SH7785 Group

SH7785 PCIC Initial Settings Sample Program

R01AN0243EJ0101
Rev.1.01
Mar 17, 2011

Introduction

This application note presents a sample program for setting the PCIC initialization items required at SH7785 startup.

Target Device

SH7785

Contents

1. Introduction

1.1 Specifications

In section 2, this application note provides supplementary documentation to the PCIC hardware manual and in section 3 it presents an application example that, after PCIC initialization, displays the device ID and other information for the PCIC device on a serial console.

1.2 Functions Used

- Pin function controller (PFC)
- Serial communication interface (SCIF1)
- PCI controller (PCIC)

1.3 Applicable Conditions

Evaluation board:	Renesas R0P7785LC0011RL		
	External memory	(Area 0):	NOR flash memory: 64 MB Spansion S29GL256P90TFIRI
		(Areas 2 and 3):	DDR2 SDRAM: 128 MB (In 32-bit mode: 512 MB) Elpida EDE1108ACSE-6E-E (four devices)
	PCI device:	Channel 0:	1 Gbit Ethernet controller (Realtek RTL8169)
		Channel 1:	2-channel SATA controller (Silicon Image SI3512)
		Channel 2:	Expansion connector
		Channel 3:	Expansion connector
Microcontroller:	SH7785		
Operating frequencies:	Internal clock:	600 MHz	
	SuperHyway clock:	300 MHz	
	Peripheral clock:	50 MHz	
	DDR2 clock:	300 MHz	
	External bus clock:	100 MHz	
	PCI bus clock:	33 MHz	
Area 0 bus width:	32 bits (MD5 pin = high, MD6 pin = high)		
Clock operating mode:	Clock mode 16 (MD0 = low, MD1 = low, MD2 = low, MD3 = low, MD4 = high)		
Endian mode:	Little endian (MD8 = high)		
Addressing mode:	29-bit addressing (MD13 = low)		
Tool chain:	Super-H RISC engine Standard Toolchain Version 9.3.2.0		
Compiler options:	Other than the options specified in the include file in the High-Performance Embedded Workshop, the default options are used. -cpu=sh4a -endian=little -include="$(PROJDIR)\inc\drv","$(PROJDIR)\inc" -object="$(CONFIGDIR)\$(FILELEAF).obj" -debug -gbr=auto -chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 -nologo		
Assembler options:	-cpu=sh4a -endian=little -round=zero -denormalize=off -include="$(PROJDIR)\inc" -include="$(PROJDIR)\inc\drv" -debug -object="$(CONFIGDIR)\$(FILELEAF).obj" -literal=pool,branch,jump,return -nolist -nologo -chgincpath -errorpath		

1.4 Technical Terms Used in this Application Note

- Configuration register space
  The rules used and the programming model for the PCI compliant device are defined.
- I/O space
  The local bus space defined by the PCI local bus specifications.
- Memory space
  The memory space defined by the PCI local bus specifications.

1.5 Applicability of this Application Note

This application note presents code that operates without an operating system installed and displays the vendor ID and device ID for each PCI device on the serial console. It also displays the MAC address of any Ethernet devices. This application note also explains the basic usage of the PCIC module. The following functions, however, are not described in this application note.

- PCI target functions
- PCI power management
- Normal mode (external bus arbiter mode)
- Pseudo round robin or fixed priority arbitration
- Parity checking and error reporting
- Exclusive access
- The four external interrupts (INTA, INTB, INTC, and INTD)

1.6 Related Application Notes

This application note verifies operation using the code presented in SH7785 Group SH7785 Initial Settings Sample Program (R01AN0242EJ0101). Please refer to that application note while using this one.

2. PCIC Operation

2.1 PCIC Overview

The PCIC module provides an interface and protocol that allows PCI devices to be accessed.

This application note presents a sample program that, after performing the required initialization, displays the vendor ID and device ID for each PCI device and the MAC address of any Ethernet devices on the serial console.

Figure 1 shows the configuration of the PCI devices used in this application note.

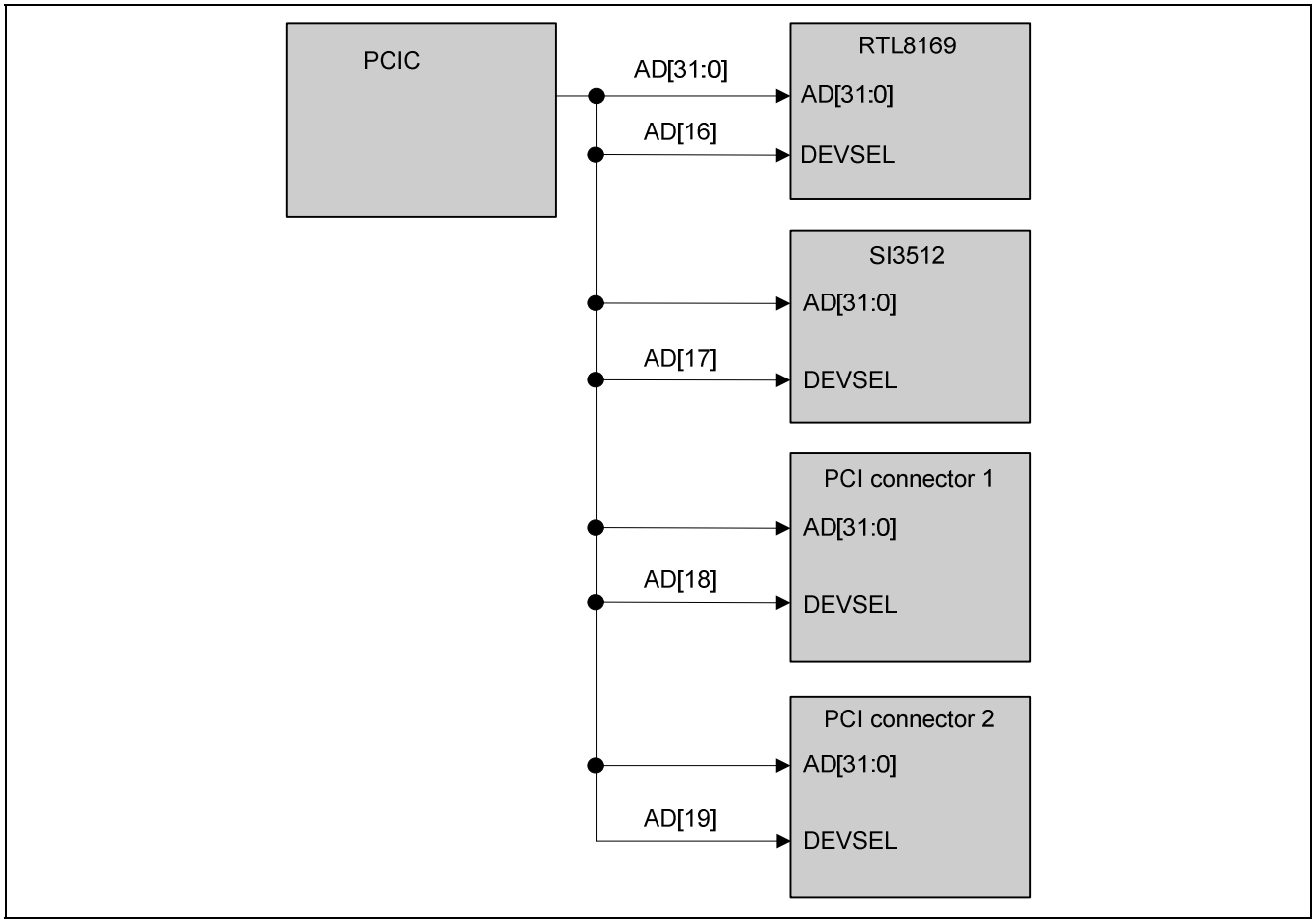
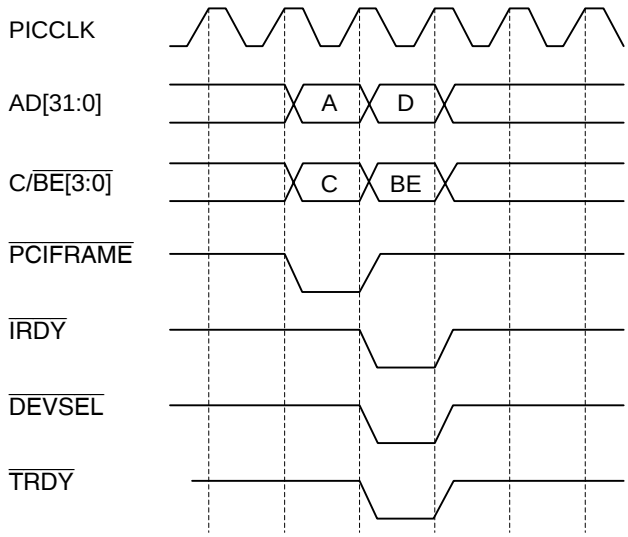
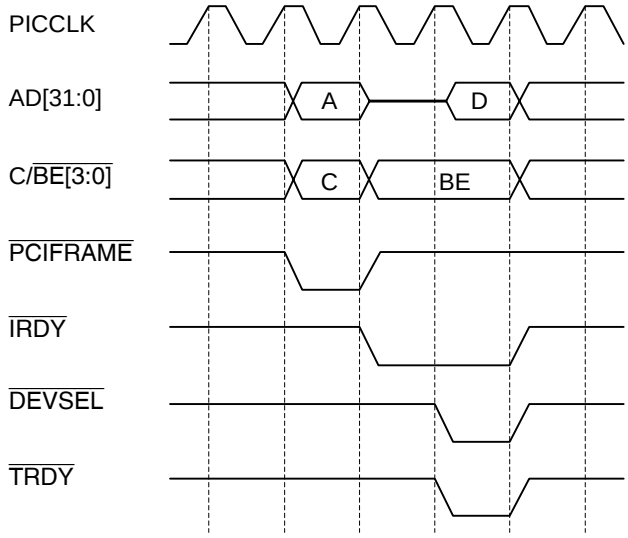


Figure 1 PCI Device Configuration

Figure 2 shows the basic timing chart for the PCI interface.



Write timing



Read timing

Figure 2 Basic PCI Timing

2.1.1 Supported PCI Commands

Table 1 lists the commands supported by the PCIC module.

Table 1 PCI Commands

C/BE[3:0] Signal	Command
0001	Special cycle
0010	I/O read
0011	I/O write
0110	Memory read
0111	Memory write
1010	Configuration read
1011	Configuration write

2.1.2 PCIC Memory Map

Table 2 lists the PCIC memory map.

Since this application note's program uses 29-bit address mode, space 0 is the only PCI memory space.

Table 2 PCIC Memory Map

Memory Space	Physical Addresses in 29-Bit Address Mode	Physical Address Size
PCI memory space 1 (area 4: when PCI is selected)	H'10000000 to H'13FFFFFF	64 MB
PCI memory space 0	H'FD000000 to H'FDFFFFFF	16 MB
Control register area	H'FE000000 to H'FE03FFFF	256 KB
PCI internal register	H'FE040000 to H'FE07FFFF	256 KB
Reserved	H'FE080000 to H'FE1FFFFF	1.5 MB
PCI I/O area	H'FE200000 to H'FE3FFFFF	2 MB

The PCIC uses four types of address space. These are memory space, control register space, PCI internal register space, and I/O space.

2.1.3 Accessing the PCI Memory Space

Figure 3 shows the memory map for the PCI bus from the SuperHyway bus.

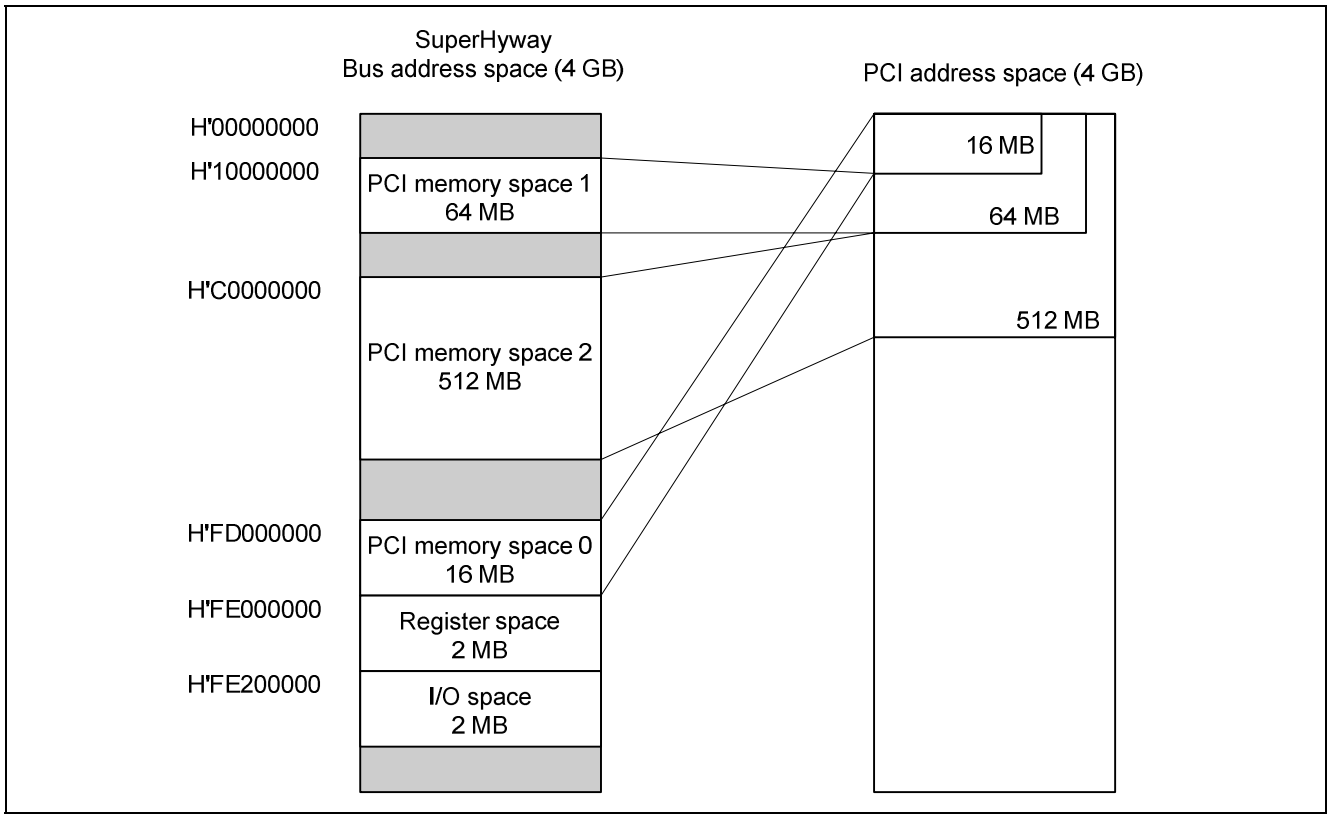


Figure 3 PCI Memory Space Access

(1) Access to PCI Memory Space

To access PCI memory space, the PCI memory bank register (PCIMBR) and PCI memory bank mask register (PCIMBMR) must be set.

For PCI memory space 0, the middle 6 bits ([23:18]) are controlled by PCI memory bank mask register (PCIMBMR0). Since this application note's program only uses PCI memory space 0, this application note does not discuss spaces other than PCI memory space 0.

- When PCIMBMR0[23:18] is B'1111 11: PCI bus address [23:18] = SuperHyway bus address [23:18]
- When PCIMBMR0[23:18] is B'0000 00: PCI bus address [23:18] = PCIMBMR0[23:18]

The upper 8 bits of the SuperHyway bus ([31:24]) are replaced with the contents of PCI memory bank register[31:24].

Figure 4 shows the access to the PCI memory space from the SuperHyway bus.

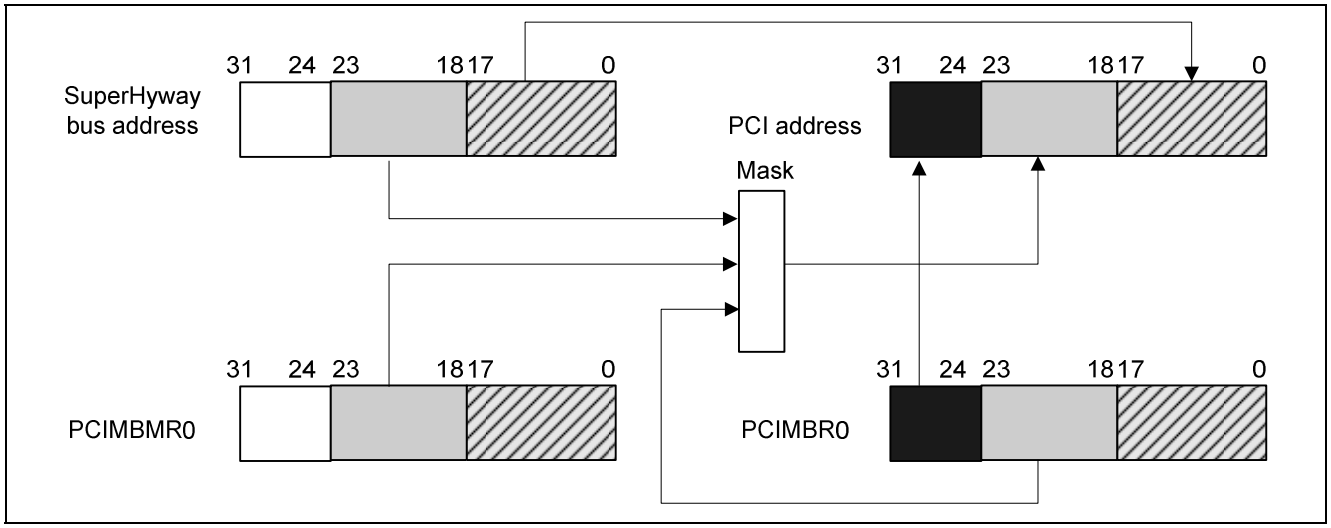


Figure 4 PCI Memory Space Access from the SuperHyway Bus

- Burst transfers
  In a PCI memory space access, if the CPU or DMA controller performs consecutive 32-byte burst accesses, a burst transfer of 32 bytes or more (that is, 64, 96, or a similar size) will be performed.

(2) Access to the PCI I/O Space

To access PCI I/O space, the PCIIO bank register (PCIIOBR) and PCIIO bank mask register (PCIIOBMR) must be set.

The lower 15 bits ([17:3]) of the SuperHyway bus address are used without modification on the PC bus.

- When PCIIOBMR[20:18] is B'111: PCI bus address [20:18] = SuperHyway bus address [20:18]
- When PCIIOBMR[20:18] is B'000: PCI bus address [20:18] = PCIIOBMR[20:18]

The upper 11 bits of the SuperHyway bus ([31:21]) are replaced with the contents of PCIIO bank register[31:21].

Figure 5 shows the access to the PCI I/O space from the SuperHyway bus.

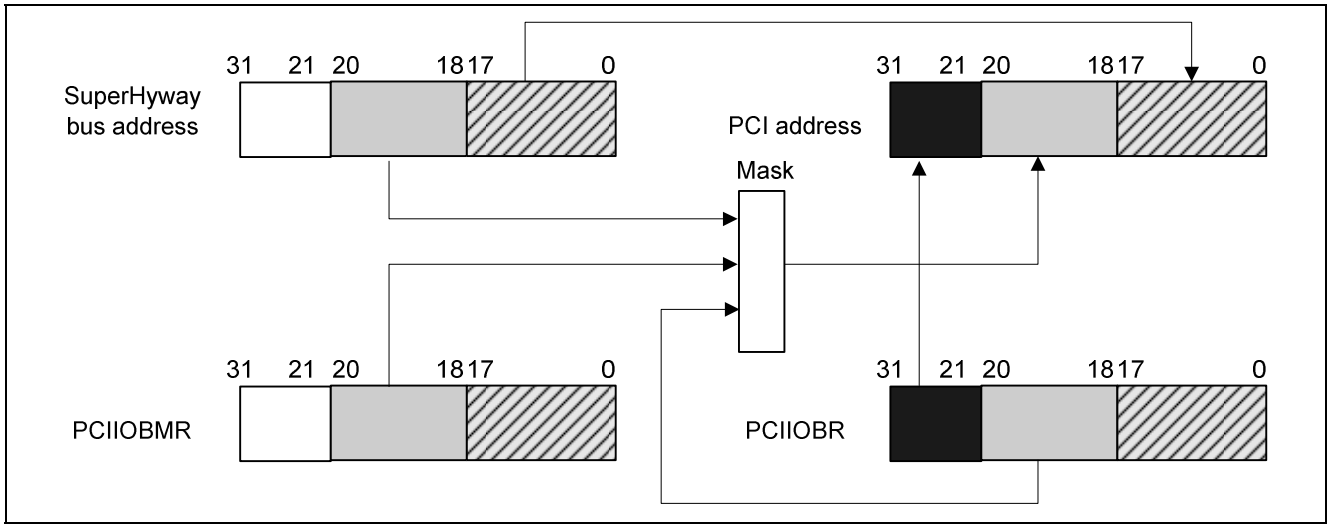


Figure 5 PCI I/O Space Access from the SuperHyway Bus

(3) PCIC Internal Register Access

All the PCIC internal registers (PCI control register, PCI configuration register, and the PCI local registers) can be accessed by the CPU. Transfers of 1, 2, and 4 bytes are supported as access sizes.

(4) PCIC Configuration Space Access

The configuration space can be accessed by setting the PCIPIO address register (PCIPAR) and the PCIPIO data register (PCIPDR) to execute a configuration cycle.

Setting the PCIPAR register and reading the PCIPDR also causes a configuration cycle to be executed.

When the PCIPAR device number is set to 0, AD16 becomes 1. It is 0 otherwise.

When the PCIPAR device number is set to 1, AD17 becomes 1. It is 0 otherwise.

2.2 PCIC Specifications

Table 3 lists the specifications of the PCIC module.

Table 3 PCIC Module Specifications

Item	Specification
PCI revision	A subset of PCI 2.2 is supported.
Bus operating frequency	33 MHz or 66 MHz
Bus width	32 bits
Host/target functions	Both the host and target functions are supported. This application note's program does not use the target function.
Interrupt	Four interrupts (INTA, INTB, INTC, and INTD) This application note's program does not use these interrupts.
Burst transfers	Transfers of up to 32 bytes are supported.

3. Application Example

This section presents an overview of the PCI device and sample settings as an example of how to operate the PCIC module.

3.1 Serial Console Specifications

Table 4 lists the specifications of the serial console.

Table 4 Serial Console Specifications

Item	Specification
Baud rate	115,200 bps
Data	8 bits
Parity bit	None
Stop bit	1 bit
Flow control	None

3.2 RTL8169 Overview

Table 5 presents an overview of the RTL8169 connected to the PCIC.

Table 5 RTL8169 Overview

Item	Description
Device	1 Gbit Ethernet controller device
Manufacturer	Realtek
Device number from the PCIC	0 (AD16 is connected)
Vendor ID	0x10EC
Device ID	0x8169
MAC address	Printed on the R0P7785LC0011RL board.

3.3 SI3512 Overview

Table 6 presents an overview of the SI3512 connected to the PCIC.

Table 6 SI3512 Overview

Item	Description
Device	SATA-2ch controller device
Manufacturer	SiliconImage
Device number from the PCIC	1 (AD17 is connected)
Vendor ID	0x1039
Device ID	0x3512

3.4 Sample Program Specifications

This section presents the specifications of the sample program and its flowchart.

3.4.1 Specifications

1. Performs the PCIC initial settings.
2. Displays the PCI device number 0 (RTL8169) Vendor ID and Device ID on the serial console.
3. Displays the PCI device number 1 (SI3512) Vendor ID and Device ID on the serial console.
4. Displays the RTL8169 MAC address on the serial console.
5. Performs writes to and reads from the SI3512 and displays the write and read data on the serial console.
6. Performs a 4 × 32 (128) byte burst transfer to the SI3512 memory space using DMA.
   (Since the SI3512 asserts the STOP signal and disconnects every data phase, this does not effect a burst transfer.)

3.4.2 Sample Program Main Flowchart

Figures 6 to 8 show the flowchart for the main section of the sample program.

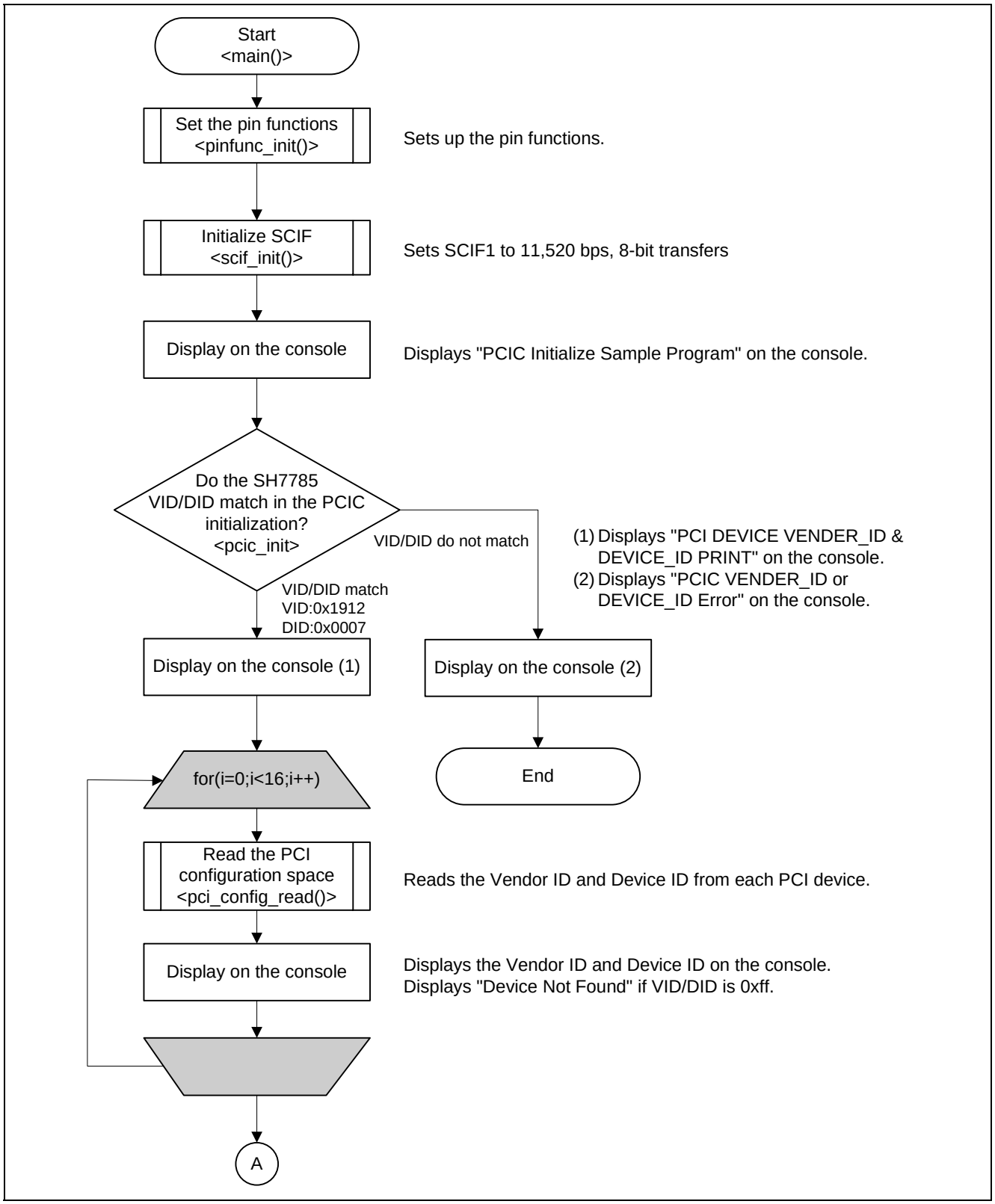


Figure 6 Sample Program Main Flowchart 1

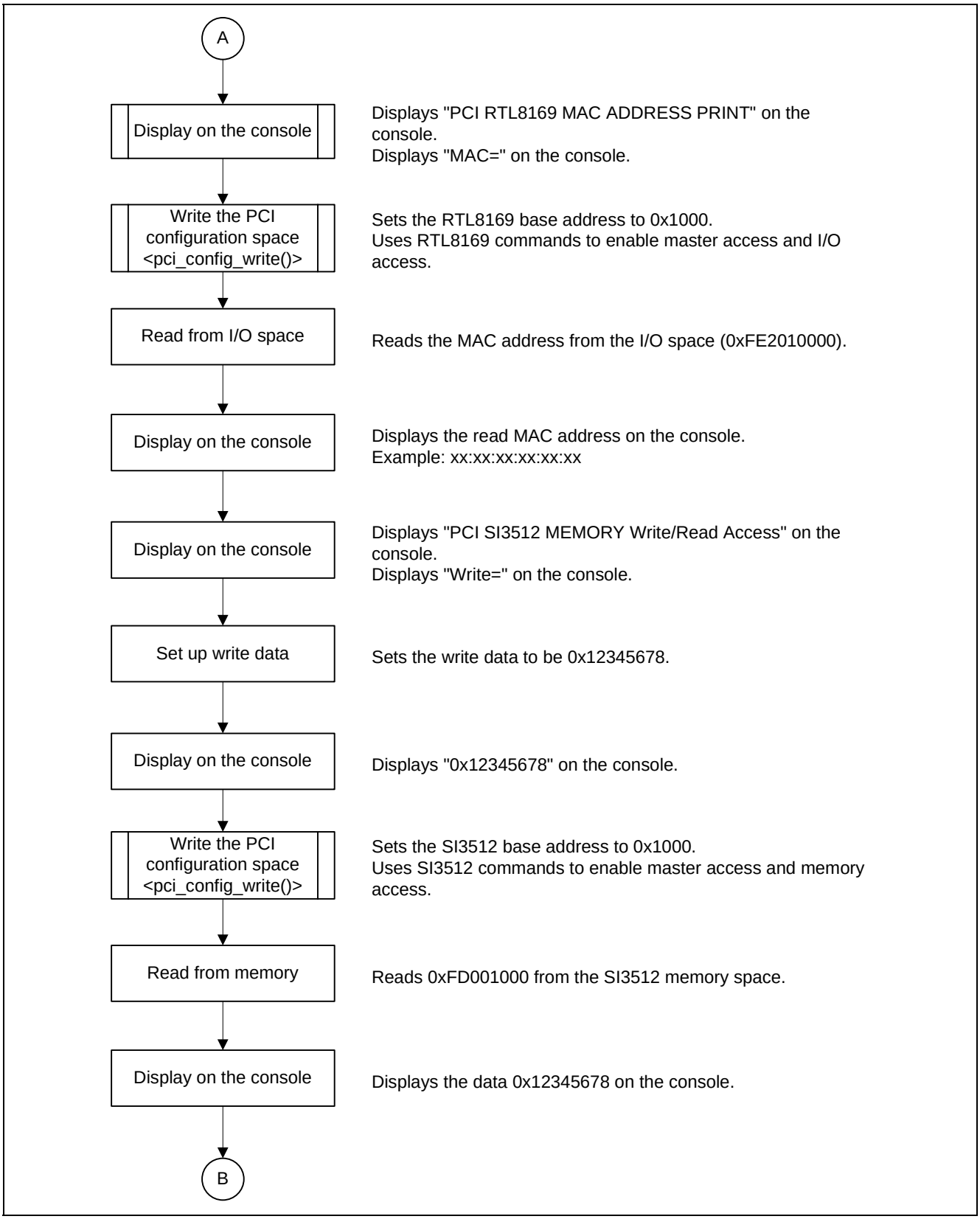


Figure 7 Sample Program Main Flowchart 2

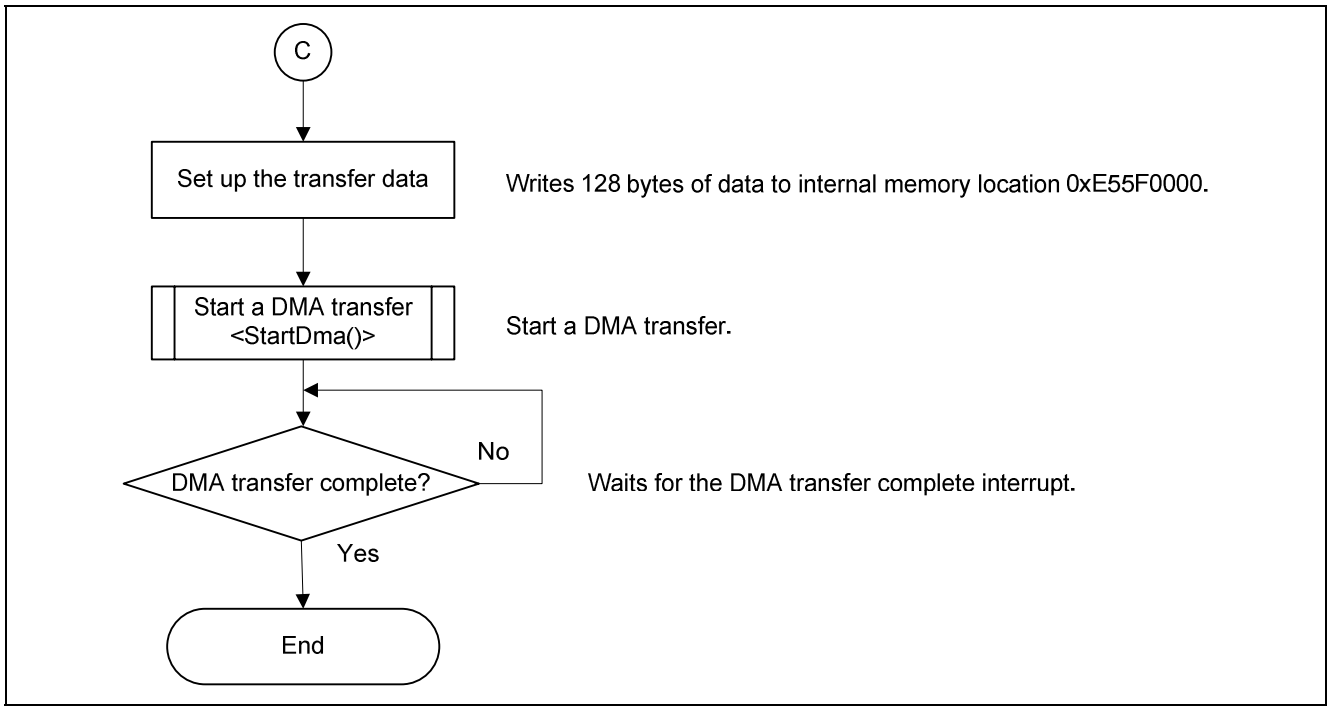


Figure 8 Sample Program Main Flowchart 3

3.4.3 Pin Function Initialization

Figures 9 and 10 show the flowcharts for pin function initialization.

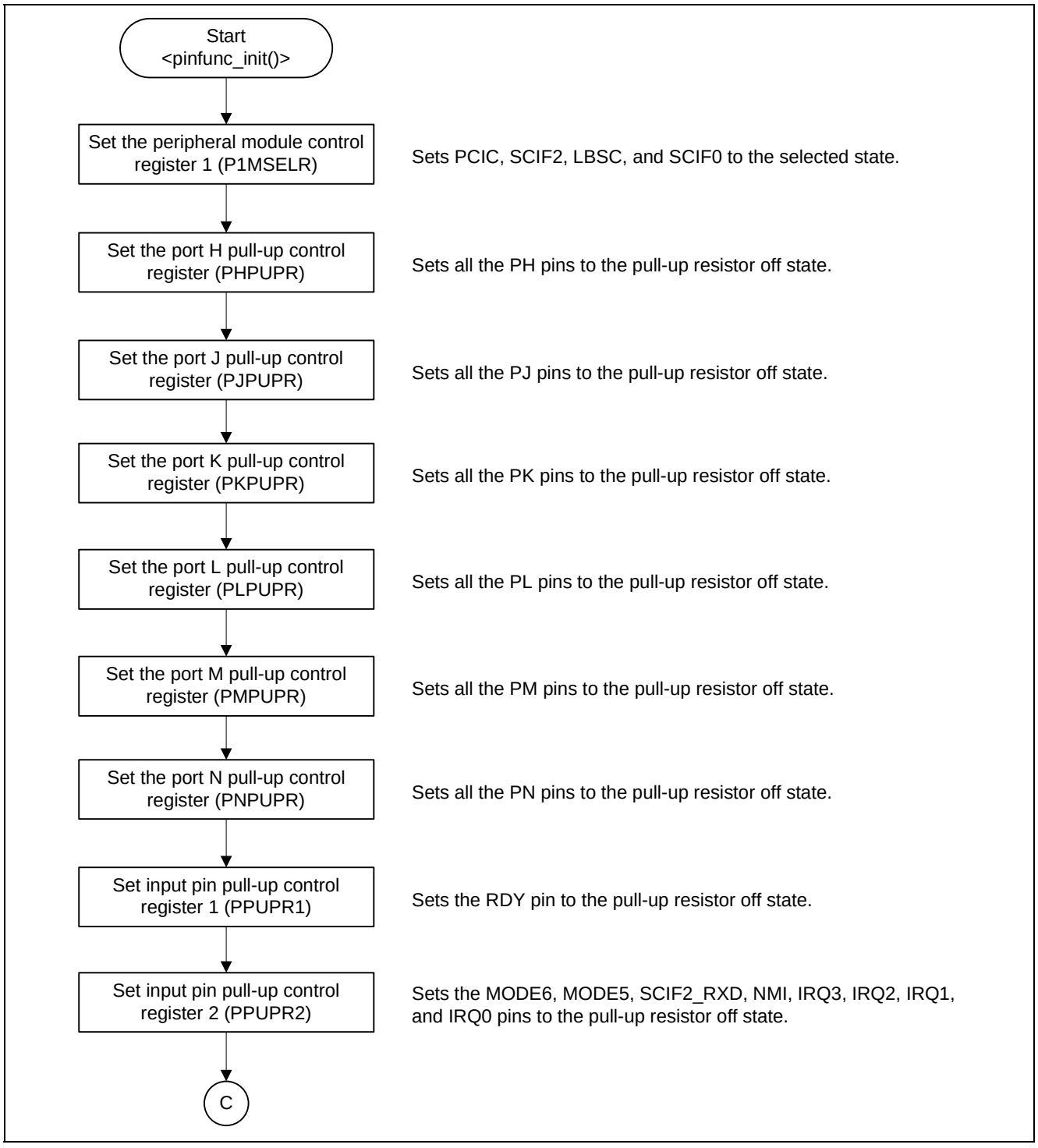


Figure 9 Pin Function Initialization Flowchart 1

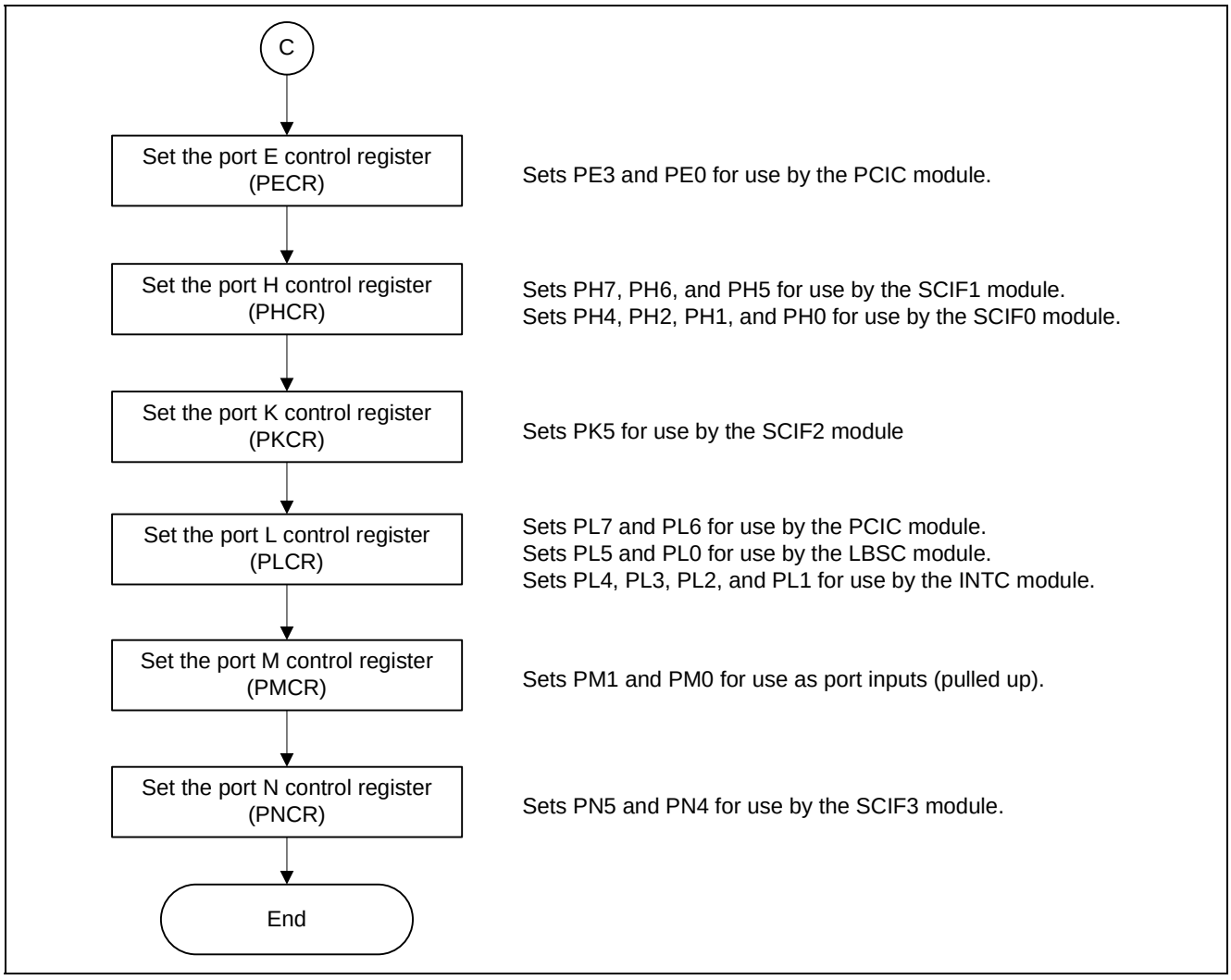


Figure 10 Pin Function Initialization Flowchart 2

3.4.4 SCIF Initialization

Figure 11 shows the flowchart for SCIF initialization.

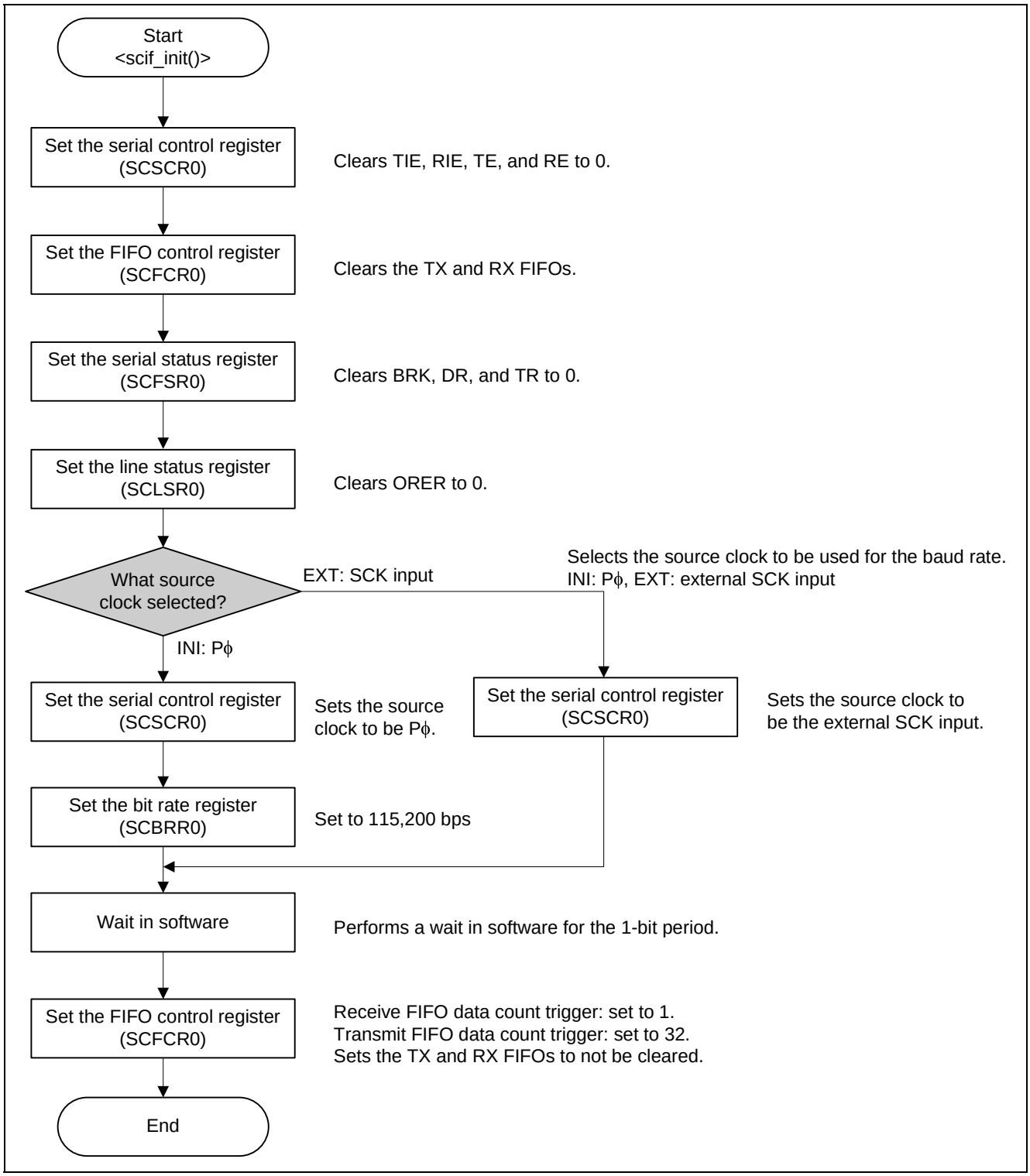


Figure 11 SCIF Initialization Flowchart

3.4.5 PCIC Initialization

Figure 12 shows the flowchart for PCIC initialization.

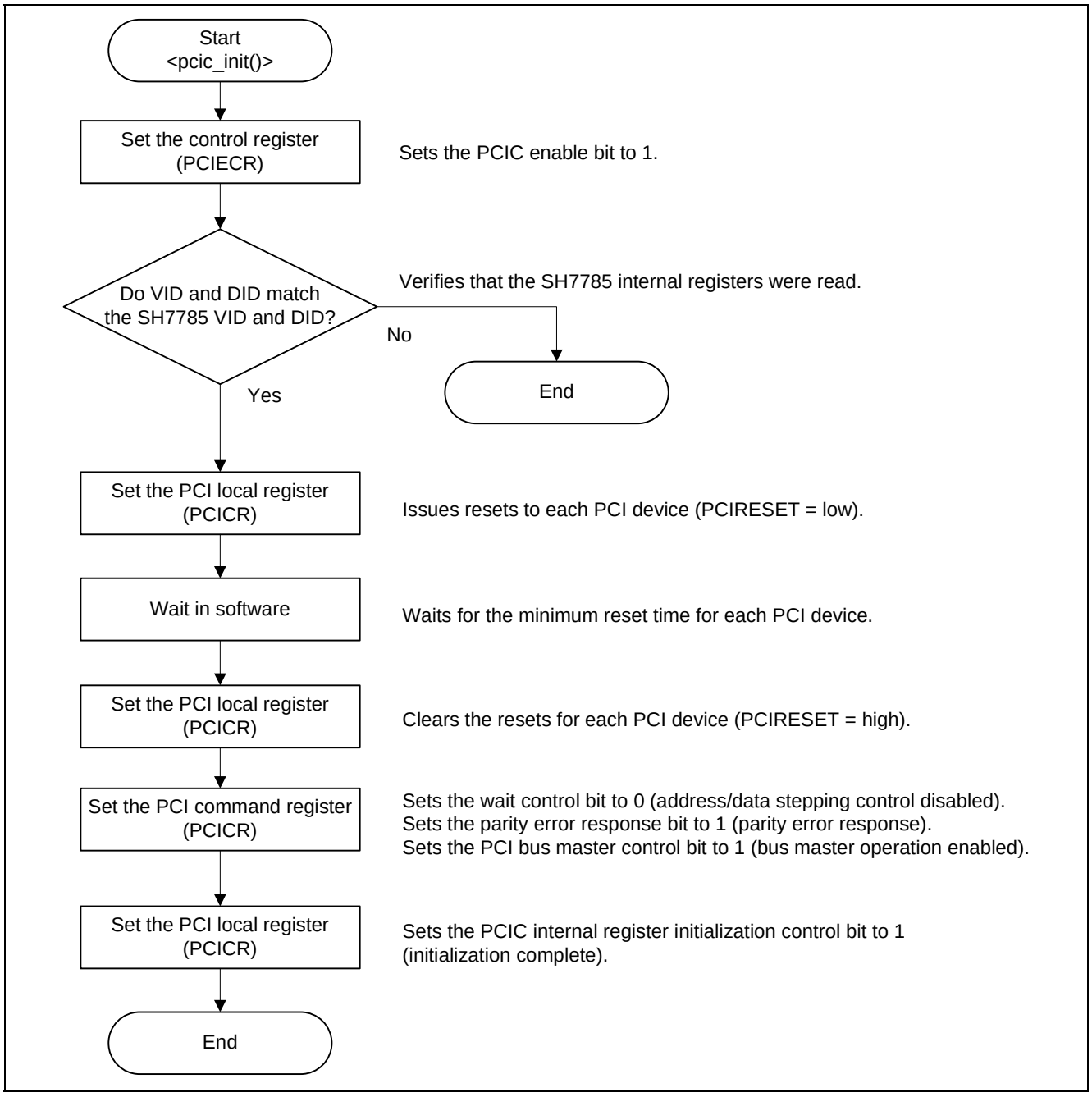


Figure 12 PCIC Initialization Flowchart

3.4.6 Configuration Space Read

Figure 13 shows the flowchart for reading from the configuration space.

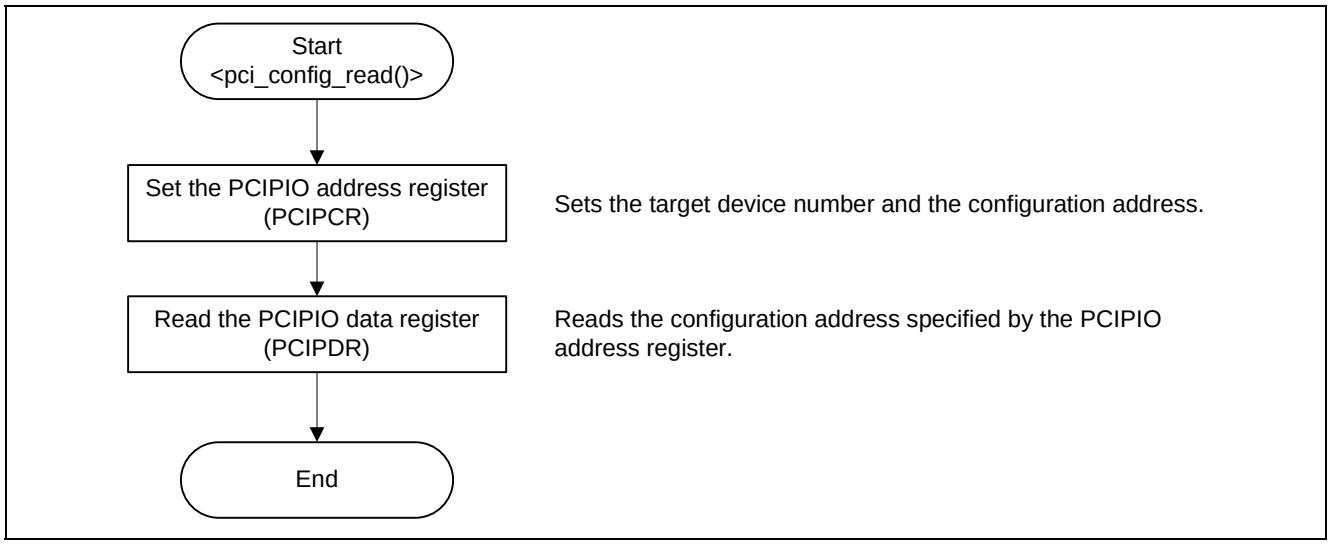


Figure 13 Configuration Space Read Flowchart

3.4.7 Configuration Space Write

Figure 14 shows the flowchart for writing to the configuration space.

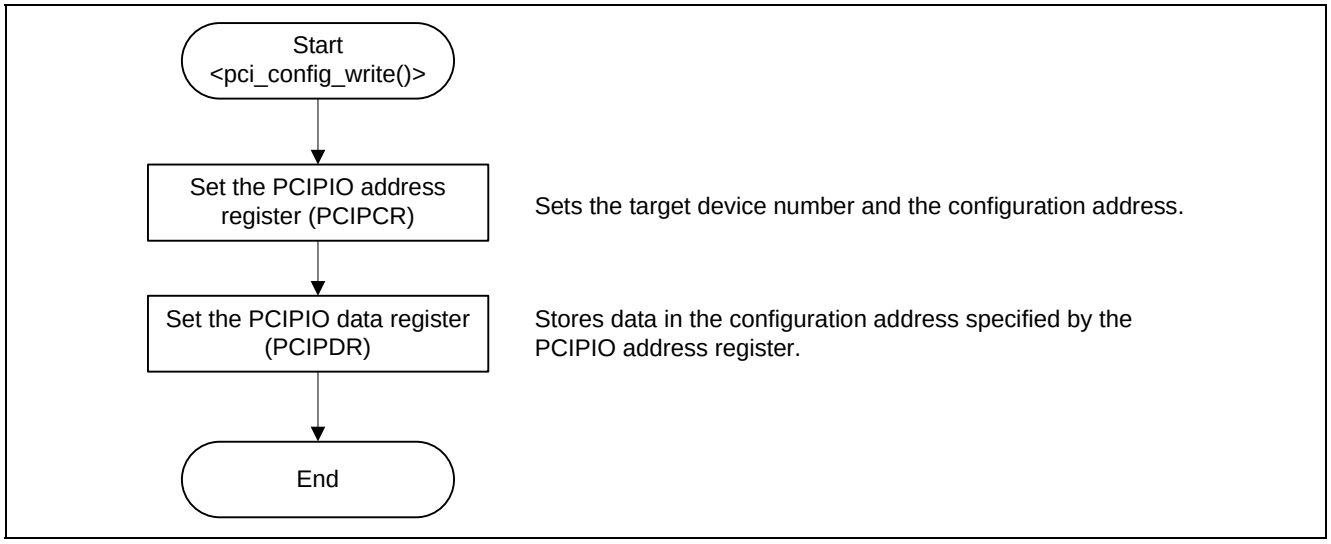


Figure 14 Configuration Space Write Flowchart

3.4.8 DMA Transfer Start

Figure 15 shows the flowchart for starting a DMA transfer.

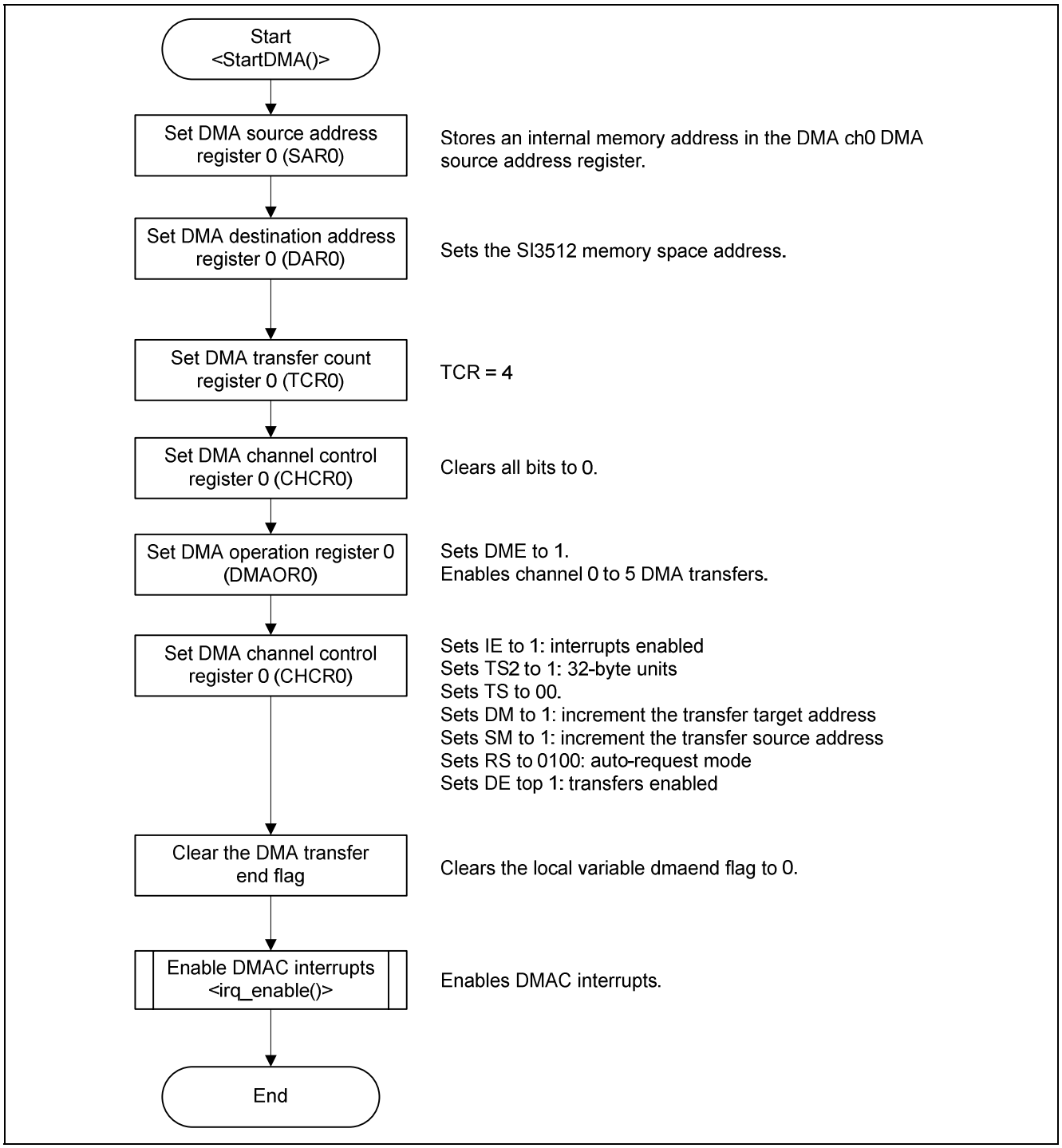


Figure 15 DMA Transfer Start Flowchart

3.4.9 DMA Interrupts

Figure 16 shows the flowchart for DMA transfer termination.

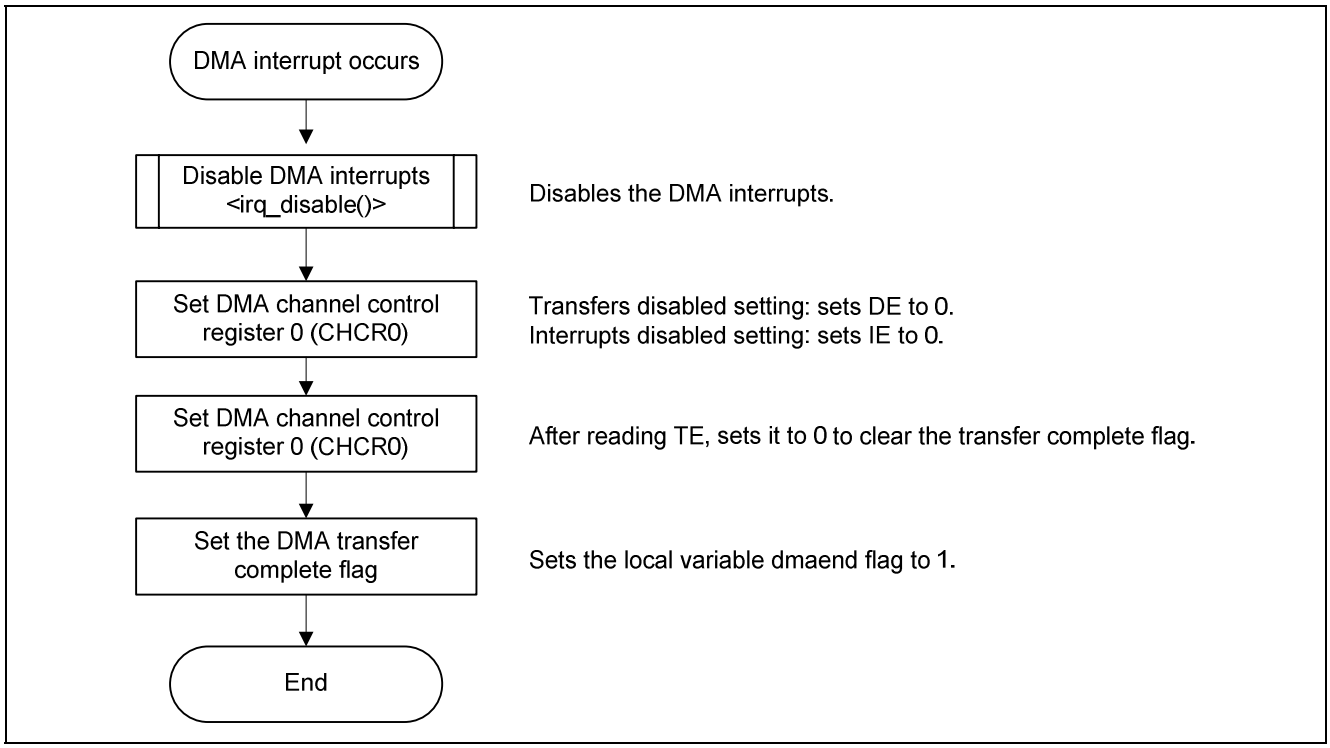


Figure 16 DMA Transfer Complete Flowchart

3.4.10 Section Allocations

Table 7 lists the section allocations used in this sample program.

Table 7 Section Allocations

Section	Section Usage	Area	Allocation Address (Virtual Address)	
P	Program area	ROM	0x00002000	P0 area (Can be cached, MMU address conversion possible)
C	Constant area	ROM		
C$BSEC	Uninitialized data area address structure	ROM		
C$DSEC	Initialized data area address structure	ROM		
D	Initialized data	ROM		
B	Uninitialized data area	RAM	0x0C000000	
R	Initialized data area	RAM		
S	Stack area	RAM	0x0DFF8000	
INTHandler	Exception/interrupt handler	ROM	0x80001000	P1 area 4 (Can be cached, MMU address conversion not possible)
VECTTBL	Reset vector table Interrupt vector table	ROM		
INTTBL	Interrupt mask table	ROM		
PIntPRG	Interrupt function	ROM		
RSTHandler	Reset handler	ROM	0xA0000000	P2 area (Can not be cached, MMU address conversion not possible)
PResetPRG	Reset program	ROM		
PnonCACHE	Program area (Cache invalid access)	ROM		

4. Sample Program

Sample program listing: PCI_Initialize.c

This is the main function in sample program.

```
001 /***********************************************************************
002 * DISCLAIMER
003
004 * This software is supplied by Renesas Electronics Corporation. and is only
005 * intended for use with Renesas products. No other uses are authorized.
006
007 * This software is owned by Renesas Electronics Corporation. and is
008 * protected under all applicable laws, including copyright laws.
009
010 * THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
011 * REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
012 * INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR
013 * A PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE
014 * EXPRESSLY DISCLAIMED.
015
016 * TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
017 * ELECTRONICS COR PORATION. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE
018 * LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL
019 * DAMAGES FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR
020 * ITS AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
021
022 * Renesas reserves the right, without notice, to make changes to this
023 * software and to discontinue the availability of this software.
024 * By using this software, you agree to the additional terms and
025 * conditions found by accessing the following link:
026 * http://www.renesas.com/disclaimer
027 ***********************************************************************/
028 /* Copyright (C) 2010. Renesas Electronics Corporation., All Rights Reserved. */
029 /*""FILE COMMENT""************ Technical reference data ****************
030 * System Name  : SH7785 Sample Program
031 * File Name    : PCI_Initialize.c
032 * Abstract     : SH7785 PCI Initial Settings Sample Program
033 * Version      : Ver 1.00
034 * Device       : SH7785
035 * Tool-Chain   : High-performance Embedded Workshop (Version 4.07.00.007)
036 *              : C/C++ Compiler Package for SuperH Family (V.9.3.2.0)
037 * OS           : None
038 * H/W Platform : This is a sample program with examples of initialization for the
039 * Description  : SH7785 PCI on the Renesas SH-4A board, model number R0P7785LC0011RL.
040 *              :
041 * Operation    :
042 * Limitation   :
043 *              :
044 ***********************************************************************
045 * History      : 30.SEP.2010 Ver. 1.00 First Release
046 *""FILE COMMENT END""*************************************************/
047 /*********************************************************************/
048 /*                                                                   */
049 /*  FILE        :PCI_Initialize.c                                    */
050 /*  DATE        :Fri, Oct 01, 2010                                   */
051 /*  DESCRIPTION :Main Program                                        */
052 /*  CPU TYPE    :Other                                               */
```

```
053 /*                                                                      */
054 /*  This file is generated by Renesas Project Generator (Ver.4.16).    */
055 /*                                                                      */
056 /************************************************************************/
057
058
059
060 #include "config.h"
061 #include "intc.h"
062 #ifdef __cplusplus
063 //#include <ios>                    // Remove the comment when you use ios
064 //_SINT ios_base::Init::init_cnt;   // Remove the comment when you use ios
065 #endif
066
067 void main(void);
068 #ifdef __cplusplus
069 extern "C" {
070 void abort(void);
071 }
072 #endif
073
074 /* ==== Function Declarations ==== */
075 void pinfunc_init( void );
076 static void scif_pr(int dev_no, char *title, char *data );
077 static unsigned long word_swap(unsigned long data);
078 void StartDMA( unsigned long sadd, unsigned long dadd, int cnt );
079 /* SCIF */
080 extern int scif_init(void);
081 extern voidscif_transmit_data( char*Data );
082 extern voidscif_transmit_data_byte( char *Data );
083 extern charscif_recive_data( char  *Data );
084
085 /* PCIC */
086 extern int init_pcic(void);
087 extern unsigned long pci_read_config(unsigned int dev_no, unsigned char index);
088 extern void pci_write_config(unsigned int dev_no, unsigned char index, unsigned long data);
089
090 /* ==== Variable Declarations ==== */
091 #define RTL8169_DEVNO  0
092 #define SI3512_DEVNO1
093 #define EXTCN1_DEVNO2
094 #define EXTCN2_DEVNO3
095
096 static char dev_name[16][8] = {
097   "RTL8169", "SI3512 ", "EXTCN1 ", "EXTCN2 ", "DEV4   ", "DEV5   ", "DEV6   ", "DEV7   ",
098   "DEV8   ", "DEV9   ", "DEV10  ", "DEV11  ", "DEV12  ", "DEV13  ", "DEV14  ", "DEV15  ",
099 };
100
101 static int dmaend;
102
103 /* ==== Macro Declarations ==== */
104
105
106 /*""FUNC COMMENT""*****************************************************
107 * ID                :
108 * Outline           : Sample program main() function
109 *                   : (PCIC initialization sample)
```

```
110 * Include           :
111 * Declaration       : void main(void)
112 * Description       : PCIC initialization main() function
113 *                   :
114 *                   :
115 *                   :
116 *                   :
117 *                   :
118 * Limitation        :
119 *                   :
120 * Argument          : none
121 * Return Value      : none
122 * Calling Functions :
123 *""FUNC COMMENT END""***************************************************/
124 void main(void)
125 {
126   int ret,i;
127   unsigned long data;
128   unsigned long datal[128];
129   char buff[20], tmp[6];
130
131   pinfunc_init();
132   scif_init();
133   scif_transmit_data("\rPCIC Initialze Sample Program\n");
134
135   if(init_pcic() != 0) {
136      scif_transmit_data("\rPCIC VENDER_ID or DEVICE_ID Error\n");
137      return;
138   }
139
140   scif_transmit_data("\n\rPCI DEVICE VENDER_ID & DEVICE_ID PRINT\n");
141   for(i=0;i<16;i++) {
142      data = pci_read_config(i, 0);
143      if(data==0xffffffff)
144         sprintf(buff, "%s", "Device Not Fount");
145      else
146         sprintf(buff, "%x", word_swap(data));
147      scif_pr(i, dev_name[i], buff);
148   }
149
150   scif_transmit_data("\n\rPCI RTL8169 MAC ADDRESS PRINT\n");
151   scif_transmit_data("\rMAC = ");
152   pci_write_config(RTL8169_DEVNO, 0x10, 0x00001000);
153   pci_write_config(RTL8169_DEVNO, 0x04, 0x00000005);
154
155   data = (*(unsigned long *)0xFE201000);
156   for(i=0;i<4;i++)
157      tmp[i] = (unsigned char)(data >> 8*i);
158
159   data = (*(unsigned long *)0xFE201004);
160   for(i=4;i<6;i++)
161      tmp[i] = (unsigned char)(data >> 8*(i-4));
162
163   for(i=0;i<5;i++) {
164      sprintf(buff, "%02x", (tmp[i] & 0x000000FF));
165      scif_transmit_data(buff);
166      scif_transmit_data(":");
```

```
167   }
168   sprintf(buff, "%02x", (tmp[5] & 0x000000FF));
169   scif_transmit_data(buff);
170   scif_transmit_data("\n");
171
172   scif_transmit_data("\n\rPCI Si3512 MEMORY Write/Read ACCESS\n");
173   scif_transmit_data("\rWrite Data = ");
174   data = 0x12345678;
175   sprintf(buff, "%02x", data);
176   scif_transmit_data(buff);
177   scif_transmit_data("\n");
178
179   pci_write_config(SI3512_DEVNO, 0x24, 0x00001000);
180   pci_write_config(SI3512_DEVNO, 0x04, 0x00000006);
181   (*(unsigned long *)0xFD00104C) = data;
182   data = 0;
183   data = (*(unsigned long *)0xFD00104C);
184
185   scif_transmit_data("\rRead Data = ");
186   sprintf(buff, "%02x", data);
187   scif_transmit_data(buff);
188   scif_transmit_data("\n");
189
190   for(i=0;i<128;i++)
191      (*(unsigned char *)(0xe55fe000 + i)) = i;
192   /* Burst Read 32Byte * 2 */
193   StartDMA( 0xe55fe000, 0xfd001000, 4 );
194   while(!dmaend);
195
196
197 }
198
199 /*""FUNC COMMENT""*****************************************************
200 * ID                :
201 * Outline           : Sample program main() function
202 *                   : (PCIC initialization sample)
203 * Include           :
204 * Declaration       : void pinfunc_init( void )
205 * Description       : Pin function settings
206 *                   :
207 *                   :
208 *                   :
209 *                   :
210 *                   :
211 * Limitation        :
212 *                   :
213 * Argument          : none
214 * Return Value      : none
215 * Calling Functions :
216 *""FUNC COMMENT END""*************************************************/
217
218 void pinfunc_init( void )
219 {
220   GPIO.P1MSELR.WORD = 0x3780;
221   GPIO.PHPUPR.BYTE = 0x00;
222   GPIO.PJPUPR.BYTE = 0x00;
223   GPIO.PKPUPR.BYTE = 0x00;
```

```
224   GPIO.PLPUPR.BYTE = 0x00;
225   GPIO.PMPUPR.BYTE = 0xFC;
226   GPIO.PNPUPR.BYTE = 0x00;
227   GPIO.PPUPR1.WORD = 0xFFFB;
228   GPIO.PPUPR2.WORD = 0xFF00;
229   GPIO.PECR.WORD = 0x0000;
230   GPIO.PHCR.WORD = 0x00C0;
231   GPIO.PKCR.WORD = 0x03FF;
232   GPIO.PLCR.WORD = 0x0000;
233   GPIO.PMCR.WORD = 0xFFFF;
234   GPIO.PNCR.WORD = 0xF0FF;
235 }
236
237 /*""FUNC COMMENT""*****************************************************
238 * ID                :
239 * Outline           : Sample program main() function
240 *                   : (PCIC initialization sample)
241 * Include           :
242 * Declaration       : static void scif_pr(int dev_no, char *title, char *data )
243 * Description       : Serial console device and data display
244 *                   :
245 *                   :
246 *                   :
247 *                   :
248 *                   :
249 * Limitation        :
250 *                   :
251 * Argument          : dev_no: Device number, *title: Device name
252 *                   : *data: Display data
253 * Return Value      : none
254 * Calling Functions :
255 *""FUNC COMMENT END""*************************************************/
256 static void scif_pr(int dev_no, char *title, char *data )
257 {
258   scif_transmit_data("\r");
259   scif_transmit_data(title);
260   scif_transmit_data(" = ");
261   scif_transmit_data(data);
262   scif_transmit_data("\n");
263 }
264
265 /*""FUNC COMMENT""*****************************************************
266 * ID                :
267 * Outline           : Sample program main() function
268 *                   : (PCIC initialization sample)
269 * Include           :
270 * Declaration       : static unsigned long word_swap(unsigned long data)
271 * Description       : Long data word swap operation
272 *                   :
273 *                   :
274 *                   :
275 *                   :
276 *                   :
277 * Limitation        :
278 *                   :
279 * Argument          : data: data
280 * Return Value      : Transformed data
```

```
281 * Calling Functions :
282 *""FUNC COMMENT END""************************************************/
283 static unsigned long word_swap(unsigned long data)
284 {
285   unsigned long tmp;
286   tmp = ((data & 0xFFFF0000) >> 16) | ((data & 0x0000FFFF) << 16);
287   return tmp;
288 }
289
290 /*""FUNC COMMENT""*****************************************************
291 * ID                :
292 * Outline           : Sample program main() function
293 *                   : (PCIC initialization sample)
294 * Include           :
295 * Declaration       : void StartDMA( unsigned long sadd, unsigned long
dadd, int cnt )
296 * Description       : DMAC ch0 initialization and DMA transfer start
297 *                   : Enables DMA interrupts.
298 *                   :
299 *                   :
300 *                   :
301 *                   :
302 * Limitation        :
303 *                   :
304 * Argument          : sadd: Start address, fdadd: Destination address, cnt: Transfer count
305  Return Value       : none
306 * Calling Functions :
307 *""FUNC COMMENT END""************************************************/
308 void StartDMA( unsigned long sadd, unsigned long dadd, int cnt )
309 {
310   *(unsigned long *)0xFC808020 = sadd;
311   *(unsigned long *)0xFC808024 = dadd;
312
313   DMAC0.TCR.BIT.CNT = cnt;
314   DMAC0.CHCR.LONG   = 0;
315   DMAC.DMAOR0.BIT.DME  = 1;  /* Enables DMAC0 to 5 */
316   DMAC0.CHCR.BIT.IE    = 1;  /* Enables interrupts */
317   DMAC0.CHCR.BIT.TS2   = 1;  /* 32-byte access */
318   DMAC0.CHCR.BIT.TS    = 0;  /* 32-byte access */
319   DMAC0.CHCR.BIT.DM    = 1;  /* Increment transfer destination */
320   DMAC0.CHCR.BIT.SM    = 1;  /* Increment transfer source */
321   DMAC0.CHCR.BIT.RS    = 4;  /* Auto-request */
322   DMAC0.CHCR.BIT.DE    = 1;  /* Enable transfers */
323   dmaend = 0;
324   irq_enable( _DMAC0 );
325 }
326
327 void dmac0_irq( void )
328 {
329   int tmp;
330   DMAC0.CHCR.BIT.IE    = 0;  /* Disables interrupts */
331   DMAC0.CHCR.BIT.DE    = 0;  /* Disables transfers */
332   tmp = DMAC0.CHCR.BIT.TE;
333   DMAC0.CHCR.BIT.TE    = 0;  /* Clears the flag */
334   dmaend = 1;
335 }
336
```

```
337 #ifdef __cplusplus
338 void abort(void)
339 {
340
341 }
342 #endif
```

Sample program listing: pcic.c

Sample PCIC initialization

```
001 /***********************************************************************
002 * DISCLAIMER
003
004 * This software is supplied by Renesas Electronics  Corp. and is only
005 * intended for use with Renesas products. No other uses are authorized.
006
007 * This software is owned by Renesas Electronics  Corp. and is protected
under
008 * all applicable laws, including copyright laws.
009
010 * THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
011 * REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
012 * INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR
013 * A PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE
014 * EXPRESSLY DISCLAIMED.
015
016 * TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
017 * TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
018 * FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
019 * FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
020 * AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
021
022 * Renesas reserves the right, without notice, to make changes to this
023 * software and to discontinue the availability of this software.
024 * By using this software, you agree to the additional terms and
025 * conditions found by accessing the following link:
026 * http://www.renesas.com/disclaimer
027 ***********************************************************************/
028 /* Copyright (C) 2010. Renesas Electronics Corp., All Rights Reserved.  */
029 /*""FILE COMMENT""************ Technical reference data ****************
030 * System Name  : SH7785 Sample Program
031 * File Name    : pcic.c
032 * Abstract     : SH7785 PCIC Initial Settings Sample Program
033 * Version      : Ver 1.00
034 * Device       : SH7785
035 * Tool-Chain   : High-performance Embedded Workshop (Version 4.07.00.007)
036 *              : C/C++ Compiler Package for SuperH Family (V.9.3.2.0)
037 * OS           : None
038 * H/W Platform : This is a sample program with examples of initialization for the
039 * Description  : SH7785 PCIC on the Renesas SH-4A board, model number R0P7785LC0011RL.
040 *              :
041 * Operation    :
042 * Limitation   :
043 *              :
044 ***********************************************************************
045 * History      : 30.SEP.2010 Ver. 1.00 First Release
046 *""FILE COMMENT END""*************************************************/
047
048 #include "config.h"
049 #include "pcic.h"
050
051 extern void delay( int cnt );
052
053 /*""FUNC COMMENT""*****************************************************
```

```
054 * ID                :
055 * Outline           : Sample program main() function
056 *                   : (PCIC initialization sample)
057 * Include           :
058 * Declaration       : int init_pcic(void)
059 * Description       : PCIC initial settings
060 *                   : VENDER_ID and DEVICE_ID discrimination
061 *                   : VENDER_ID and DEVICE_ID discrimination
062 *                   : PCI bus master settings
063 *                   :
064 *                   :
065 * Limitation        :
066 *                   :
067 * Argument          : none
068 * Return Value      : -1: Either the VENDER_ID or the DEVICE_ID does not match.
069 * Calling Functions :
070 *""FUNC COMMENT END""**************************************************/
071 int init_pcic(void)
072 {
073   PCIC.PCIECR.BIT.ENBL = 1;
074   if( (PCIC.PCIVID != VENDER_ID)
075      && PCIC.PCIDID != DEVICE_ID)
076      return -1;
077
078   /* Toggle PCI reset pin */
079   PCIC.PCICR.LONG = PCICR_PREFIX | PCICR_PRST;
080   delay(10000);
081   PCIC.PCICR.LONG = PCICR_PREFIX;
082   PCIC.PCICMD.BIT.WCC = 0;/* Address/data stepping control disabled */
083   PCIC.PCICMD.BIT.PER = 1;/* Parity error response */
084   PCIC.PCICMD.BIT.BM  = 1;/* PCI bus master */
085
086   PCIC.PCICR.LONG = PCICR_PREFIX | PCICR_CFIN;
087
088   return 0;
089 }
090
091 /*""FUNC COMMENT""*****************************************************
092 * ID                :
093 * Outline           : Sample program main() function
094 *                   : (PCIC initialization sample)
095 * Include           :
096 * Declaration       : unsigned long pci_read_config(unsigned int dev_no, unsigned char index)
097 * Description       : Reads the PCI configuration data for the target device.
098 *                   :
099 *                   :
100 *                   :
101 *                   :
102 *                   :
103 * Limitation        :
104 *                   :
105 * Argument          : dev_no: Device number, index: Configuration address
106 * Return Value      : Read data
107 * Calling Functions :
108 *""FUNC COMMENT END""**************************************************/
109 unsigned long pci_read_config(unsigned int dev_no, unsigned char index)
110 {
```

```
111 unsigned long par_data = 0x80000000 | (dev_no << 11);
112
113   PCIC.PCIPAR.LONG = par_data | (index & 0xfc);
114     return PCIC.PCIPDR;
115 }
116
117 /*""FUNC COMMENT""*****************************************************
118 * ID                :
119 * Outline           : Sample program main() function
120 *                   : (PCIC initialization sample)
121 * Include           :
122 * Declaration       : void pci_write_config(unsigned int dev_no, unsigned char index, unsigned long data)
123 * Description       : Writes the PCI configuration data for the target device.
124 *                   :
125 *                   :
126 *                   :
127 *                   :
128 *                   :
129 * Limitation        :
130 *                   :
131 * Argument          : dev_no: Device number, index: Configuration address, data: Write data
132 * Return Value      : none
133 * Calling Functions :
134 *""FUNC COMMENT END""*************************************************/
135 void pci_write_config(unsigned int dev_no, unsigned char index, unsigned long data)
136 {
137     unsigned long par_data = 0x80000000 | (dev_no << 11);
138     PCIC.PCIPAR.LONG = par_data | (index & 0xfc);
139     PCIC.PCIPDR = data;
140 }
```

Sample program listing: pcic.h

This is the header used by pcic.c.

```
01 #ifndef _PCIC_H
02 #define _PCIC_H
03
04
05 #define VENDER_ID 0x1912
06 #define DEVICE_ID 0x0007
07
08 /* SH7785 PCI Local Registers */
09 #define PCICR_PREFIX    0xA5000000  /* CR prefix for write */
10 #define PCICR_PRST      0x00000002  /* PCI Reset Assert */
11 #define PCICR_CFIN      0x00000001  /* Central Fun. Init Done */
12 #define PCICR_PFCS      0x00000800  /* TRDY/IRDY Enable */
13 #define PCICR_FTO       0x00000400  /* TRDY/IRDY Enable */
14 #define PCICR_PFE       0x00000200  /* Target Read Single */
15 #define PCICR_TBS       0x00000100  /* Target Byte Swap */
16 #define PCICR_ARBM      0x00000040  /* PCI Arbitration Mode */
17 #define PCICR_IOCS      0x00000004  /* INTA output assert */
18
19 #endif /* _PCIC_H */
```

Sample program listing: scif.c

Sample initialization for the serial communications interface channel 0.

```
001 /***********************************************************************
002 * DISCLAIMER
003
004 * This software is supplied by Renesas Electronics  Corp. and is only
005 * intended for use with Renesas products. No other uses are authorized.
006
007 * This software is owned by Renesas Electronics  Corp. and is protected
008 * under all applicable laws, including copyright laws.
009
010 * THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
011 * REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
012 * INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR
013 * A PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE
014 * EXPRESSLY DISCLAIMED.
015
016 * TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
017 * TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
018 * FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
019 * FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
020 * AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
021
022 * Renesas reserves the right, without notice, to make changes to this
023 * software and to discontinue the availability of this software.
024 * By using this software, you agree to the additional terms and
025 * conditions found by accessing the following link:
026 * http://www.renesas.com/disclaimer
027 ***********************************************************************/
028 /* Copyright (C) 2010. Renesas Electronics Corp., All Rights Reserved.  */
029 /*""FILE COMMENT""************ Technical reference data ****************
030 * System Name  : SH7785 Sample Program
031 * File Name    : scif.c
032 * Abstract     : SH7785 SCIF Initial Settings Sample Program
033 * Version      : Ver 1.00
034 * Device       : SH7785
035 * Tool-Chain   : High-performance Embedded Workshop (Version 4.07.00.007)
036 *              : C/C++ Compiler Package for SuperH Family (V.9.3.2.0)
037 * OS           : None
038 * H/W Platform : This is a sample program with examples of initialization for the
039 * Description  : SH7785SCIF on the Renesas SH-4A board, model number R0P7785LC0011RL.
040 *              :
041 * Operation    :
042 * Limitation   :
043 *              :
044 ******************************************************************
045 * History      : 30.SEP.2010 Ver. 1.00 First Release
046 *""FILE COMMENT END""*************************************************/
047
048
049 #include"scif.h"
050
051 /*""FUNC COMMENT""*****************************************************
052 * ID               :
053 * Outline          : Sample program main() function
054 *                  : (PCIC initialization sample)
```

```
055 * Include           :
056 * Declaration       : int delay( int cnt )
057 * Description       : Software wait
058 *                   : Iterates the for statement cnt times.
059 *                   :
060 *                   :
061 *                   :
062 *                   :
063 * Limitation        :
064 *                   :
065 * Argument          : cnt
066 * Return Value      : none
067 * Calling Functions :
068 *""FUNC COMMENT END""**************************************************/
069 void delay( int cnt )
070 {
071   int i;
072   for(i=0;i<cnt;i++);
073 }
074
075 /*""FUNC COMMENT""*****************************************************
076 * ID                :
077 * Outline           : Sample program main() function
078 *                   : (PCIC initialization sample)
079 * Include           :
080 * Declaration       : int scif_init(void)
081 * Description       : SCIF initial settings
082 *                   :
083 *                   :
084 *                   :
085 *                   :
086 *                   :
087 * Limitation        :
088 *                   :
089 * Argument          : none
090 * Return Value      : -1: Baud rate clock calculation error
091 * Calling Functions :
092 *""FUNC COMMENT END""*************************************************/
093 int scif_init(void)
094 {
095   unsigned short data;
096   int t = -1, cnt = 0;
097
098   SCIF.SCSCR.WORD = 0x0000;  /* TIE, RIE, TE, RE Clear */
099
100   SCIF.SCFCR.BIT.TFCL = 1;/* Tx FIFO Clear */
101   SCIF.SCFCR.BIT.RFCL = 1;/* Rx FIFO Clear */
102
103   SCIF.SCFSR.WORD = 0x0000;  /* BRK, DR, TR Clear */
104
105 #if defined(CONFIG_SCIF_CLK_EXTERNAL)
106   SCIF.SCSCR.BIT.CKE = 2; /* Clock source: SCK */
107 #elif  defined(CONFIG_SCIF_CLK_PCLK)
108   SCIF.SCSCR.BIT.CKE = 0; /* Clock source : PCLK */
109   t = SCBRR_VALUE(CONFIG_BPS, CONFIG_SCIF_CLK_PCLK);
110 #endif /* CONFIG_SCIF_CLK */
111
```

```
112   if(t > 0) {
113      while(t >= 256) {
114         cnt++;
115         t >> 2;
116      }
117      if(cnt > 3)
118         return -1;
119
120      SCIF.SCSMR.BIT.CKS = cnt;
121      SCIF.SCBRR = t;
122   }
123   delay(1000);
124
125   SCIF.SCFCR.BIT.RTRG = 0;
126   SCIF.SCFCR.BIT.TTRG = 0;
127   SCIF.SCFCR.BIT.TFCL = 1;/* Tx FIFO Clear */
128   SCIF.SCFCR.BIT.RFCL = 1;/* Rx FIFO Clear */
129
130   SCIF.SCFCR.BIT.TFCL = 0;/* Tx FIFO Not Clear */
131   SCIF.SCFCR.BIT.RFCL = 0;/* Rx FIFO Not Clear */
132   SCIF.SCSCR.BIT.TE  = 1;
133   SCIF.SCSCR.BIT.RE  = 1;
134   return 0;
135 }
136
137 /*""FUNC COMMENT""*****************************************************
138 * ID                :
139 * Outline           : Sample program main() function
140 *                   : (PCIC initialization sample)
141 * Include           :
142 * Declaration       : void scif_transmit_data( char *Data )
143 * Description       : Multibyte SCIF data transfer
144 *                   :
145 *                   :
146 *                   :
147 *                   :
148 *                   :
149 * Limitation        :
150 *                   :
151 * Argument          : *Data: Transfer data storage area
152 * Return Value      : none
153 * Calling Functions :
154 *""FUNC COMMENT END""*************************************************/
155 void scif_transmit_data( char*Data )
156 {
157   while( *Data )
158   {
159      while(!(SCIF.SCFSR.BIT.TDFE));/* Wait until the transfer data write enabled state is achieved. */
160      SCIF.SCFTDR = *Data;              /* Transfer data setting */
161      Data++;
162      while(!(SCIF.SCFSR.BIT.TEND));/* Wait for transfer complete. */
163      SCIF.SCFSR.BIT.TDFE = 0;
164      SCIF.SCFSR.BIT.TEND = 0;
165   }
166 }
167
168 /*""FUNC COMMENT""*****************************************************
```

```
169 * ID                :
170 * Outline           : Sample program main() function
171 *                   : (PCIC initialization sample)
172 * Include           :
173 * Declaration       : void scif_transmit_byte_data( char *Data )
174 * Description       : Single byte SCIF data transfer
175 *                   :
176 *                   :
177 *                   :
178 *                   :
179 *                   :
180 * Limitation        :
181 *                   :
182 * Argument          : *Data: Transfer data storage area
183 * Return Value      : none
184 * Calling Functions :
185 *""FUNC COMMENT END""**************************************************/
186 void scif_transmit_data_byte( char *Data )
187 {
188       while(!(SCIF.SCFSR.BIT.TDFE));/* Wait until the transfer data write enabled state is achieved. */
189       SCIF.SCFTDR = *Data;              /* Transfer data setting */
190       while(!(SCIF.SCFSR.BIT.TEND));/* Wait for transfer complete. */
191       SCIF.SCFSR.BIT.TDFE = 0;
192       SCIF.SCFSR.BIT.TEND = 0;
193 }
194
195 /*""FUNC COMMENT""*****************************************************
196 * ID                :
197 * Outline           : Sample program main() function
198 *                   : (PCIC initialization sample)
199 * Include           :
200 * Declaration       : char scif_recive_data( char *Data )
201 * Description       : SCIF data reception
202 *                   :
203 *                   :
204 *                   :
205 *                   :
206 *                   :
207 * Limitation        :
208 *                   :
209 * Argument          : *Data: Receive data storage area
210 * Return Value      : -1: Reception error
211 * Calling Functions :
212 *""FUNC COMMENT END""**************************************************/
213 char scif_recive_data( char  *Data )
214 {
215   unsigned char  ReadData, i = 0;
216   char  ret_cd = 0;
217
218   for(;;)
219   {
220      if(( SCIF.SCFSR.BIT.ER  ) ||
221         ( SCIF.SCFSR.BIT.BRK ) ||
222         ( SCIF.SCFSR.BIT.DR  ))    /* Did an error occur? */
223      {
224         ReadData = SCIF.SCFRDR; /* Dummy data read */
225         ret_cd = -1;/* Reception error setting */
```

```
226         SCIF.SCFSR.WORD &= 0x0000; /* Error clear */
227         SCIF.SCLSR.WORD &= 0x0000;
228      }
229      else if( SCIF.SCFSR.BIT.RDF ) /* Data reception? */
230      {
231         *Data = SCIF.SCFRDR; /* Data acquisition */
232         SCIF.SCFSR.BIT.RDF = 0; /* Receive sign clear */
233         SCIF.SCFSR.BIT.DR  = 0; /* Receive sign clear */
234         scif_transmit_data_byte( Data );
235         if( *Data == '\n' )  /* Was the acquired data CR? */
236         {
237            break;/* Terminate processing */
238         }
239         if( *Data == 0x0d )  /* Was the acquired data CR? */
240         {
241            break;/* Terminate processing */
242         }
243         Data++;  /* Set to the next data acquisition address. */
244         if( ++i == 4 )
245         {
246            ret_cd = -1;
247         }
248      }
249      if( ret_cd == -1 )
250      {
251         break;
252      }
253   }
254   return( ret_cd );
255 }
256
257
```

Sample program listing: scif.h

This is the header file used by scif.c.

```
01
02 #ifndef _SCIF_H
03 #define _SCIF_H
04
05 #include "config.h"
06
07 #if defined(CONFIG_SCIF0)
08 #define SCIF   (*(volatile struct st_scif    *)0xFFEA0000) /* SCIF0   Address */
09 #elif defined(CONFIG_SCIF1)
10 #define SCIF   (*(volatile struct st_scif    *)0xFFEB0000) /* SCIF1   Address */
11 #elif defined(CONFIG_SCIF2)
12 #define SCIF   (*(volatile struct st_scif    *)0xFFEC0000) /* SCIF2   Address */
13 #elif defined(CONFIG_SCIF3)
14 #define SCIF   (*(volatile struct st_scif    *)0xFFED0000) /* SCIF3   Address */
15 #elif defined(CONFIG_SCIF4)
16 #define SCIF   (*(volatile struct st_scif    *)0xFFEE0000) /* SCIF4   Address */
17 #elif defined(CONFIG_SCIF5)
18 #define SCIF   (*(volatile struct st_scif    *)0xFFEF0000) /* SCIF5   Address */
19 #endif /* CONFIG_SCIFn */
20
21 //#define SCBRR_VALUE(bps, clk) ((clk+16*bps)/(16*bps)-1)
22 #define SCBRR_VALUE(bps, clk) ((clk)/(32*bps)-1)
23
24 /* SCFCR */
25 #define  RTRG1 0
26 #define  RTRG161
27 #define  RTRG322
28 #define  RTRG483
29 #define  TTRG320
30 #define  TTRG161
31 #define  TTRG2 2
32 #define  TTRG0 3
33
34
35
36 #endif /* _SCIF_H */
```

5. Results of Program Execution

The following occur when this program is run.

- The Vendor ID and Device ID for each PCI device are displayed on the console.
- The RTL8169 MAC address is displayed on the console.

6. Reference Documents

- Software Manual
  SH4-A Software Manual (REJ09B0003)
  (The latest version can be downloaded from the Renesas Electronics Web site.)
- Hardware Manual
  SH7785 Group Hardware Manual (REJ09B0261)
  (The latest version can be downloaded from the Renesas Electronics Web site.)

Website and Support

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

Inquiries
http://www.renesas.com/inquiry

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.01	Mar.17.11	—	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.
7F, 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 1.0