EPSON
EXCEED YOUR VISION

Self-Testing Sample Program and Reference Information for Compliance with the IEC 60730 Standard

(For the S1C17803 / Applicable to the S1C17 Family)

Please follow the directions in “NOTICE_Application Notes Sample Programs.pdf” in the compressed download package.

SEIKO EPSON CORPORATION

Rev.1.0

NOTICE

No part of this material may be reproduced or duplicated in any form or by any means without the written permission of Seiko Epson. Seiko Epson reserves the right to make changes to this material without notice. Seiko Epson does not assume any liability of any kind arising out of any inaccuracies contained in this material or due to its application or use in any product or circuit and, further, there is no representation that this material is applicable to products requiring high level reliability, such as, medical products. Moreover, no license to any intellectual property rights is granted by implication or otherwise, and there is no representation or warranty that anything made in accordance with this material will be free from any patent or copyright infringement of a third party. This material or portions thereof may contain technology or the subject relating to strategic products under the control of the Foreign Exchange and Foreign Trade Law of Japan and may require an export license from the Ministry of Economy, Trade and Industry or other approval from another government agency.

All brands or product names mentioned herein are trademarks and/or registered trademarks of their respective companies.

©SEIKO EPSON CORPORATION 2009, All rights reserved.

Table of Contents

1. Description 1

2. Sample 2

2.1 Structure 2

2.2. Operating Procedure 2

2.3. Sample Program 3

2.3.1 c17SelfTest.h definition list 3
2.3.2 RAM R/W test (for 16-bit devices) 3
2.3.3 RAM R/W test (for 8-bit devices) 4
2.3.4 Register R/W test 4
2.3.5 Status register R/W test 6
2.3.6 RAM March-C test 7
2.3.7 Checksum 9
2.3.8 CRC (calculation) 9
2.3.9 CRC (table lookup) 10
2.3.10 Interrupt test 10
2.3.11 Main clock stability test 12

3. Reference Information 14

3.1 Diagnosis of communication errors with external interfaces 14

Revision History 15

1. Description

The International Electrotechnical Commission (IEC) has issued the IEC 60730 standard for development of household appliances. Consumer electronics sold and used in Europe are required by law to comply with this safety standard. The purpose of this standard is to protect consumers from hazards arising from malfunctions and defects in final products by discovering them in a timely manner through periodic self-testing.

Software control for microcontrollers is categorized according to the following standards.

Class A: Control functions not intended to be relied upon for the safety of the equipment
(e.g., lighting fixtures)

Class B: Control functions intended to prevent unsafe operation of the controlled equipment
(e.g., washing machines, refrigerators, freezers, dishwashers)

Class C: Control functions intended to prevent special hazards (e.g., combustion appliances)

The majority of control software for household appliances falls under Class B, and the following self-testing is recommended for final products.

- Diagnosis of microcontroller and program counter stack failure
- Diagnosis of interrupt cycle abnormalities
- Diagnosis of abnormalities in the operating clock frequency of the microcontroller
- Diagnosis of abnormalities in the ROM/RAM memory
- Diagnosis of communication errors with external interfaces

For more detailed information, refer to Annex H of IEC 60730.

This document covers the sample software and reference information for self-testing the Seiko Epson S1C17 Family of products. The self-testing sample software is designed to run on the S1C17803, but tests can also be performed on other models in the S1C17 Family using part of the sample program.

Self-testing includes:

- Memory failure test (read/write test, March-C test)
- Integrity testing of data in the memory (generates checksum and CRC)
- Interrupt test (interrupt cycle and interrupt count check)
- Main clock stability test (operating frequency check)

The read/write test and March-C test perform a read and write test in ranges specified for the memory, register, stack pointer, and status register.

For generating a checksum and CRC, an error detection code is requested and returned for data in the memory in the specified range.

The interrupt test counts how many interrupts occur in a certain period of time and returns that as a value.

The main clock stability test uses the subclock (32KHz) to check that the main clock is operating at a normal operating frequency.

2. Sample

2.1 Structure

The files in the selftest folder are as follows.

Table 2.1 List of sample files

File name	Content
boot.c	Startup module
vector.c	Vector table definition
main.c	Calls the test function
c17TestRam16.c	RAM R/W test (for 16-bit devices)
c17TestRam8.c	RAM R/W test (for 8-bit devices)
c17TestRegister.s	General purpose register, stack pointer R/W test
c17TestPsr.c	Status register R/W test
c17RwPsr.c	Software exception for causing the status register R/W test to be performed
c17TestRamMarchc.c	RAM March-C test
c17TestChksum.c	Calculates checksum
c17TestCrc.c	Calculates CRC (calculation)
c17TestCrcTbl.c	Calculates CRC (table lookup)
c17803TestIntterupt.c	Interrupt test
c17803TestClk.c	Main clock stability test
vector.h	Vector header file
c17SelfTest.h	Header file used by the self-testing sample program
s1c17803.h	Standard I/O definition file
selftest_gnu17IDE.mak	Makefile
selftest_gnu17IDE.lds	Linker script file
selftest_gnu17IDE.par	Parameter setting file
selftest_gnu17IDE.cmd	GDB command file
GDB17 Launch for selftest.launch	GDB launcher file
Other files	Other project related files

2.2. Operating Procedure

Importing a project

(1) Run GNU17 IDE and import the “selftest” project.

* Refer to “3. Software Development” in the S5U1C17001C Manual.

Build

(1) Using GNU17 IDE, build the “selftest” project.

(2) Connect the S5U1C17803T (SVT17803 CPU Board) and SVT17 ICD BOARD or S5U1C33001H to the PC with a USB cable.

(3) Reset the SVT17 ICD BOARD or S5U1C33001H and SVT33L17 CPU Board.

Run

(1) In GNU17 IDE, click [Run] -> [External Tools] -> [External Tools Configurations].

(2) In the program, select [GDB17 Launch for selftest] and click [Run].

2.3. Sample Program

This section explains the sample program provided.

2.3.1 c17SelfTest.h definition list

Table 2.3. c17SelfTest.h definition list

Definition name	Value	Content
E_OK	0	Indicates a good test result.
E_MEMORY	1	Indicates a problem with the memory test.
E_REGISTER	2	Indicates a problem with the register test.
E_CLOCK	3	Indicates a problem with the main clock test.

2.3.2 RAM R/W test (for 16-bit devices)

Performs a read-write test on the specified RAM area.

Function name	c17TestRam16
Parameter 1	volatile unsigned short *chkAddr
Parameter 2	unsigned short chkNum
Return value	short (E_OK or E_MEMORY)
Function overview	Writes 0xaa55 (0x55aa) in the memory in the quantity specified by chkNum from the address specified by chkAddr. Then the values are read and compared with the value written. E_OK is returned if the values are the same, and E_MEMORY if they are different.
Caution	The last bit of chkAddr is always handled as 0. Operation is not guaranteed if the specified memory area overlaps the stack area. c17TestRam16 checks memory equivalent to chkNum × 2 bytes.

Procedure

Test procedure	1. The logical product of chkAddr and 0xfffffe is found. 2. Steps (1) through (7) are repeated until the number of tests in chkNum from chkAddr has been performed. (1) The memory value indicated by chkAddr is backed up. (2) 0xaa55 is written in the memory indicated by chkAddr. (3) The memory value written in (2) is checked to see if it is 0xaa55. ◆ If it is not 0xaa55 (i) The memory value indicated by chkAddr is returned to the backed up value. (ii) E_MEMORY is returned. End. (4) 0x55aa is written in the memory indicated by chkAddr. (5) The memory value written in (4) is checked to see if it is 0x55aa. ◆ If it is not 0x55aa (i) The memory value indicated by chkAddr is returned to the backed up value. (ii) E_MEMORY is returned. End. (6) The memory value indicated by chkAddr is returned to the backed up value. (7) chkAddr is incremented to the next memory address. 3. E_OK is returned. End.

Program source

c17TestRam16.c

2.3.3 RAM R/W test (for 8-bit devices)

Function name	c17TestRam8
Parameter 1	volatile unsigned char *chkAddr
Parameter 2	unsigned short chkNum
Return value	short (E_OK or E_MEMORY)
Function overview	0xa5 (0x5a) is written in the memory in the quantity specified by chkNum from the address specified by chkAddr. Then the values are read and compared with the value written. E_OK is returned if the values are the same, and E_MEMORY if they are different.
Caution	Operation is not guaranteed if the specified memory area overlaps the stack area. c17TestRam8 checks memory equivalent to chkNum × 1 byte.

Procedure

Test procedure	1. Steps (1) through (7) are repeated until the number of tests in chkNum from chkAddr has been performed. (1) The memory value indicated by chkAddr is backed up. (2) 0xa5 is written in the memory indicated by chkAddr. (3) The memory value written in (2) is checked to see if it is 0xa5. ◆ If it is not 0xa5 (i) The memory value indicated by chkAddr is returned to the backed up value. (ii) E_MEMORY is returned. End. (4) 0x5a is written in the memory indicated by chkAddr. (5) The memory value written in (4) is checked to see if it is 0x5a. ◆ If it is not 0x5a (i) The memory value indicated by chkAddr is returned to the backed up value. (ii) E_MEMORY is returned. End. (6) The memory value indicated by chkAddr is returned to the backed up value. (7) chkAddr is incremented to the next memory address. 2. E_OK is returned. End.

Program source

c17TestRam8.c

2.3.4 Register R/W test

Performs a read-write test on the general purpose register and stack pointer in the CPU.

Function name	c17TestRegister
Parameter	void
Return value	short (E_OK or E_REGISTER)
Function overview	0x555555 and 0xaaaaaa are written in the general purpose register in the CPU. Then the values are read and compared with the value written. Similarly, 0x555554 and 0xaaaaa8 are written in the stack pointer and tested. E_OK is returned if the values are the same, and E_REGISTER if they are different.

Procedure

Test procedure	1. The values in register r4 through r7 are backed up. 2. 0x555555 is written in registers r0 through r7. 3. The values of the respective registers are compared. (1) r0 and 0x555555 are compared. (2) r0 and r1 are compared. (3) r0 and r2 are compared. (4) r0 and r3 are compared. (5) r0 and r4 are compared. (6) r0 and r5 are compared. (7) r0 and r6 are compared. (8) r0 and r7 are compared. ◆ E_REGISTER is returned if (1) through (8) above are different. End. 4. 0xaaaaaa is written in registers r0 through r7. 5. The values of the respective registers are compared. (1) r0 and 0xaaaaaa are compared. (2) r0 and r1 are compared. (3) r0 and r2 are compared. (4) r0 and r3 are compared. (5) r0 and r4 are compared. (6) r0 and r5 are compared. (7) r0 and r6 are compared. (8) r0 and r7 are compared. ◆ E_REGISTER is returned if (1) through (8) above are different. End. 6. The stack pointer is backed up. 7. 0x555554 is written as the stack pointer. 8. The stack pointer value and 0x555554 are compared. ◆ If the value differs (1)The stack pointer is returned to the backed up value. (2)E_REGISTER is returned. End. 9. 0xaaaaa8 is written as the stack pointer. 10. The stack pointer value and 0xaaaaa8 are compared. ◆ If the value differs (1)The stack pointer is returned to the backed up value. (2)E_REGISTER is returned. End. 11. The stack pointer value is returned to the backed up value. 12. r4 through r7 register values are returned to the backed up values. 13. E_OK is returned. End.

Program source

c17TestRegister.s

2.3.5 Status register R/W test

Performs a read-write test on the status register in the CPU.

Function name	c17TestPsr
Parameter	void
Return value	short (E_OK or E_REGISTER)
Function overview	0x55 and 0xaa are written in the status register in the CPU. Then the values are read and compared with the value written. E_OK is returned if the values are the same, and E_REGISTER if they are different.

Function name	c17WritePsr
Parameter	void
Return value	void
Function overview	Software exception for backing up and writing to the status register. The status register values saved in the stack are backed up to backupPsrVal. Then the wPsrVal value is written in the status register.

Function name	c17ReadPsr
Parameter	void
Return value	void
Function overview	Software exception for comparing the values of the status register. The status register value saved in the stack and wPsrVal are compared. E_OK is saved in result if the values are the same, and E_REGISTER if they are different. Then the backupPsrVal value is written in the status register.

Global variables

wPsrVal	Stores the values to write in the status register.
backupPsrVal	Stores the status register values.
result	Test results (E_OK or E_REGISTER)

Procedure

Test procedure	1. The value of wPsrVal is set as 0x55. 2. Software exception c17WritePsr is induced. The value in the status register immediately prior to the software exception induced by c17WritePsr is backed up in backupPsrVal, and the value wPsrVal (0x55) is written in the status register. 3. Software exception c17ReadPsr is induced. The value in the status register immediately prior to the software exception induced by c17ReadPsr is compared with wPsrVal (0x55), and the result of the comparison is saved in result. 4. The result value is checked. ◆ If the value is E_REGISTER E_REGISTER is returned. End. 5. The value of wPsrVal is set as 0xaa. 6. Software exception c17WritePsr is induced. The value in the status register immediately prior to the software exception induced by c17WritePsr is backed up in backupPsrVal, and the value wPsrVal (0xaa) is written in the status register.

	7. Software exception c17ReadPsr is induced. The value in the status register immediately prior to the software exception induced by c17ReadPsr is compared with wPsrVal (0xaa), and the result of the comparison is saved in result. 8. The result value is checked. ◆ If the value is E_REGISTER E_REGISTER is returned. End. 9. E_OK is returned. End.

Program source

c17TestPsr.c
c17RwPsr.s

2.3.6 RAM March-C test

Performs a March-C test on the specified RAM area.

Function name	c17TestRamMarchc
Parameter 1	volatile unsigned char *chkAddr
Parameter 2	unsigned short chkNum
Return value	short (E_OK or E_MEMORY)
Function overview	March-C tests are run from the address indicated by chkAddr in the quantity specified by chkNum. E_OK is returned if there is no problem, and E_MEMORY if there is.
Caution	Operation is not guaranteed if the specified memory area overlaps the stack area. The area of memory tested is overwritten with 0x00.

Function name	c17TestBitChkw0
Parameter	volatile unsigned char *addr
Return value	short (E_OK or E_MEMORY)
Function overview	A function that writes 0's from the lower bit of specified byte and then performs comparison. 0's are written in order from the lower bit of the address indicated by addr. Then the written values and read values are compared. E_OK is returned if the values are the same, and E_MEMORY if they are different.

Function name	c17TestBitChkw1
Parameter	volatile unsigned char *addr
Return value	short (E_OK or E_MEMORY)
Function overview	A function that writes 1's from the lower bit of specified byte and then performs comparison. 1's are written in order from the lower bit of the address indicated by addr. Then the written values and read values are compared. E_OK is returned if the values are the same, and E_MEMORY if they are different.

2. Sample

Procedure

Test procedure	1. The ending address is found from the starting address and test scope. 2. 0 is written in all parts of the memory in the test scope. 3. c17TestBitChkw1 is called for all parts of the memory in the test scope from the starting address. 1 is written in order from the lower bit. ◆ If E_MEMORY is returned E_MEMORY is returned. End. 4. c17TestBitChkw0 is called for all parts of the memory in the test scope from the starting address. 0 is written in order from the lower bit. ◆ If E_MEMORY is returned E_MEMORY is returned. End. 5. c17TestBitChkw1 is called for all parts of the memory in the test scope from the ending address. 1 is written in order from the lower bit. ◆ If E_MEMORY is returned E_MEMORY is returned. End. 6. c17TestBitChkw0 is called for all parts of the memory in the test scope from the ending address. 0 is written in order from the lower bit. ◆ If E_MEMORY is returned E_MEMORY is returned. End. 7. All memory values in the test scope are checked to see if they are 0x00. ◆ If they are not 0x00 E_MEMORY is returned. End. 8. E_OK is returned. End.

Program source

c17TestRamMarchc.c

2.3.7 Checksum

Calculates the checksum of the specified memory area.

Function name	c17TestChksum
Parameter 1	unsigned char *chkAddr
Parameter 2	unsigned short chkNum
Return value	unsigned short
Function overview	The memory values are read from the address indicated by chkAddr in the quantity specified by chkNum, and their sum is returned.

Procedure

Test procedure	1. All memory values in the test scope are added to find the checksum. 2. The checksum value is returned.

Program source

c17TestChksum.c

2.3.8 CRC (calculation)

Calculates the CRC value of the specified memory area with a mathematical formula. The generating polynomial of the sample program is $x^{16} + x^{12} + x^5 + x^0$, and the default value is 0xffff.

Function name	c17TestCrc
Parameter 1	unsigned char *chkAddr
Parameter 2	unsigned short chkNum
Return value	unsigned short
Function overview	The memory values are read from the address indicated by chkAddr in the quantity specified by chkNum, and the CRC is calculated.

Procedure

Test procedure	1. The crc default value is 0xffff. Steps 2. and 3. are performed for all parts of the memory in the test scope. 2. The exclusive OR of the specified address shifted 8 bits to the left and crc make the new crc. 3. The steps below are repeated 8 times. The highest-order bit of crc is checked to see whether it is 1. ◆ If it is 1 The exclusive OR of crc shifted 1 bit to the left and 0x1021 make the new crc. ◆ If the value is other than 1 crc shifted 1 bit to the left makes the new crc. 4. crc reversed makes the new crc. 5. The crc value is returned. End.

Program source

c17TestCrc.c

2.3.9 CRC (table lookup)

Finds the CRC value of the specified memory area using the CRC-CCITT table. This can be found faster than by calculation by computation.

Function name	c17TestCrcTbl
Parameter 1	unsigned char *chkAddr
Parameter 2	unsigned short chkNum
Return value	unsigned short
Function overview	The memory values are read from the address indicated by chkAddr in the quantity specified by chkNum. Then the CRC is found by table lookup.

Procedure

Test procedure	1. The crc default value is 0xffff. Step 2. is performed for all parts of the memory in the test scope. 2. The exclusive OR of the table shown below and crc shifted 8 bits to the left makes the new crc. The index of the table is the exclusive OR of the memory value and crc shifted 8 bits to the right. 3. crc reversed makes the new crc. 4. The crc value is returned. End.

Program source

c17TestCrcTbl.c

2.3.10 Interrupt test

Counts how many times the tested interrupt occurs until the interrupts occurring in a fixed period reached the specified number.

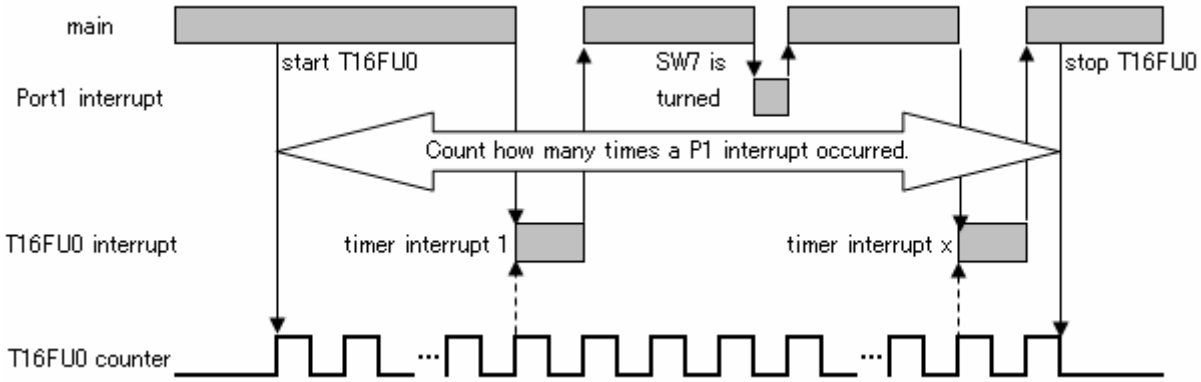

Figure 2.3.10 Diagram of the interrupt test

Function name	c17803TestIntterupt
Parameter	unsigned short numInt
Return value	unsigned short
Function overview	Counts how many times interrupts occur (using Port 13 (SW7) interrupts) until the number of interrupts indicated by numInt (using the CLG 16-bit timer).

Function name	c17803T16FInt
Parameter	void
Return value	void
Function overview	16-bit timer interrupt handler.

Function name	c17803P1Int
Parameter	void
Return value	void
Function overview	P1 port interrupt handler.

Function name	c17803InitIntTestInt
Parameter	void
Return value	void
Function overview	The initialization function of the interrupt function used by c17803TestInterrupt.

Procedure

Test procedure	1. Default setting of the counter value, 16-bit timer interrupt, and Port 1 interrupt is performed. 2. Stand by until the number of 16-bit timer interrupts indicated in numInt occur. ◆ If a port interrupt occurs during this time The return value counter is incremented by 1. 3. The 16-bit timer is stopped, and port interrupts are prohibited. 4. The counter value is returned. End.

Program source

c17803TestInterrupt.c

2.3.11 Main clock stability test

A test to check whether the operation of the main clock is within the specified error range.

This test uses the subclock RTC (32KHz).

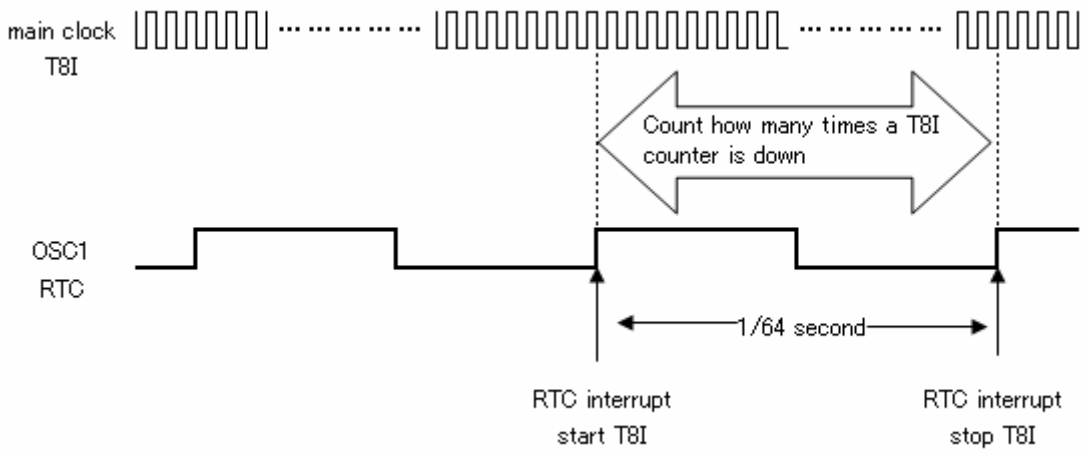

Figure 2.3.11 Diagram of the main clock stability test

Function name	c17803TestClk
Parameter 1	unsigned long baseFreq
Parameter 2	unsigned short range
Return value	int (E_OK or E_CLOCK)
Function overview	Checks whether the main clock is operating within the permitted error range. The ideal frequency value is specified with baseFreq, and the permitted error (%) is specified with range. If the test results are within the permitted error range, E_OK is returned, otherwise E_CLOCK is returned.

Function name	c17803RtcInt
Parameter	void
Return value	void
Function overview	Real-time clock interrupt handler

Function name	c17803T8IInt
Parameter	void
Return value	void
Function overview	8-bit timer interrupt handler

Function name	c17803InitIntTestClk
Parameter	void
Return value	void
Function overview	The initialization function of the interrupt function used by c17803TestClk.

Procedure

Test procedure	1. Default setting of the real-time clock and 8-bit timer is performed. 2. 1/64-second interrupt of the real-time clock is induced. In order to allow the real-time clock interrupts to stabilize, the 8-bit timer is started with the first interrupt and measurement is performed on the second interrupt. ◆ During this time, the 8-bit timer is made to count down by the main clock. 3. The 8-bit timer value is read out. 4. The main clock frequency is found from the value in 3. The main clock frequency is the value in 3. multiplied by 64 and the inverse number of the dividing ratio. 5. The value from 4. is checked to see whether it is within the permitted error. ◆ If it is within the permitted error E_OK is returned. End. ◆ If it is outside the permitted error E_CLOCK is returned. End.

Program source

c17803TestClk.c

3. Reference Information

This section covers reference information concerning programming and hardware.

3.1 Diagnosis of communication errors with external interfaces

In order to check that data is not corrupted when it is sent, it is necessary to guarantee all communication as far as possible using transmitted data redundancy. One means of assuring the integrity of the communication is to use the sample program CRC.

Furthermore, to avoid endless loops caused by communication errors, it can be effective to implement timeout detection.

Supplement:

It is important both to detect communication errors and to achieve a configuration where communication errors do not readily occur. Since signal conductors on the circuit run parallel with other digital lines, they may be affected by the change-point from 1 to 0 and 0 to 1 of these signals. If the signal conductor is surrounded by several other conductors that change concurrently, the signal is accompanied by significant noise. Therefore it is best to make these parallel sections short, and keep signals apart.

Furthermore, in order to prevent electromagnetic induction noise caused by mutual inductance, avoid arranging signal conductors with a large current close to circuits that are comparatively vulnerable to noise such as oscillators, analog input devices and the like.

Revision History

Attachment-1

Code No.	Page	Contents
411808700	All	First edition

EPSON

International Sales Operations

AMERICA

EPSON ELECTRONICS AMERICA, INC.
2580 Orchard Parkway,
San Jose, CA 95131, USA
Phone: +1-800-228-3964 FAX: +1-408-922-0238

EUROPE

EPSON EUROPE ELECTRONICS GmbH
Riesstrasse 15, 80992 Munich,
GERMANY
Phone: +49-89-14005-0 FAX: +49-89-14005-110

ASIA

EPSON (CHINA) CO., LTD.
7F, Jinbao Bldg., No.89 Jinbao St.,
Dongcheng District,
Beijing 100005, CHINA
Phone: +86-10-6410-6655 FAX: +86-10-6410-7320

SHANGHAI BRANCH
7F, Block B, Hi-Tech Bldg., 900 Yishan Road,
Shanghai 200233, CHINA
Phone: +86-21-5423-5522 FAX: +86-21-5423-5512

SHENZHEN BRANCH
12F, Dawning Mansion, Keji South 12th Road,
Hi-Tech Park, Shenzhen 518057, CHINA
Phone: +86-755-2699-3828 FAX: +86-755-2699-3838

EPSON HONG KONG LTD.
20/F, Harbour Centre, 25 Harbour Road,
Wanchai, Hong Kong
Phone: +852-2585-4600 FAX: +852-2827-4346
Telex: 65542 EPSCO HX

EPSON TAIWAN TECHNOLOGY & TRADING LTD.
14F, No. 7, Song Ren Road,
Taipei 110, TAIWAN
Phone: +886-2-8786-6688 FAX: +886-2-8786-6660

EPSON SINGAPORE PTE., LTD.
1 HarbourFront Place,
#03-02 HarbourFront Tower One, Singapore 098633
Phone: +65-6586-5500 FAX: +65-6271-3182

SEIKO EPSON CORP.
KOREA OFFICE
50F, KLI 63 Bldg., 60 Yoido-dong,
Youngdeungpo-Ku, Seoul 150-763, KOREA
Phone: +82-2-784-6027 FAX: +82-2-767-3677

SEIKO EPSON CORP.
SEMICONDUCTOR OPERATIONS DIVISION

IC Sales Dept.
IC International Sales Group
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5814 FAX: +81-42-587-5117

Document Code: 411808700
First Issue September 2009 in JAPAN Ⓥ