsigned long *wrp_l = (unsigned long *)wrp;
467      unsigned long *rdp_l = (unsigned long *)rdp;
468      while( cnt-- ){
469          /* ---- Write processing ---- */
470          while(RQSPI.SPSR.BIT.SPTEF == 0){
471              /* Waits until the transmit buffer becomes empty */
472          }
473          if( wrp != NULL){
474              RQSPI.SPDR.LONG = *wrp_l++;
475          }
476          else{
477              RQSPI.SPDR.LONG = tmp; /* Transmits the dummy data */
478          }
```

3.18 Sample Program List “rqspi.c” (11)

```
479          /* ---- Read processing ---- */
480          while( RQSPI.SPBDCR.BIT.RXBC < 4 ){
481              /* Waits for data reception */
482          }
483          if(rdp != NULL){
484              *rdp_l++ = RQSPI.SPDR.LONG;
485          }
486          else{
487              tmp = RQSPI.SPDR.LONG; /* Receives the dummy data  */
488          }
489       }
490     }
491     /* ==== When the access size is 16-bit ==== */
492     else if( bitsz == SPCMD_SPB_16BITS ){
493       unsigned short *wrp_w = (unsigned short *)wrp;
494       unsigned short *rdp_w = (unsigned short *)rdp;
495       while( cnt-- ){
496          /* ---- Write processing ---- */
497          while(RQSPI.SPSR.BIT.SPTEF == 0){
498              /* Waits until the transmit buffer becomes empty */
499          }
500          if( wrp != NULL){
501              RQSPI.SPDR.WORD = *wrp_w++;
502          }
503          else{
504              RQSPI.SPDR.WORD = (unsigned short)tmp;    /* Transmits the dummy data */
505          }
506          /* ---- Read processing ---- */
507          while( RQSPI.SPBDCR.BIT.RXBC < 2 ){
508              /* Waits for a data reception */
509          }
510          if(rdp != NULL){
511              *rdp_w++ = RQSPI.SPDR.WORD;
512          }
513          else{
514              tmp = RQSPI.SPDR.WORD; /* Receives the dummy data */
515          }
516       }
517     }
```

3.19 Sample Program List “rqspi.c” (12)

```
518     /* ==== When the access size is 8-bit ==== */
519     else{
520       unsigned char *wrp_c = (unsigned char *)wrp;
521       unsigned char *rdp_c = (unsigned char *)rdp;
522       while( cnt-- ){
523           /* ---- Write processing ---- */
524           while(RQSPI.SPSR.BIT.SPTEF == 0){
525               /* Waits until the transmit buffer becomes empty */
526           }
527           if( wrp != NULL){
528               RQSPI.SPDR.BYTE = *wrp_c++;
529           }
530           else{
531               RQSPI.SPDR.BYTE = (unsigned char)tmp; /* Transmits the dummy data */
532           }
533           /* ---- Read processing ---- */
534           while( RQSPI.SPBDCR.BIT.RXBC < 1 ){
535               /* Waits for a data reception */
536           }
537           if(rdp != NULL){
538               *rdp_c++ = RQSPI.SPDR.BYTE;
539           }
540           else{
541               tmp = RQSPI.SPDR.BYTE; /* Receives the dummy data */
542           }
543       }
544     }
545   }
546   /* End of File */
```

3.20 Sample Program List “rqspi.h” (1)

```
1     /***********************************************************************
2     *   DISCLAIMER

      (omitted)

27    ***********************************************************************
28    *   Copyright (C) 2010 Renesas Electronics Corporation. All Rights Reserved.
29    *""FILE COMMENT""*********** Technical reference data *********************
30    *   System Name : SH7267 Sample Program
31    *   File Name   : rqspi.h
32    *   Abstract    : RQSPI quad serial flash memory high-speed read
33    *   Version     : 1.00.00
34    *   Device      : SH7266/SH7267
35    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    *               : C/C++ compiler package for the SuperH RISC engine family
37    *               :                              (Ver.9.03 Release02).
38    *   OS          : None
39    *   H/W Platform: R0K57267(CPU board), Spansion[S25FL032P](serial flash)
40    *   Description :
41    ***********************************************************************
42    *   History     : Aug.20,2010 Ver.1.00.00
43    *""FILE COMMENT END""*****************************************************/
44    #ifndef _RQSPI_H_
45    #define _RQSPI_H_
46
47    /* ==== Macro definition ==== */
48    #define SPI_SINGLE  0x0000 /* Specifies the mode in io_rqapi_set_cmd function
49                                (Single-SPI) */
50    #define SPI_DUAL_WR 0x0020/* Specifies the mode in io_rqapi_set_cmd function
51                                (Dual-SPI, programming) */
52    #define SPI_DUAL_RD 0x0030/* Specifies the mode in io_rqapi_set_cmd function
53                                (Dual-SPI, reading) */
54    #define SPI_QUAD_WR 0x0040/* Specifies the mode in io_rqapi_set_cmd function
55                                (Quad-SPI, programing) */
56    #define SPI_QUAD_RD 0x0050/* Specifies the mode in io_rqapi_set_cmd function
57                                (Quad-SPI, reading) */
58
59    /* ==== Function prototype declaration  ==== */
60    void io_rqspi_initialize(void);
61    void io_rqspi_set_cmd( int idx, unsigned short mode, void *wrp, void *rdp,
62                           unsigned long sz);
63    int io_rqspi_transfer( int seq );
64
65    #endif /* _RQSPI_H_ */
66    /* End of File */
```

3.21 Sample Program List “dmac.c” (1)

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Electronics Corporation and is only
5     *   intended for use with Renesas products.  No other uses are authorized.
6     *
7     *   This software is owned by Renesas Electronics Corporation and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ***********************************************************************
28    *   Copyright (C) 2011 Renesas Electronics Corporation. All Rights Reserved.
29    *""FILE COMMENT""*********** Technical reference data *************************
30    *   System Name : SH7267 Sample Program
31    *   File Name   : dmac.c
32    *   Abstract    : RQSPI quad serial flash memory high-speed read
33    *   Version     : 1.00.00
34    *   Device      : SH7266/SH7267
35    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    *               : C/C++ compiler package for the SuperH RISC engine family
37    *               :                              (Ver.9.03 Release02).
38    *   OS          : None
39    *   H/W Platform: R0K57267(CPU board), Spansion[S25FL032P](serial flash)
40    *   Description :
41    ***********************************************************************
42    *   History     : Apr.14,2011 Ver.1.00.00
43    *""FILE COMMENT END""*****************************************************/
44    #include "iodefine.h"
45
46    /* ==== Prototype declaration ==== */
47    void io_dmac_read_SPDR_reg_16burst( void *rdp, int cnt );
48    static void io_init_dmac4(void *src, void *dst, int count);
49
```

3.22 Sample Program List “dmac.c” (2)

```
/*""FUNC COMMENT""*****************************************************************
 * ID            :
 * Outline       : DMAC setting
 *-----------------------------------------------------------------------------
 * Include       : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration   : void io_dmac_read_SPDR_reg_16burst( void *rdp, int cnt );
 *-----------------------------------------------------------------------------
 * Description   : Initializes channel 4 on the direct memory access controller (DMAC).
 *               : Transfers the receive data from the RQSPI using the on-chip peripheral
 *               : module request (SPRI). Execute this module after setting the
 *               : RQSPI to Dual/Quad-SPI operation (reception).
 *               : Transfer unit is 16bytes (4 x 4).
 *-----------------------------------------------------------------------------
 * Argument      : void          *rdp   ; I : Buffer address to store the read data
 *               : int            cnt   ; I : Number of times to read
 *               : unsigned short bitsz ; I : Access size to the data register
 *-----------------------------------------------------------------------------
 * Return Value  : void
 *-----------------------------------------------------------------------------
 * Note          :
 *""FUNC COMMENT END""*************************************************************/
void io_dmac_read_SPDR_reg_16burst( void *rdp, int cnt )
{
  unsigned long *src = (unsigned long *)&RQSPI.SPDR.LONG;

  /* ==== Sets an receive data count trigger ==== */
  RQSPI.SPBFCR.BIT.RXTRG = 5; /* Larger than 16bytes */
  while( RQSPI.SPBDCR.BIT.RXBC < 16 ){
    /* heck if the SPRFF is set with the trigger count before changing */
  }
  /* ==== Sets DMA transfer ==== */
  io_init_dmac4( src, rdp, cnt );

  /* ==== REnables the RQSPI reception interrupt request ==== */
  RQSPI.SPCR.BIT.SPRIE = 1;

  /* ==== Waits until the transfer is completed ==== */
  while(DMAC.CHCR4.BIT.TE == 0) {
    /* wait */
  }
  RQSPI.SPCR.BIT.SPRIE = 0;   /* Disable receive interrupt request */
  DMAC.CHCR4.BIT.DE = 0;      /* Disable transfer */
  DMAC.CHCR4.BIT.TE = 0;      /* Clear the TE flag */
}

```

3.23 Sample Program List “dmac.c” (3)

```
96    /*""FUNC COMMENT""*****************************************************************
97     * ID            :
98     * Outline       : DMAC setting
99     *-------------------------------------------------------------------------------
100    * Include       : iodefine.h
101    *-------------------------------------------------------------------------------
102    * Declaration   : static void io_init_dmac4(void *src, void *dst, int count,
103    *               :                           unsigned short bitsz);
104    *-------------------------------------------------------------------------------
105    * Description   : Initializes the direct memory access controller (DMAC) channel 4.
106    *               : Transfers the receive data from the RQSPI using the on-chip peripheral
107    *               : module request (SPRI). Does not use the reload function.
108    *-------------------------------------------------------------------------------
109    * Argument      : void *src ; I : Transfer source address
110    *               : void *dst ; O : Transfer destination address
111    *               : int count ; I : Number of times to transfer
112    *-------------------------------------------------------------------------------
113    * Return Value  : void
114    *-------------------------------------------------------------------------------
115    * Note          :
116    *""FUNC COMMENT END""*************************************************************/
117   static void io_init_dmac4(void *src, void *dst, int count)
118   {
119     /* ==== Sets the standby control register 2  ==== */
120     CPG.STBCR2.BIT.MSTP8 = 0;              /* Clears the DMAC module standby  */
121
122     /* ====Disable transferring DMA_ch4 ==== */
123     DMAC.CHCR4.BIT.DE = 0x0;               /* Disable DMA */
124
125     /* ====Sets the DMA source address register_4 (SAR_4) ==== */
126     DMAC.SAR4.LONG = (unsigned long)src;
127
128     /* ====Sets the destination address register_4 (DAR_4) ==== */
129     DMAC.DAR4.LONG = (unsigned long)dst;
130
131     /* ====Sets the DMA transfer count register_4 (DMATCR_4) ==== */
132     DMAC.DMATCR4.LONG = count;
133
134     /* ====Sets the DMA channel control register_4 ( CHCR_4) ==== */
135     DMAC.CHCR4.LONG = 0x02004818;
136          /*
137              bit31    : TC DMATCR 転送：0-----------    One time transfer
138              bit30    : reserve 0
139              bit29    : RLDSAR ON : 0---------------   Invalid reload function (RSAR)
140              bit28    : RLDDAR ON : 0---------------   Invalid reload function (RDAR)
141              bit27    : reserve 0
142              bit26    : DAF : 0-------------------     Not used
143              bit25    : SAF : 1-------------------     16byte transfer as 4byte x 4
144              bit24    : reserve 0
145              bit23    : DO over run0 : 0----------     Not used
146              bit22    : TL TEND low active : 0----     Not used
```

3.24 Sample Program List “dmac.c” (4)

```
147             bit21    : reserve 0
148             bit20    : TEMASK :0-----------------    Not used
149             bit19    : HE :0---------------------    Not used
150             bit18    : HIE :0--------------------    Not used
151             bit17    : AM :0---------------------    Not used
152             bit16    : AL :0---------------------    Not used
153             bit15-14 : DM1:0 DM0:1---------------    Increase destination addresses
154             bit13-12 : SM1:0 SM0:0---------------    Fix source address
155             bit11-8  : RS : B'1000---------------    Transfer source is expansion
156                                                       source selector
157             bit7     : DL : DREQ level : 0 -------    Not used
158             bit6     : DS : DREQ select :0 Low level  Not used
159             bit5     : TB :cycle :0---------------    Cycle steal mode
160             bit4-3   : TS : transfer size : B'11---    16byte transfer
161             bit2     : IE : interrupt enable : 0---    Disable interruptions
162             bit1     : TE : transfer end : 0-------    TClear the TE flag (0-clear after
163                                                       one read is only valid)
164             bit0     : DE : DMA enable bit : 0-----    Disable DMA transfer
165          */
166     /* ====Sets the DMA expansion resource selector 2 (DMARS2) ==== */
167     DMAC.DMARS2.BYTE.CH4 = 0xA2;        /* RQSPI reception */
168
169     /* ----Sets the DMA operation register (DMAOR)---- */
170     DMAC.DMAOR.WORD |= 0x0007;  /* Sets the DME bit. Write 1 in the AE bit and the*/
171                                 /* NMI flag to avoid being cleared the address error*/
172                                 /* flag and the NMIF flag */
173     /* =======Enables transfer DMA_ch4 ======= */
174     DMAC.CHCR4.BIT.DE = 0x1;
175   }
176   /* End of File */
```

4. References

Software Manual

SH-2A/SH-2A-FPU Software Manual Rev. 4.00

The latest version can be downloaded from the Renesas Electronics website.

Hardware Manual

SH7266 Group, SH7267 Group User's Manual: Hardware Rev. 1.00

The latest version can be downloaded from the Renesas Electronics website.

Website and Support

Renesas Electronics website

http://www.renesas.com/

Inquiries

http://www.renesas.com/contact/

Revision History	SH7266/SH7267 Groups Application Note Renesas Squad Serial Peripheral Interface High Speed Read Processing on Serial Flash Memory

Rev.	Date	Description	
		Page	Summary
1.00	Sep. 28, 2011	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
   - "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
   - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
   - "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.1