To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

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

Application Note

Multimedia Processor for Mobile Applications

SPI Interface

EMMA Mobile1

Document No. S19901EJ1V0AN00
Date Published Aug. 2009

PREFACE

Purpose The purpose of this document is to specify the usage of EMMA Mobile1 SPI interface.

Organization This document includes the following:

- Introduction
- Usage of SPI Interface
- Example of SPI Operation
- SPI Driver Function

Notation Here explains the meaning of following words in text:

Note Explanation of item indicated in the text

Caution Information to which user should afford special attention

Remark Supplementary information

Related document The following tables list related documents.

Reference Document

Document Name	Version	Author	Description
S19265EJ1V0UM00_ASMUGIO.pdf	1st Edition	NECEL	EMMA Mobile 1 SMU&GPIO User's manual
S19268EJ1V0UM00_1chip.pdf	1st Edition	NECEL	EMMA Mobile 1 one Chip User's manual
S19261EJ1V0UM00_SPI.pdf	1st Edition	NECEL	EMMA Mobile 1 SPI User's manual
S19255EJ1V0UM00_DMA.PDF	1st Edition	NECEL	EMMA Mobile 1 DMA User's manual

Disclaimers

- **The information contained in this document is subject to change without prior notice in the future. Refer to the latest applicable data sheet(s) and user manual when designing a product for mass production.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC 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 NEC Electronics products listed in this documents or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customers' equipment shall be done under the full responsibility of the customer. NEC Electronics assume no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.

Note)

1. "NEC Electronics" as used in this document means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
2. "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above)
3. All trademarks or registered trademarks are the property of their respective owners. Registered trademarks ® and trademarks™ are not noted in this document.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1 Introduction

1.1 Outline

This document will show users how to use the SPI interface of EMMA Mobile1.

1.2 Development Environment

- Hardware environment of this project is listed as below.

Table 1-1 Hardware Environment

Name	Version	Maker
EMMA Mobile 1 evaluation board (PSKCH2Y-S-0016-01)	-	NEC Electronics
PARTNER-Jet ICE ARM	M20	Kyoto Microcomputer Co. Ltd

- Software used in this project is listed as below.

Table 1-2 Software Environment

Name	Version	Maker
GNUARM Toolchain	V4.3.2	GNU
WJETSET-ARM	V5.10a	Kyoto Microcomputer Co. Ltd

Chapter 2 Usage of SPI Interface

2.1 Outline of SPI Data Transfer

There are four modes of SPI data transfer operation:

- CPU-master,
- CPU-slave,
- DMA-master
- DMA-slave.

Dual port SRAM with 32 bits × 32 words is used for transmission and reception.

Normal SPI data transfer procedure flow chart is shown as below.

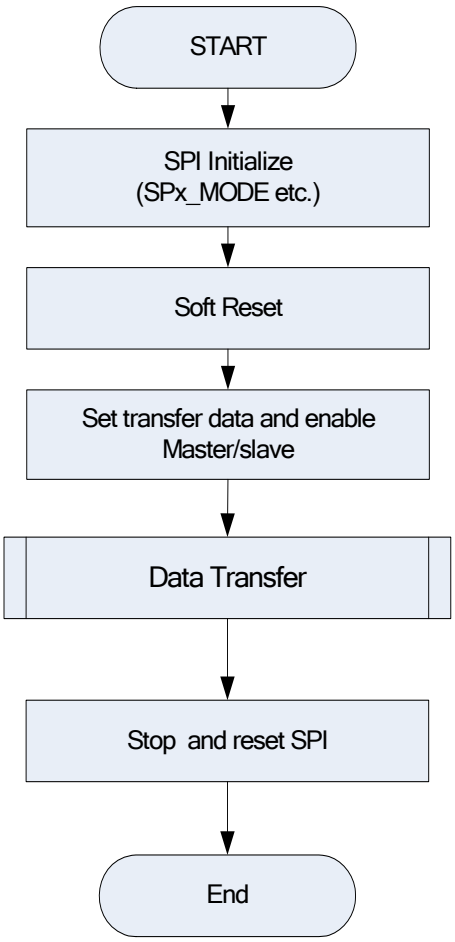

Figure 2-1 Normal SPI Data Transfer Flow

Note:

1. About the explanation of all the SPI registers mentioned in this document, please refer to "**CHAPTER 3 REGISTERS**" of "**EMMA Mobile 1 SPI User's Manual**".
2. More details about the transfer operation in every mode (CPU-master, CPU-slave, DMA-master and DMA-slave), please refer to the "**CHAPTER 5 USAGE**" of "**EMMA Mobile 1 SPI User's Manual**".

2.2 Detail of Normal SPI Data Transfer Procedure

2.2.1 SPI Initialize

- Set SPI clock.
  More about clock control please refer to "**3.2 Register Functions (ASMU)**" of "**EMMA Mobile 1 SMU&GPIO User's Manual**".
  Register list:

APBCLKCTRL0	//specifies whether to enable automatic control for PCLK of SPI2
APBCLKCTRL1	//specifies whether to enable automatic control for PCLK of SPI0, 1
CLKCTRL	//specifies whether to enable automatic control for SCLK of SPI0,1,2
GCLKCTRL3ENA	// enables writing to each bit of the GCLKCTRL3 register
GCLKCTRL3	// specifies whether to supply a clock for SPI0,1,2
DIVSP0SCLK	// specifies the division factor for SP0_SCLK
DIVSP1SCLK	// specifies the division factor for SP1_SCLK
DIVSP2SCLK	// specifies the division factor for SP2_SCLK

- Switch SPI pin function and enable input function.
  The details, please refer to "**CHAPTER 8 ALTERNATE PIN FUNCTION SWITCHING**" of "**EMMA Mobile 1 One Chip User's Manual**".
  Register list:

CHG_PINSEL_SP0	// switches the SPI0 pin functions
CHG_PINSEL_G64	// switches the SPI1 pin functions
CHG_PINSEL_G80	// switches the SPI1 pin functions
CHG_PINSEL_DTV	// switches the SPI2 pin functions
CHG_PULL1	// selects pull-up or pull-down and enable inputs for SPI0
CHG_PULL_G72	// selects pull-up or pull-down and enable inputs for SPI1
CHG_PULL0	// selects pull-up or pull-down and enable inputs for SPI2

- Set SPI mode, including set operation mode (CPU or DMA; master or slave), select CSx, set transfer bit number, select the polarity of SCLK, and set SPI register data transfer start signal.
  Register list:

SPx_MODE	// specifies the operation mode of the SPx module
SPx_POL	// selects the polarity of SCLK and CS
SPx_TIECS	// fixes the output of SPx_CS0 to SPx_CS5.
SPx_CONTROL2	// controls fixed-length transfer in DMA master mode

  (x = 0, 1 or 2)

2.2.2 Soft Reset

SPI Soft reset by set the "RST" of the control register "SPx_CONTROL" to "1". And set it to "0" to cancel reset state.
Register list:

SPx_CONTROL (x = 0, 1 or 2)

More detail about SPI soft reset, please refer to "**4.6 Reset Control**" of "**EMMA Mobile 1 SPI User's Manual**".

2.2.3 SPI Start

Via this step, the transfer data will be specified. Then start the transfer via setting "WRT", "RD", and "START" of register "SPx_CONTROL".

Register List:

SPx_CONTROL (x = 0, 1 or 2)

If using DMA for SPI writing operation, the SPI interrupt should be enabled and the following settings of DMA must be done before starting transfer operation via setting register "SPx_CONTROL".

- **DMA Initialization(reset and open clock)**

  Register list:

  DMA reset setting related register:

  ASMU_RESETREQ0ENA

  ASMU_RESETREQ0

  DMA clock setting related register:

  ASMU_GCLKCTRL0

  ASMU_GCLKCTRL0ENA

- **DMA setting**

  Register list:

  P2M DMA transfer setting related register:

  DMA_P2M_LCHx_AADD

  DMA_P2M_LCHx_BADD

  DMA_P2M_LCHx_BOFF

  DMA_P2M_LCHx_BSIZE

  DMA_P2M_LCHx_BSIZE_COUNT

  DMA_P2M_LCHx_LENG

  DMA_P2M_LCHx_MODE

  M2P DMA transfer setting related register:

  DMA_M2P_LCHx_BADD

  DMA_M2P_LCHx_AADD

  DMA_M2P_LCHx_AOFF

  DMA_M2P_LCHx_ASIZE

  DMA_M2P_LCHx_ASIZE_COUNT

  DMA_M2P_LCHx_LENG

  DMA_M2P_LCHx_MODE

  (x= 12, 13, 14 corresponding to SPI 0, 1, 2)

- **DMA interrupt setup**

  Register list:

  P2M Clear DMA interrupt source related register:

  DMA_P2M_PE0_LCH12LCH14_INT_REQ_CL
  DMA_P2M_PE0_LCH12LCH14_INT_ENABLE
  DMA_P2M_PE0_LCH12LCH14_INT_ENABLE_CL

  M2P DMA interrupt source related register:

  DMA_M2P_PE0_LCH12LCH14_INT_REQ_CL
  DMA_M2P_PE0_LCH12LCH14_INT_ENABLE
  DMA_M2P_PE0_LCH12LCH14_INT_ENABLE_CL

- **DMA Start**

  Register list:

  P2M start register

  DMA_P2M_CONT

  M2P start register

  DMA_M2P_CONT

More about DMA operations please refer to "**CHAPTER 3 DESCRIPTION OF FUNCTIONS**" and "**EMMA Mobile 1 DMA Application Note**".

2.2.4 Data transfer

After setting "WRT/RD", and "START" of register "SPx_CONTROL", the data transfer operation start, and interrupt will occur to indicate the ending. The interrupt is different according to the transfer result (normal or abnormal) and operation mode (CPU-master, CPU-slave, DMA-master and DMA-slave).

2.2.5 SPI Interrupt Sources

SPI can issue seven types of interrupt.

TX_STOP	// Indicates that data is no longer stored in the transmit FIFO
RX_STOP	// Indicates that the amount of received data has reached the value set to the RX_FIFO_FULL bit of the SPx_CONTROL2 register.
TERR	// Indicates that the number of SPx_SCLK_I cycles does not match the value set to the NB_A bit of the SPx_MODE register.
RDV	// Indicates that reception of 1 frame is complete in CPU mode.
END	// Indicates that transmission and reception of one frame is complete in CPU mode.
TX_UDR	// Indicates that an underrun has occurred in the transmit FIFO.
RX_OVR	// Indicates that an overrun has occurred in the receive FIFO.

Please refer to "**4.2 Interrupt Generation**" of "**EMMA Mobile 1 SPI User's Manual".**

- SPI Module Interrupt

  Clear the interrupt source of TX_STOP, RX_STOP, TERR, RDV, END, TX_EDR and RX_OVR. After that, prohibits issue of the interrupt request to TX_STOP, RX_STOP, TERR, RDV, END, TX_EDR and RX_OVR.

  Register list:

SPx_FFCLR	// clears interrupt sources
SPx_ENCLR	// disables the issuance of interrupt requests
SPx_ENSET	// enables the issuance of interrupt requests.

  (x = 0, 1 or 2)

- SPI Interrupt

  Register list:

INT_IT0_IDS0	// enable SPI interrupt

Chapter 3 Example of SPI Operation

3.1 SPI CPU Example

3.1.1 Outline of SPI CPU Example

This example is designed for the SPI data transfer in CPU master mode.
In RTC setting example, set the RTC time in PMIC by SPI0.
In RTC reading example, read RTC time in PMIC by SPI0.
Figure 3-1 shows the circuit of the example.

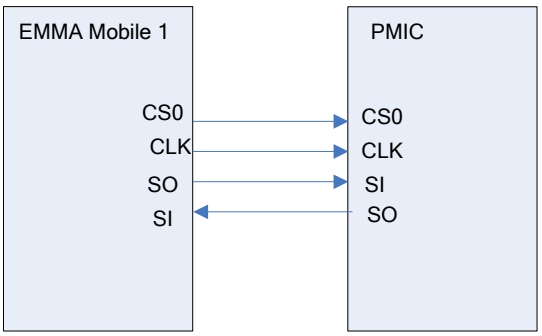

Figure 3-1 Circuit of the SPI CPU Master Mode Data Transfer Example

More details about the usage of PMIC registers please refer to the data sheet of the PMIC

3.1.2 Flow of Data Transfer Procedure

The figure 3-2 shows the procedure of SPI CPU master mode test.

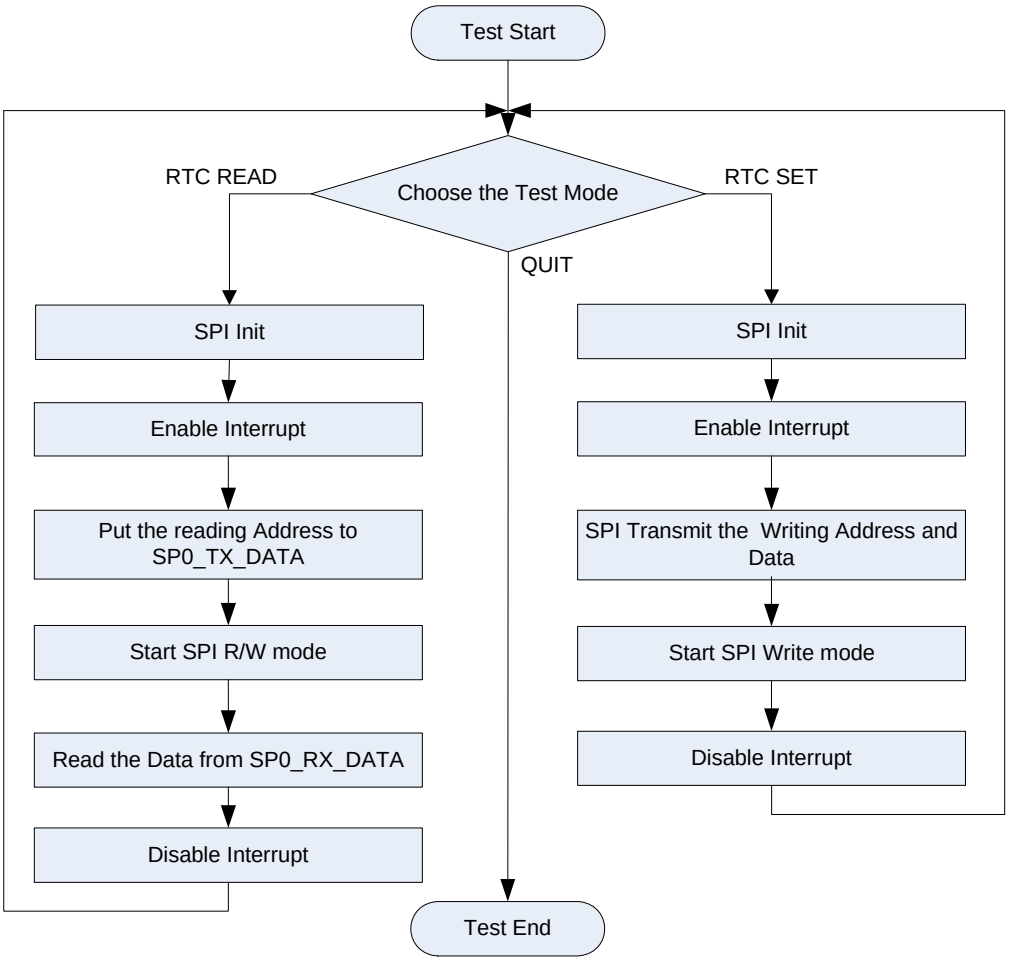

Figure 3-2 Procedure of SPI CPU Master Mode Test

3.1.3 Detail of RTC Reading Example Procedure

- **SPI Initialize**

1. Set SPI0 clock and cancel reset state

More about clock control please refer to "**3.2 Register Functions (ASMU)**" of "**EMMA Mobile 1 SMU&GPIO User's Manual**".

Register list:

```
APBCLKCTRL1[7]        // 0: Disable automatic control ; 1: Enable automatic control
CLKCTRL[2]            // 0: Disable automatic control ; 1: Enable automatic control
GCLKCTRL3ENA[22:21]   // 0: Disable setting; 1: Enable setting
GCLKCTRL3[22:21]      // 0: Close clock; 1: Open clock
DIVSP0SCLK = 72H      // SPI_SCLK = 229.376 MHz / 32 = 7.168 MH
RESETREQ1[22]         // 0: Reset; 1: Cancel reset
```

2. Switch SPI0 pin function and enable input function.

The details, please refer to "**CHAPTER 8 ALTERNATE PIN FUNCTION SWITCHING**" of "**EMMA Mobile 1 One Chip User's Manual**".

Register list:

```
CHG_PINSEL_SP0  = 0
CHG_PULL1[32:16] = 1511H
```

3. Set SPI0 mode, including set operation mode (CPU and master), select CS0, set transfer bit number 16, select the polarity of SCLK, and set SPI register data transfer start signal.

Register list:

```
SP0_MODE = 0f00H
SP0_POL    = 7004H
SP0_TIECS = 0
SP0_CONTROL2 = 0
```

4. SPI0 soft reset

Register list:

```
SP0_CONTROL[8] = 1;          //SPI0 soft reset
SP0_CONTROL[8] = 0;          //Cancel SPI0 soft reset
```

- **Enable the Interrupt**

Enable the SPI0 interrupt: Add the SPI0 interrupt handle function address into the interrupt handler hook function.

Register List:

```
SP0_ENCLR = 0x7f      //Mask all interrupt of SPI
SP0_FFCLR = 0x7f      //Clear interrupt
SP0_ENSET = 0x13      //Enable TERR, TX_UDR, RX_OVR interrupt
IT0_IEN0[24] = 1;     //Enable SPI0
```

- **Transmit the Reading Address**

In the RTC reading test, will use R/W mode. In CPU master mode transfer, if error occurs, interrupt will occur to indicate transmission error. If bit 0 and bit 6 of SPI0_CONTROL register are both ‘0’, the transmission ended normally.

1. Write the reading address to SPI0_TX_DATA,
2. Set bit[3:0] of SPI0_CONTROL to 0DH, will start R/W mode.

- **Read Received Data**

  Read the received data by reading SPI0_RX_DATA.

- **SPI0 End**

  After the data transmit, disable the interrupt before quit from this test.

  Register List and configuration:

  SP0_FFCLR = 7FH;

  SP0_ENCLR = 7FH;

  IT0_IDS0[24] = 1;

- **SPI0 Transfer Timing**

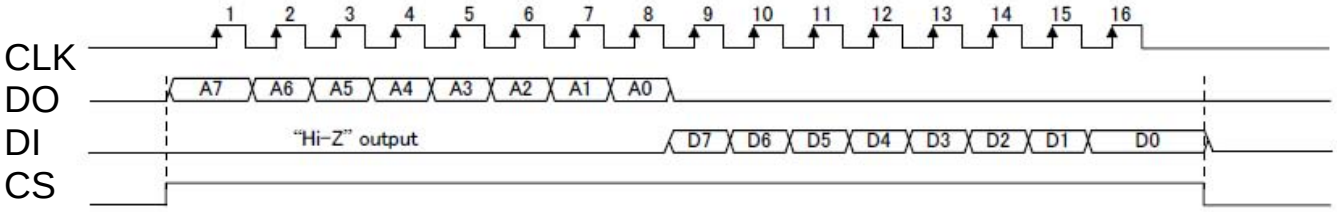

Figure 3-3 SPI R/W Mode Timing

3.1.4 Detail of RTC Setting Example Procedure

- **SPI Init**

1. Set SPI0 clock and cancel reset state.

More about clock control please refer to "**3.2 Register Functions (ASMU)**" of "**EMMA Mobile 1 SMU&GPIO User's Manual**".

Register list:

```
APBCLKCTRL1[7]      // 0: Disables automatic control ; 1: Enables automatic control
CLKCTRL[2]          // 0: Disables automatic control ; 1: Enables automatic control
GCLKCTRL3ENA[22:21] // 0: Disable setting; 1: Enable setting
GCLKCTRL3[22:21]    // 0: Close clock; 1: Open clock
DIVSP0SCLK = 72H    // SPI_SCLK = 229.376 MHz / 32 = 7.168 MH
RESETREQ1[22]       // 0: Reset; 1: Cancel reset
```

2. Switch SPI0 pin function and enable input function.

The details, please refer to "**CHAPTER 8 ALTERNATE PIN FUNCTION SWITCHING**" of "**EMMA Mobile 1 One Chip User's Manual**".

Register list:

```
CHG_PINSEL_SP0  = 0
CHG_PULL1[32:16] = 1511H
```

3. Set SPI0 mode, including set operation mode (CPU and master), select CS0, set transfer bit number 16, select the polarity of SCLK, and set SPI register data transfer start signal.

Register list:

```
SP0_MODE = 0f00H
SP0_POL    = 7004H
SP0_TIECS = 0
SP0_CONTROL2 = 0
```

4. SPI0 soft reset

Register list:

```
SP0_CONTROL[8] = 1;     //SPI0 soft reset
SP0_CONTROL[8] = 0;     //Cancel SPI0 soft reset
```

- **Enable the SPI interrupt**

Enable the SPI0 interrupt: Add the SPI0 interrupt handler function address into the interrupt handler hook function.

Register List:

```
SP0_ENCLR = 0x7f        //Mask all interrupt of SPI
SP0_FFCLR = 0x7f        //Clear interrupt
SP0_ENSET = 0x13        //Enable TERR, TX_UDR, RX_OVR interrupt
IT0_IEN0[24] = 1;       //Enable SPI0
```

- **Data Transmit**

  In the RTC setting test, start the SPI by setting the SP0_CONTROL to 09H, only enable the transmission. In CPU master mode transfer, if error occurs, interrupt will occur to indicate transmission error. If bit 0 of SPI0_CONTROL register is '0', the transmission ended normally.

- **SPI0 end**

  After the data transmit, disable the interrupt before quit from this test.

  Register List and configuration:

  SP0_FFCLR = 7FH;

  SP0_ENCLR = 7FH;

  IT0_IDS0[24] = 1;

- **SPI0 Transfer Timing**

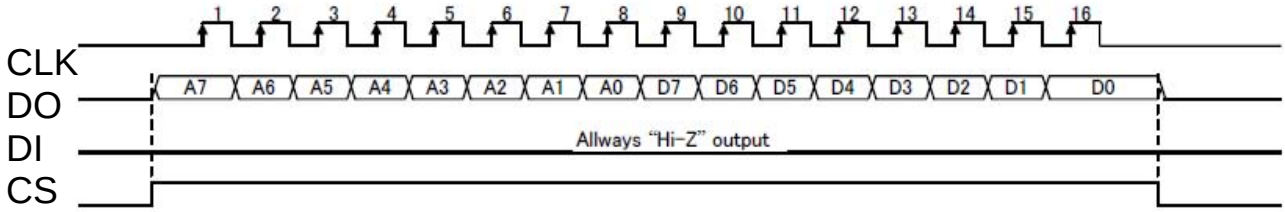

Figure 3-4 SPI Write Mode Timing

3.2 SPI DMA Example

3.2.1 Outline of SPI DMA Example

This example is designed for the SPI data transfer in DMA master send and DMA slave receive mode[NOTE1]. First, make sure the hardware connect.

SPI1_CS------SPI2_CS,
SPI1_CLK----SPI2_CLK,
SPI1_SO-----SPI2_SI,
SPI1_SI-------SPI2_SO,

Figure 3-5 shows the structure of this example.

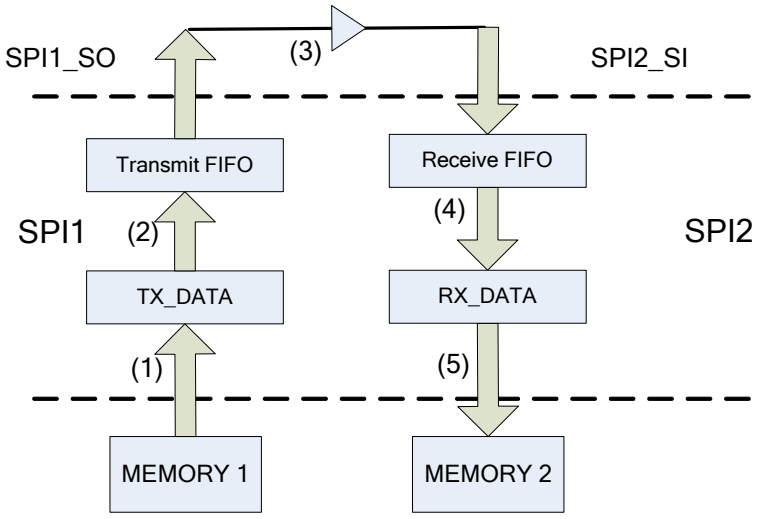

Figure 3-5 Structure of SPI DMA Mode Data Transfer Example

1) The data stored in memory 1 are written to SP1_TX_DATA with DMA M2P channel 13 by DMA controller,
2) The data in TX_DATA register will automatically send to transmit FIFO by SPI controller,
3) Start SPI transfer, the data will transmit from SPI1 to SPI2.
4) The data in RX_DATA register will automatically read from transmit FIFO by SPI controller,
5) The data in RX_DATA are written to memory 2 with DMA P2M channel 14 by DMA controller.

With comparing the data in Memory1 and Memory2, can judge whether the SPI transfer is normal or not.

Note

This example test SPI data transfer in DMA master send and slave receive mode. About SPI data transfer in DMA master receive and slave send mode, only need to modify SPI mode register and SPI control2 register, and other registers is set the same to this example.

3.2.2 Flow of Data Transfer Procedure

The figure 3-6 shows the procedure of SPI-DMA data transfer operation

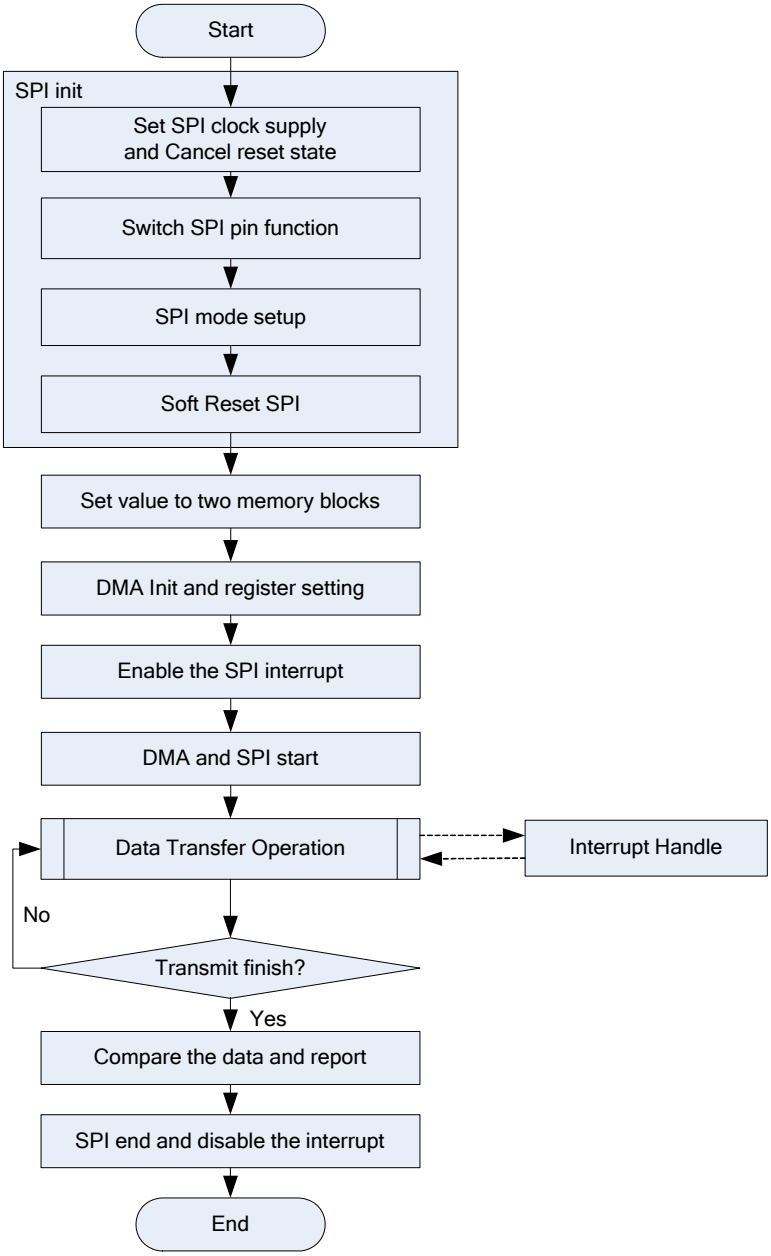

Figure 3-6 Procedure of SPI-DMA Data Transfer Operation

3.2.3 Detail of Data Transfer Procedure

- **SPI Initialize**

1. Set SPI clock and cancel reset state

More about clock control please refer to "**3.2 Register Functions (ASMU)**" of "**EMMA Mobile 1 SMU&GPIO User's Manual**".

Register list:

```
APBCLKCTRL1[8]          // 0: Disable automatic control ; 1: Enable ---PCLK of SPI1
APBCLKCTRL0[7]          // 0: Disable automatic control ; 1: Enable ---PCLK of SPI2
CLKCTRL[4:3]            // 0: Disable automatic control ; 1: Enable ---SCLK of SPI1,2
GCLKCTRL3ENA[26:23]     // 0: Disable setting; 1: Enable setting
GCLKCTRL3[26:23]        // 0: Close clock; 1: Open clock---PCLK and SCLK of SPI1,2
DIVSP1SCLK = 72H        // SPI1_SCLK = 229.376 MHz / 32 = 7.168 MH
DIVSP2SCLK = 72H        // SPI2_SCLK = 229.376 MHz / 32 = 7.168 MH
RESETREQ1[24:23]        // 0: Reset; 1: Cancel reset
```

2. Switch SPI pin function and enable input function.

The details about registers function, please refer to "**CHAPTER 8 ALTERNATE PIN FUNCTION SWITCHING**" of "**EMMA Mobile 1 One Chip User's Manual**".

Register list:

```
CHG_PINSEL_G64[25:18]   = aaH;          //switch pin to the SPI1 ordinary function
CHG_PULL_G72[31:4]      = 555_5555H;    //enable SPI1 input function
CHG_PINSEL_DTV          = 01H;          //switch pin to the SPI2 ordinary function
CHG_PULL0[23:8]         = 5555H;        //enable SPI2 input function
```

3. SPI setup

Set operation mode (CPU or DMA; master or slave), select CS, set transfer bit number, select the polarity of SCLK, and set SPI register data transfer start signal.

The details about SPI registers, please refer to "**CHAPTER 3 REGISTERS**" of "**EMMA Mobile 1 SPI User's Manual**".

Register list:

```
SP1_MODE        = 1F01H;        //SPI1, CS0, 32bit DMA master
SP1_POL         = F000H;
SP1_CONTROL2 = 200H;            //Stop transfer when the transmit FIFO becomes
                                empty during DMA master transmission.
SP2_MODE        = 1F03H;        //SPI2, CS0, 32bit DMA slave
SP2_POL         = F000H;
SP2_CONTROL2 = 0;
```

4. SPI soft reset

Register list:

```
SP1_CONTROL[8] = 1;             //SPI1 soft reset
```

```
SP1_CONTROL[8] = 1;          //SPI1 soft reset
SP2_CONTROL[8] = 0;          //Cancel SPI2 soft reset
SP2_CONTROL[8] = 0;          //Cancel SPI2 soft reset
```

CAUTION:

During SPI DMA data transfer, please don't use soft reset function. In DMA mode, SPI soft reset will be used in SPI init part or after SPI transfer finish.

- **Set value to the two memory blocks**

In order to differentiate the two memory blocks, set different value to them.

Memory 1: all set 5a5a_5a5aH.

Memory 2: all set 1111_1111H.

- **DMA Init and registers setting**

Set parameters of DMA M2P and P2M transmission. For example, source data address, offset, block size, data length etc.

Register list and configuration:

M2P channel 13 for SPI1:

```
DMA_M2P_LCH13_AADD = 3100_0000H;
DMA_M2P_LCH13_ASIZE = 0000_0200H;
DMA_M2P_LCH13_AOFF = 0;
DMA_M2P_LCH13_LENG = 0000_0200H;
DMA_M2P_LCH13_BADD = SP1_TX_DATA;
DMA_M2P_LCH13_MODE = e4e4_0000H;
```

P2M channel 14 for SPI2 :

```
DMA_P2M_LCH14_AADD = SP2_RX_DATA;
DMA_P2M_LCH14_ASIZE = 0000_0200H;
DMA_P2M_LCH14_AOFF = 0;
DMA_P2M_LCH14_LENG = 0000_0200H;
DMA_P2M_LCH14_BADD = 3100_2000H;
DMA_P2M_LCH14_MODE = e4e4_0000H;
```

- **Enable SPI interrupt**

Enable the issuance of interrupt request of SPI1 and SPI2. in this example, enable three error interrupt: TERR, TX_UDR, RX_OVR

Register list and configuration:

```
SP1_FFCLR = 0000_007FH;      //Clear all interrupt source of SPI1
SP1_ENCLR = 0000_007FH;      //Disable all issuance of interrupt of SPI1
SP1_ENSET = 0000_0013H;      //Enable TERR, TX_UDR, RX_OVR 0f SPI1
SP2_FFCLR = 0000_007FH;      //Clear all interrupt source of SPI2
SP2_ENCLR = 0000_007FH;      //Disable all issuance of interrupt of SPI2
```

```
SP2_ENSET = 0000_0013H;        //Enable TERR, TX_UDR, RX_OVR of SPI2
IT0_IEN0[25]  = 1;             //Enable SPI1 interrupt
IT0_IEN1[10]  = 1;             //Enable SPI2 interrupt
```

- **DMA Start**

After complete setting of DMA parameters and enable interrupt source, configure control register “DMA_M2P_CONT” and “DMA_P2M_CONT” to active DMA transfer.

Register list and configuration:

```
DMA_M2P_CONT[13] = 1B;
DMA_P2M_CONT[14] = 1B;
```

- **SPI Start**

After the SPI and DMA registers setting and DMA starting, start the SPI to enable the data transfer.

Register list and configuration:

```
SP1_CONTROL = 09H
SP2_CONTROL = 05H
```

- **Data Transfer**

When all the data specified with the length are transmitted, the DMA Length interrupt will be issued. After received the DMA Length interrupt signal, “TX_EMP” of register “SPx_CONTROL” will be checked to adjust whether the transmit FIFO is empty. If it is empty, the SPI DMA transmission procedure will be end normally.

- **Data Compare**

After the data transfer, compare whether the data in two memory blocks is same or not.

- **SPI end**

After the data transfer and data compare, Disable and clear SPI interrupt.

Register list and configuration:

```
SP1_ENCLR = 7FH;          //Disable SPI1 issuance of interrupt
SP1_FFCLR = 7FH;          //Clear SPI1 source
SP2_ENCLR = 7FH;          //Disable SPI2 issuance of interrupt
SP2_FFCLR = 7FH;          //Clear SPI2 source

IT0_IDS0[25]  = 1;        //Disable SPI1 interrupt
IT0_IDS1[10]  = 1;        //Disable SPI2 interrupt
```

APPENDIX A SPI Driver Function

A.1 Function List

The following table shows the SPI driver interface functions:

Table A-1 SPI Driver Function List

Class	Function Name	Function Detail
External function	em1_spi_set_sclk	Set SPI the clock of SPI_CLK
	em1_spi_setup	SPI setup
	em1_spi_start	SPI start
	em1_spi_soft_reset	SPI soft reset
	em1_spi_end	SPI end
	em1_spi_get_status	Get SPI status
	em1_spi_spi0_irq	SPI0 interrupt handle
	em1_spi_spi1_irq	SPI1 interrupt handle
	em1_spi_spi2_irq	SPI2 interrupt handle
	em1_spi_set_irq	Enable SPI interrupt
	em1_spi_clear_irq	Clear SPI interrupt source
	em1_spi_send	SPI send data
	em1_spi_rw	SPI send address and receive data
	em1_spi_receive	SPI receive data

A.2 Global Variable Define

Table A-2 Global Variable Define

Name	Type	Detail
f_spi_test	BOOL	The flag of spi test
f_spi_cpu_test	BOOL	The flag of spi cpu mode test
f_spi_dma_test	BOOL	The flag of spi dma mode test

A.3 Structure Define

Table A-3 Structure Define

Structure Name	Detail
SPI_SETUP_ST	SPI register sturcture

A.3.1 SPI_SETUP_ST

Table A-4 Structure of SPI_SETUP_ST

Member	Detail
spi_n	SPI channel, (SPI0, SPI1 or SPI2)
cs	SPI cs channel,(cs0,1,2…)
mode	The operation mode
pol	Select the polarity of SCLK and cs signals
tiecs	Fixes the output of SPx_CS0 to SPx_CS5
enset	Permits to issue an interrupt request
cont2	Control fixed-length transfer in DMA master mode

A.4 Function Details

A.4.1 Set SPI SCLK Function

[Function Name]

em1_spi_set_sclk

[Format]

DRV_RESULT em1_spi_set_sclk (uchar spi_n, uint sclk);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number
sclk	uint	I	The clock of SPI CLK

[Function Return]

DRV_OK

DRV_ERR_PARAM

[Flow Chart]

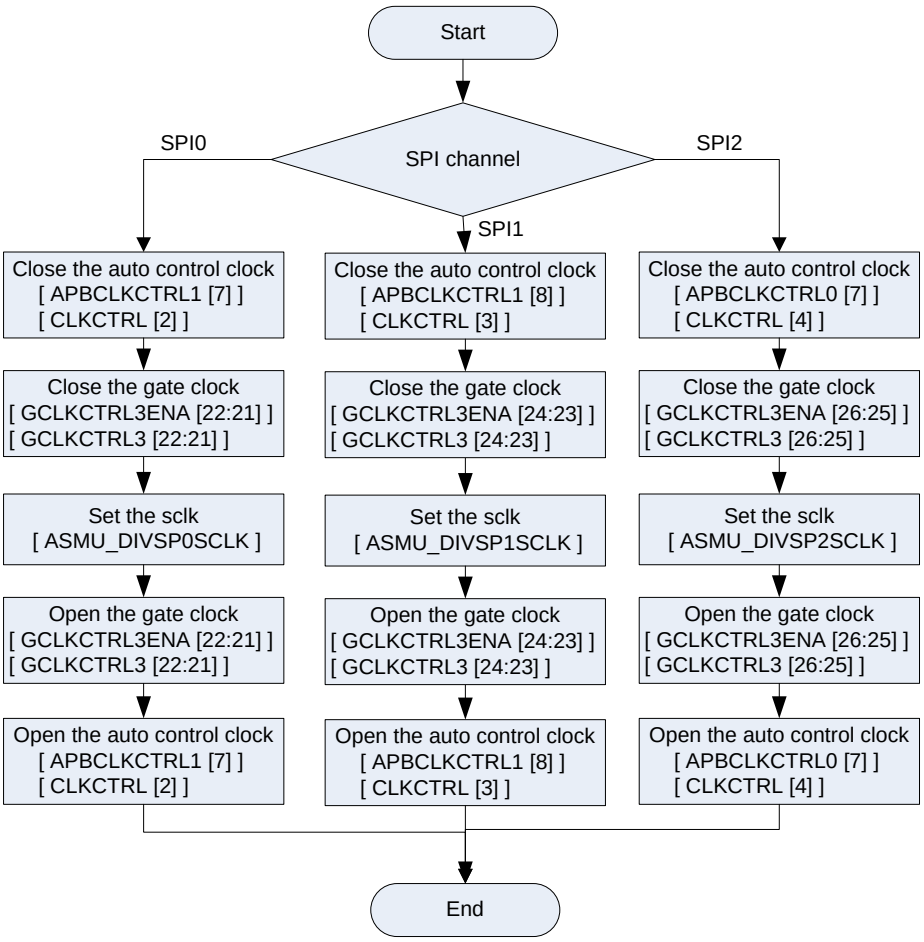

Figure A-1 Stop Display

A.4.2 SPI Setup Function

[Function Name]

em1_spi_setup

[Format]

DRV_RESULT em1_spi_setup(SPI_SETUP_ST* spi_st);

[Argument]

Parameter	Type	I/O	Detail
spi_st	SPI_SETUP_S	I	Spi setup param

[Function Return]

DRV_OK, DRV_ERR_PARAM

[Flow Chart]

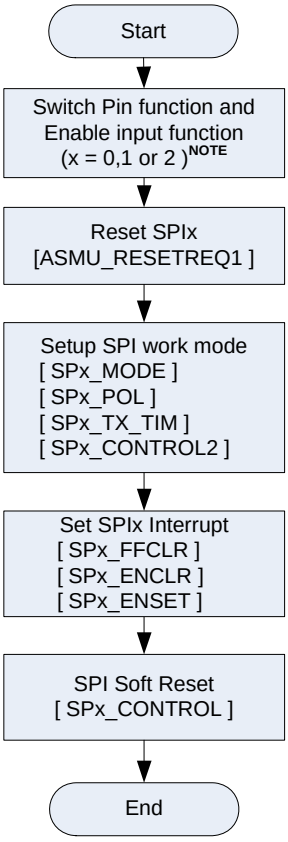

Figure A-2 Stop Display

[Note]

1) SPI0 switch pin function by setting register CHG_PINSEL_SP0
SPI0 switch pin function by setting register CHG_PINSEL_G64 and CHG_PINSEL_G80
SPI0 switch pin function by setting register CHG_PINSEL_DTV
Enable SPI0 pin input function by setting register CHG_PULL1
Enable SPI1 pin input function by setting register CHG_PULL_G72

Enable SPI2 pin input function by setting register CHG_PULL0

A.4.3 SPI Start

[Function Name]

em1_spi_start

[Format]

DRV_RESULT em1_spi_start(uchar spi_n, uchar mode);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number
mode	uchar	I	SPI start mode(transfer or receive)

[Function Return]

DRV_OK
DRV_ERR_PARAM

[Flow Chart]

None

[Note]

None

A.4.4 SPI Soft Reset

[Function Name]

em1_spi_soft_reset

[Format]

DRV_RESULT em1_spi_soft_reset(uchar spi_n);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number

[Function Return]

DRV_OK

DRV_ERR_PARAM

[Flow Chart]

None

[Note]

None

A.4.5 SPI End

[Function Name]

em1_spi_end

[Format]

DRV_RESULT em1_spi_end(uchar spi_n,uchar dma_flg);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number
dma_flag	uchar	I	Mode flag

[Function Return]

DRV_OK
DRV_ERR_PARAM

[Flow Chart]

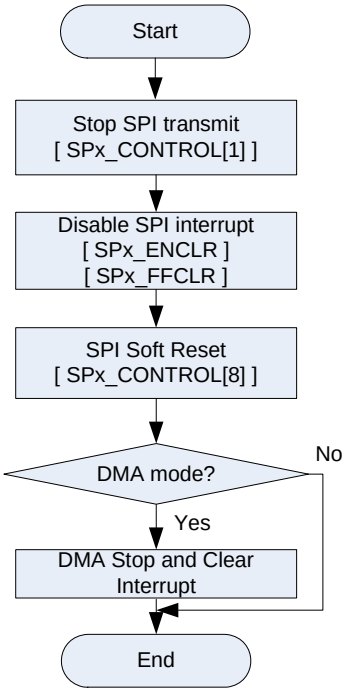

Figure A-3 SPI End

[Note]

None

A.4.6 Get SPI Status Function

[Function Name]

em1_spi_get_status

[Format]

uint em1_spi_get_status(uchar spi_n);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number

[Function Return]

DRV_ERR_PARAM
SPI status register value

[Flow Chart]

None

[Note]

None

A.4.7 SPI0 Interrupt Handle

[Function Name]

em1_spi0_irq

[Format]

void em1_spi0_irq (void);

[Argument]

None

[Function Return]

None

[Flow Chart]

None

[Note]

None

A.4.8 SPI1 Interrupt Handle

[Function Name]

em1_spi1_irq

[Format]

void em1_spi1_irq (void);

[Argument]

None

[Function Return]

None

[Flow Chart]

None

[Note]

None

A.4.9 SPI2 Interrupt Handle

[Function Name]

em1_spi2_irq

[Format]

void em1_spi2_irq (void);

[Argument]

None

[Function Return]

None

[Flow Chart]

None

[Note]

None

A.4.10 SPI Interrupt Setup

[Function Name]

em1_spi_set_irq

[Format]

DRV_RESULT em1_spi_set_irq(uchar spi_n);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number

[Function Return]

DRV_OK
DRV_ERR_STATE
DRV_ERR_PARAM

[Flow Chart]

None

[Note]

None

A.4.11 SPI Interrupt Clear

[Function Name]

em1_spi_clear_irq

[Format]

DRV_RESULT em1_spi_clear_irq (uchar spi_n);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number

[Function Return]

DRV_OK
DRV_ERR_PARAM

[Flow Chart]

None

[Note]

None

A.4.12 SPI Send

[Function Name]

em1_spi_send

[Format]

DRV_RESULT em1_spi_send_single_word (uchar spi_n, uint data);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number
data	uint	I	Transfer data

[Function Return]

DRV_OK
DRV_ERR_PARAM
DRV_ERR_STATE

[Flow Chart]

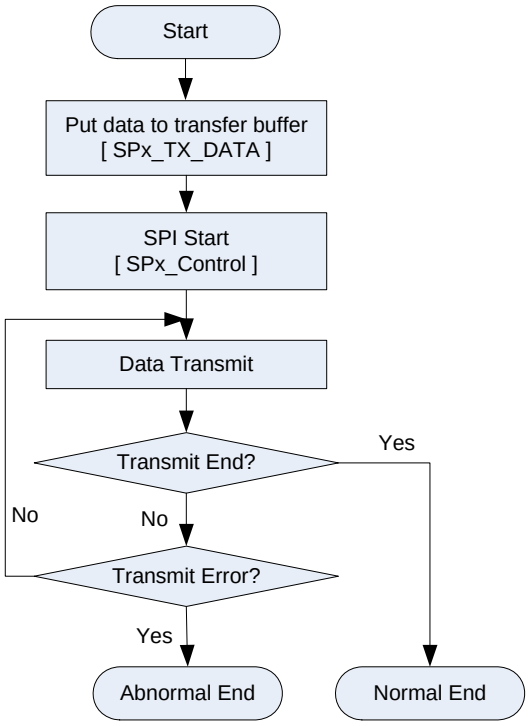

Figure A-4 SPI Send Data Flow

[Note]

None

A.4.13 SPI RW

[Function Name]

em1_spi_rw

[Format]

uint em1_spi_rw (uchar spi_n, uint addr);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number
addr	uint	I	The address which read

[Function Return]

DRV_ERR_PARAM

Received data

[Flow Chart]

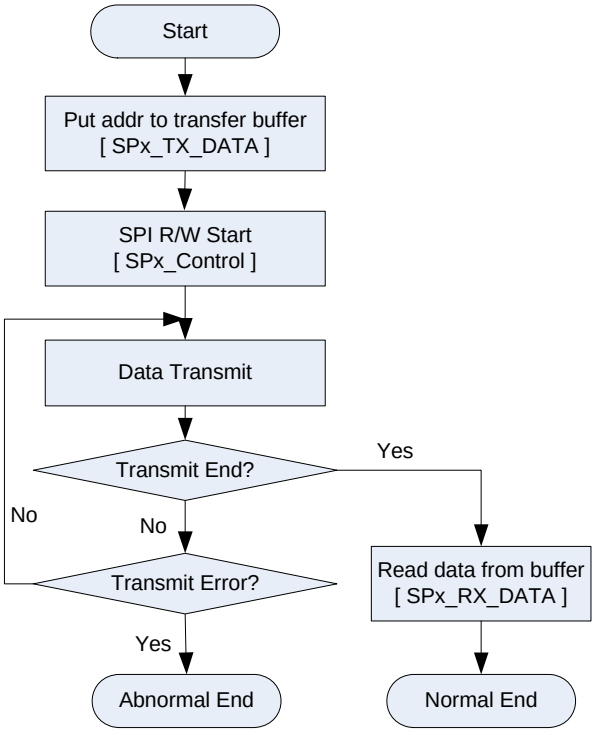

Figure A-5 SPI R/W Mode Flow

[Note]

None

A.4.14 SPI Receive

[Function Name]

em1_spi_receive

[Format]

uint em1_spi_receive (uchar spi_n);

[Argument]

Parameter	Type	I/O	Detail
spi_n	uchar	I	SPI number

[Function Return]

DRV_ERR_PARAM

Received data

[Flow Chart]

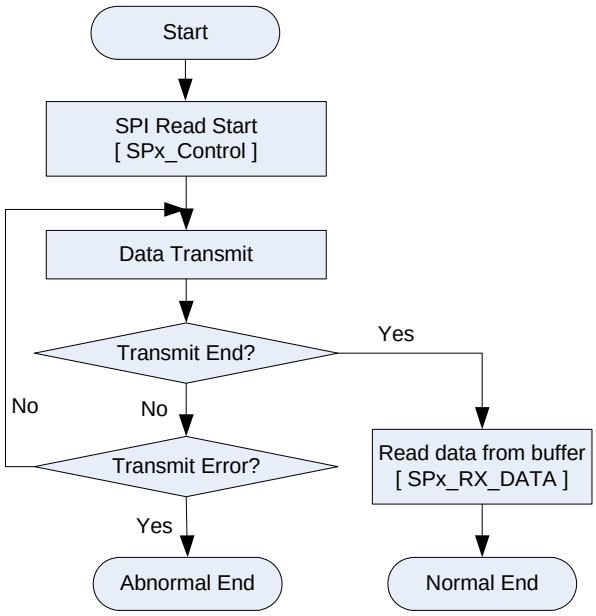

Figure A-6 SPI Receive Data Flow

[Note]

None

ANNEX Modification History

Number	Modification Contents	Author	Date
Ver 1.00	New version		Aug. 4 .2009