AVR319: Using the USI module for SPI communication

8-bit AVR® Microcontrollers

Application Note

Features

- **C-code driver for SPI master and slave**
- **Uses the USI module**
- **Supports SPI Mode 0 and 1**

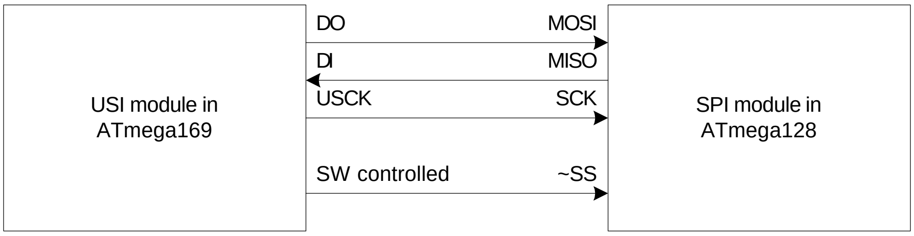
Introduction

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between an AVR device and peripheral devices or between several AVR devices. The strength of the SPI bus includes ease of use, high communication speed and a vast amount of peripheral devices supporting it.

The Universal Serial Interface (USI) module on devices like ATmega169, ATtiny26 and ATtiny2313 has a dedicated Three-wire mode. The USI provides the basic hardware resources needed for synchronous serial communication. Combined with a minimum of control software, the USI allows higher transfer rates, less CPU load and in general uses less code space than solutions based on software only.

This application note describes a SPI interface implementation, in form of a full-featured driver and an example of usage for this driver. The driver handles transmission according to SPI Modes 0 and 1.

Figure 1. Example application setup



Rev. 2582A-AVR-09/04

Theory

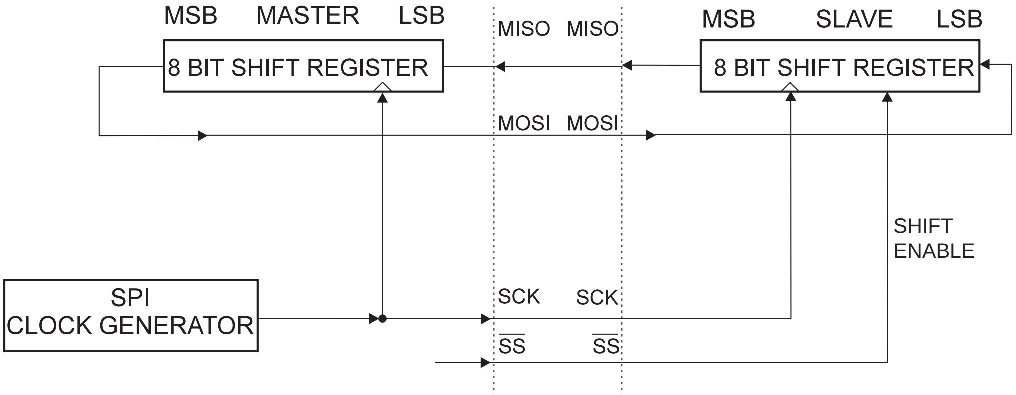
This section gives a short description of the SPI interface and the USI module. For more detailed information refer to the datasheets.

Serial Peripheral Interface

The Serial Peripheral Interface allows high-speed synchronous data transfer between an AVR device and peripheral devices or between several AVR devices.

The interconnection between Master and Slave devices with SPI is shown in Figure 2. The system consists of two shift registers, and a master clock generator. The SPI Master initiates the communication cycle when pulling low the Slave Select (SS) pin of the desired Slave. Master and Slave prepare the data to be sent in their respective shift registers, and the Master generates the required clock pulses on the SCK line to interchange data.

Figure 2. SPI Master-slave interconnection



SPI Data Modes

The four combinations of SCK phase and polarity with respect to serial data are determined by clock phase (CPHA) and clock polarity (CPOL) settings. The SPI data transfer formats are shown in Figure 3 and Figure 4 below. Data bits are shifted out and latched in on opposite edges of the SCK signal, ensuring sufficient time for data signals to stabilize.

Figure 3. SPI Transfer Format with CPHA = 0

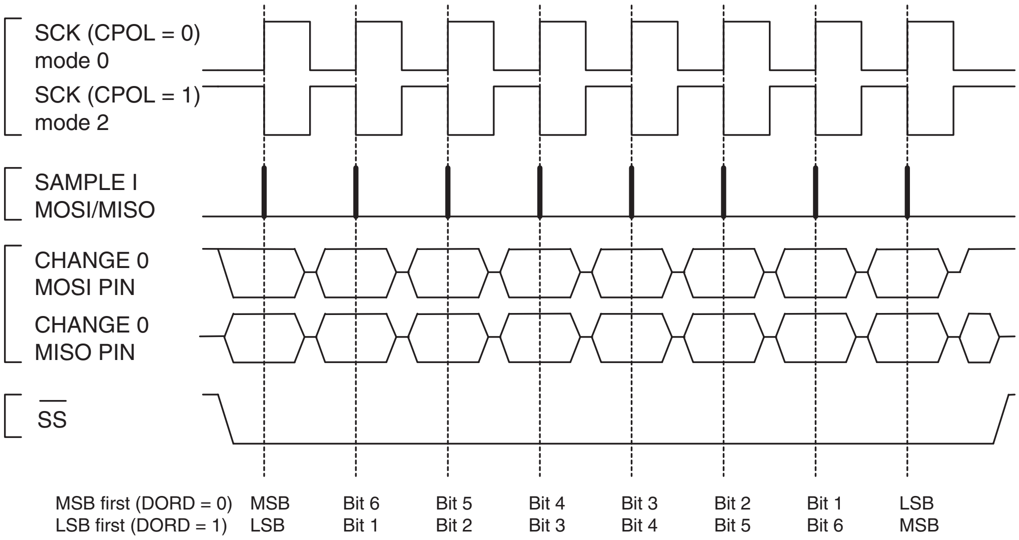
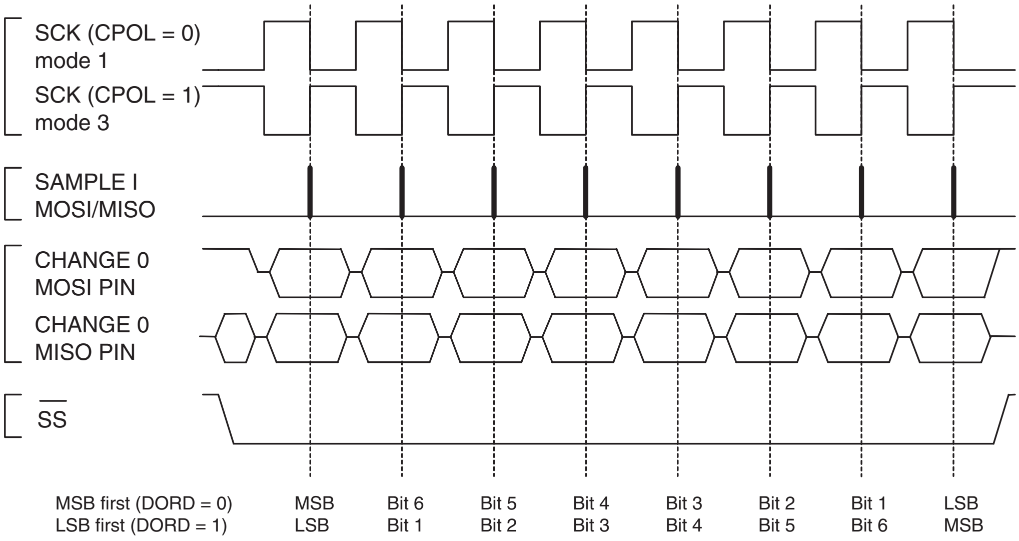


Figure 4. SPI Transfer Format with CPHA = 1



Universal Serial Interface

The Universal Serial Interface provides the basic hardware resources needed for synchronous serial communication. The main features of the USI are:

- Two-wire Synchronous Data Transfer
- Three-wire Synchronous Data Transfer
- Data Received Interrupt
- Wakeup from Idle Mode

The USI Three-wire mode is compliant with the Serial Peripheral Interface (SPI) mode 0 and 1, but does not have the slave select (SS) pin functionality. However, this feature can be implemented in software if necessary. Figure 5 below shows the USI module block diagram, and Figure 6 shows the module in Three-wire mode.

Figure 5. Universal Serial Interface, Block Diagram

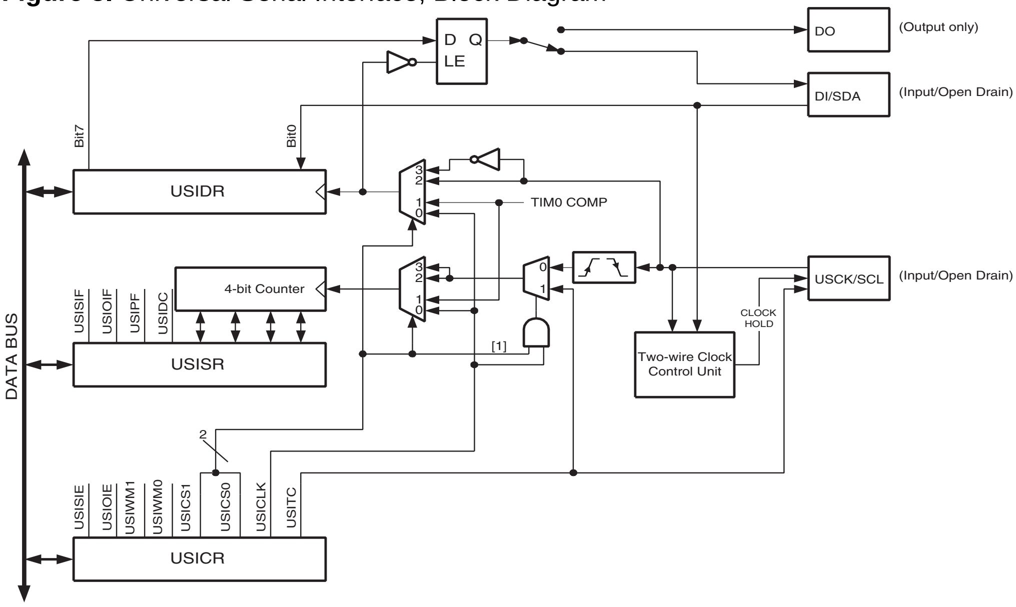
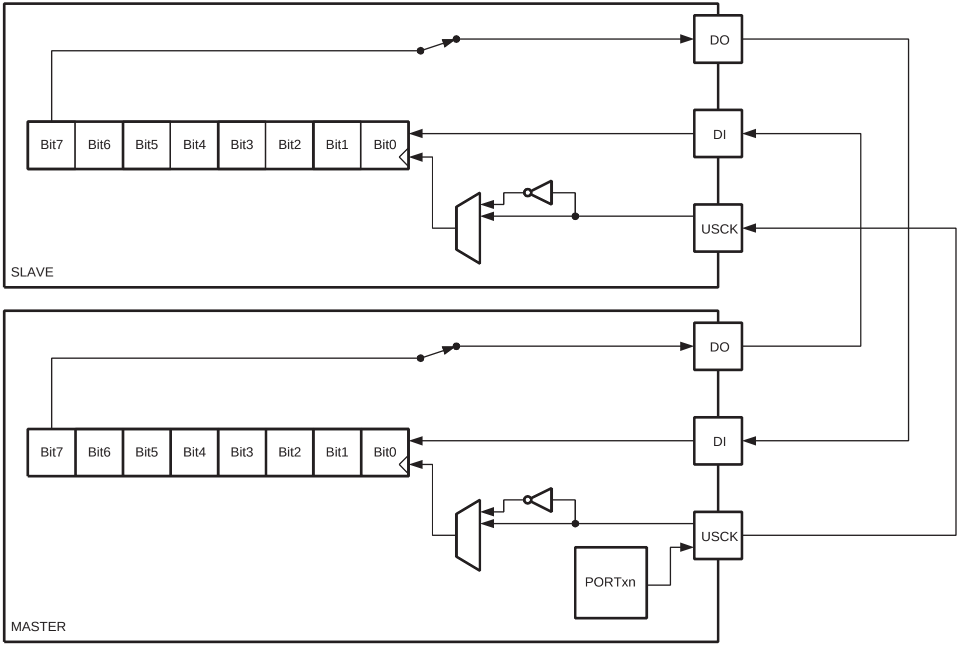


Figure 6. Three-wire Mode Operation, Simplified Diagram



The USI Data Register (USIDR) is an 8-bit Shift Register that contains the incoming and outgoing data. The register has no buffering so the data must be read as quickly as possible to ensure that no data is lost. The USI Status Register (USISR) contains a 4-bit counter. Both the shift register and the counter are clocked simultaneously by the same clock source. This allows the counter to count the number of bits received or transmitted and sets a flag alternatively generates an interrupt when the transfer is complete. The clock can be selected to use three different sources: The USCK pin, Timer/Counter0 Compare Match or from software.

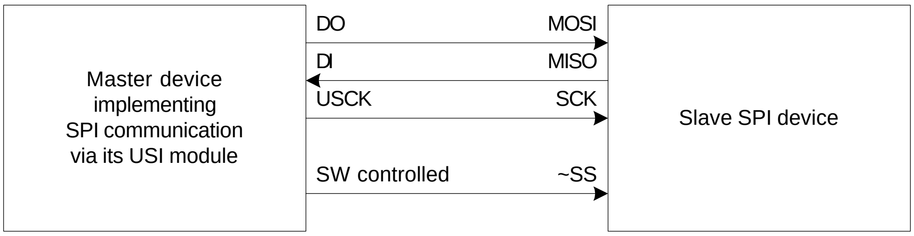
Figure 6 shows two USI units operating in Three-wire mode, one as Master and one as Slave. The two shift registers are interconnected in such way that after eight USCK clocks, the data in each register are interchanged. The same clock also increments the USI's 4-bit counter. The Counter Overflow Interrupt Flag, or USIOIF, can therefore be used to determine when a transfer is completed. The clock is generated by the Master device software by toggling the USCK pin via the PORT Register or by writing a one to the USITC bit in USICR.

It is the master device's responsibility to give the slave device time to prepare its next byte before starting a new transfer.

Implementation

This application note describes the implementation of a SPI driver for both master and slave communication. An example setup with the USI module as a SPI master is shown in Figure 7 below. The driver is written as a standalone driver that easily can be included into the main application. Use the code as an example, or customize it for own use. All relevant functions and global variables are prefixed with '`spiX_`', so a quick search-and-replace is enough to rename the driver interface in case of naming conflicts.

Figure 7. USI module setup as SPI master



The driver uses the USI module and the USI counter overflow interrupt. Therefore, interrupts must be enabled to be able to use the driver. In master mode the driver also uses Timer/Counter0. The T/C0 compare match interrupt is used to generate the master clock signal.

The driver interface consists of these functions:

- *spiX_initmaster* which initializes the driver in master mode.
- *spiX_initslave* which initializes the driver in slave mode.
- *spiX_put* which starts a transfer in master mode, or prepares a byte in slave mode.
- *spiX_get* which returns the last incoming byte.
- *spiX_wait* which waits for a transfer to finish.

Note that the Slave Select (SS) line of the SPI bus must be controlled manually in software if required by the slave device. When using the USI as a SPI slave, you also need to watch the incoming SS line with for instance an interrupt line if required. The driver in this application note does not use the SS line.

The following global variables are also available:

- *spiX_status* which contains the driver status flags.
- *storedUSIDR* which contains the last incoming byte. This should be accessed with the `spiX_get()` function only.

The functions are documented in the source code. In addition, the simplified flowchart for a typical byte transfer session is given in Figure 8 below. The example source code accompanying this application note shows an implementation of such a session.

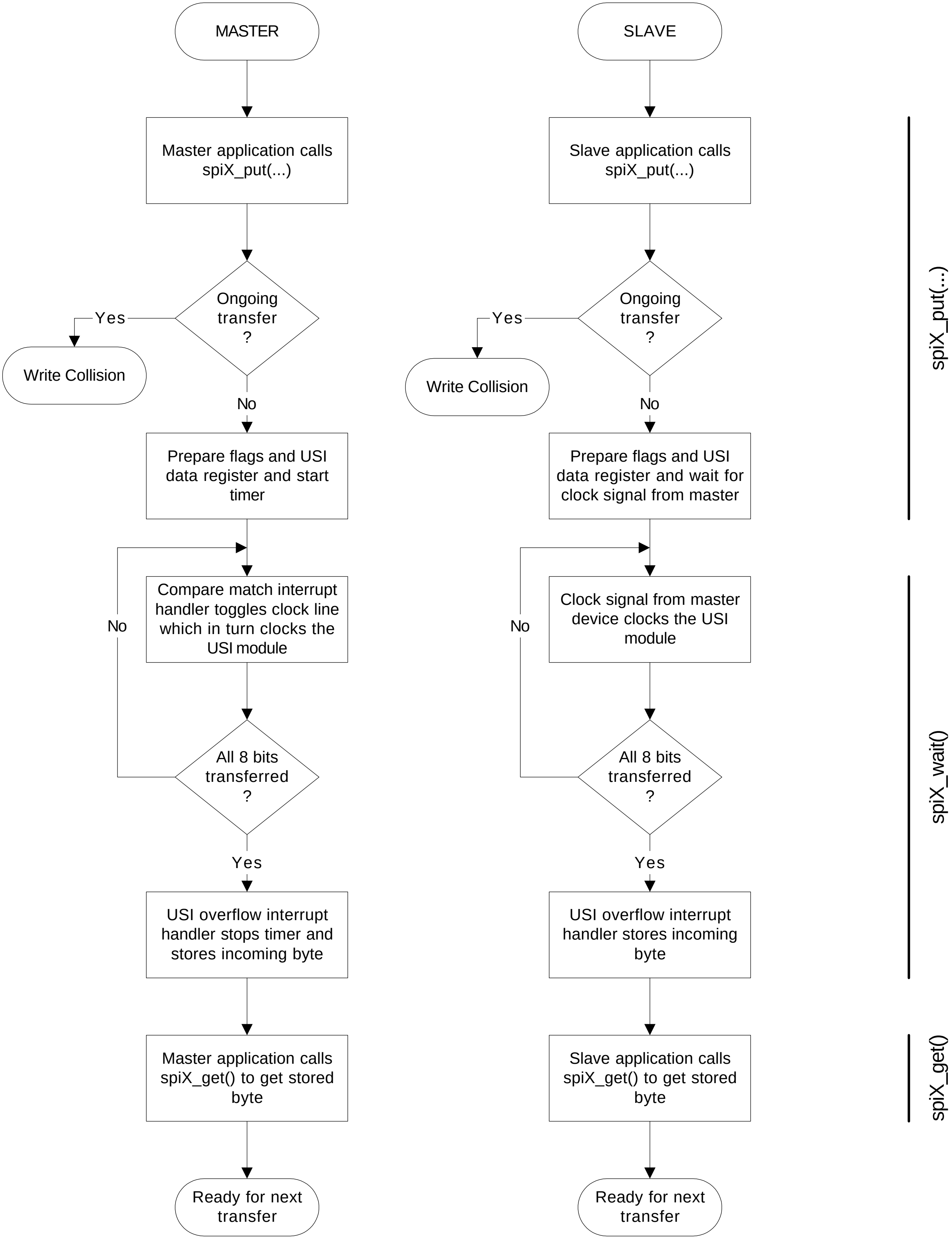
Note that the part of the flowchart below grouped under the `spiX_wait()` function is performed by interrupt handlers and not the function itself. The function just waits for the transfer complete flag to be set.

The example code is written for the IAR Embedded Workbench AVR C compiler version 3.2.

Literature References

- ATmega169 Datasheet
- Application Note AVR310: Using the USI module as a TWI master

Figure 8. One byte transfer in master and slave mode


MASTER
SLAVE
Master application calls spiX_put(...)
Slave application calls spiX_put(...)
Ongoing transfer ?
Yes
Write Collision
No
Ongoing transfer ?
Yes
Write Collision
No
Prepare flags and USI data register and start timer
Prepare flags and USI data register and wait for clock signal from master
Compare match interrupt handler toggles clock line which in turn clocks the USI module
No
Clock signal from master device clocks the USI module
No
All 8 bits transferred ?
Yes
All 8 bits transferred ?
Yes
USI overflow interrupt handler stops timer and stores incoming byte
USI overflow interrupt handler stores incoming byte
Master application calls spiX_get() to get stored byte
Slave application calls spiX_get() to get stored byte
Ready for next transfer
Ready for next transfer
spiX_put(...)
spiX_wait()
spiX_get()

Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. **EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.** Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Atmel's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.