
TWI Program Examples

References

- Atmel 8051 Microcontrollers Hardware Manual



**8051
Microcontrollers**

Application Note

Rev. 4353A–80C51–07/04





1. Introduction

This Application Note provides to customers C and Assembler program examples for TWI.

These examples are developed for the different configuration modes of this feature.

2. C Examples

2.1 Program

```
/**
 * @file $RCSfile: TWI.c,v $
 *
 * Copyright (c) 2004 Atmel.
 *
 * Please read file license.txt for copyright notice.
 *
 * @brief This file is an example to use TWI.
 *
 * This file can be parsed by Doxygen for automatic documentation
 * generation.
 * Put here the functional description of this file within the software
 * architecture of your program.
 *
 * @version $Revision: 1.0 $ $Name: $
 */
/* @section I N C L U D E S */
#include "reg_C51.h"
char TWI_data;
char rw=0; /* 0=write, 1=read */
char slave_adr;
bit b_TWI_busy=0;
/**
 * FUNCTION_PURPOSE: this function setup TWI in master mode and sends data to
 * slave.
 * FUNCTION_INPUTS: void
 * FUNCTION_OUTPUTS: void
 */
void main(void)
{
    SSCON |= 0x40; /* enable TWI */
    EA=1; /* interrupt enable */
    IEN1 |= 0x02; /* enable TWI interrupt */

    while(1) /* end less */
    {
        if(!b_TWI_busy && ((SSCON&0x10)!=0x10)) /* if the TWI is free */
        {
            b_TWI_busy=1; /* flag busy =1 */
            TWI_data=0x55; /* data example to send */
            slave_adr=0x01; /* slave adresse example */
            rw=0; /* 0=write */
            SSDAT = 0x00; /* clear buffer before sending data */
            SSCON |= 0x20; /* TWI start sending */
        }
    }
}
```

```

/**
 * FUNCTION_PURPOSE:TWI interrupt, task witch process the different status of
 * TWI
 * FUNCTION_INPUTS:void
 * FUNCTION_OUTPUTS:void
 */
void it_TWI(void) interrupt 8 using 1
{

switch(SSCS)                                /* TWI status tasking */
{
    case(0x00):                             /* A start condition has been sent */
    {                                       /* SLR+R/W are transmitted, ACK bit received */
        b_TWI_busy=0;                    /* TWI is free */
        break;
    }
    case(0x08):                             /* A start condition has been sent */
    {                                       /* SLR+R/W are transmitted, ACK bit received */
        SSCON &= ~0x20;                  /* clear start condition */
        SSDAT = (slave_adr<<1)|rw; /* send slave adress and read/write bit */
        SSCON |= 0x04;                   /* set AA */
        break;
    }

    case(0x10):                             /* A repeated start condition has been sent */
    {                                       /* SLR+R/W are transmitted, ACK bit received */
        SSCON &= ~0x20;                  /* clear start condition */
        SSDAT = (slave_adr<<1)|rw; /* send slave adress and read/write bit */
        SSCON |= 0x04;                   /* set AA */
        break;
    }

    case(0x18):                             /* SLR+W was transmitted, ACK bit received */
    {

        SSDAT = TWI_data;                /* Transmit data byte, ACK bit received */
        SSCON |= 0x04;                   /* set AA */
        break;
    }

    case(0x20):                             /* SLR+W was transmitted, NOT ACK bit received */
    {
        SSCON |= 0x10;                   /* Transmit STOP */
        b_TWI_busy=0;                    /* TWI is free */
        break;
    }
}

```

```

        case(0x28):                /* DATA was transmitted, ACK bit received */
        {
            SSCON |= 0x10;          /* send STOP */
            b_TWI_busy=0;           /* TWI is free */
            break;
        }

        case(0x30):                /* DATA was transmitted, NOT ACK bit received */
        {
            SSCON |= 0x10;          /* Transmit STOP */
            b_TWI_busy=0;           /* TWI is free */
            break;
        }

        case(0x38):                /* Arbitration lost in SLA+W or DATA. */
        {
            SSCON |= 0x10;          /* Transmit STOP */
            b_TWI_busy=0;           /* TWI is free */
            break;
        }
    }
    SSCON &= ~0x08;                /* clear flag */
}

```

2.2 SFR Register Definition

```

/*H*****
**
* NAME: AT89C51XD2.h
*-----
-
* PURPOSE: SFR Description file for AT89C51xD2 products
*         ON KEIL compiler
*****
*/

#define Sfr(x, y)  sfr x = y
#define Sbit(x, y, z)  sbit x = y^z
#define Sfr16(x,y)    sfr16 x = y

/*-----*/
/* Include file for 8051 SFR Definitions */
/*-----*/

/* BYTE Register */
Sfr (P0 , 0x80);

Sbit (P0_7 , 0x80, 7);
Sbit (P0_6 , 0x80, 6);
Sbit (P0_5 , 0x80, 5);
Sbit (P0_4 , 0x80, 4);
Sbit (P0_3 , 0x80, 3);
Sbit (P0_2 , 0x80, 2);
Sbit (P0_1 , 0x80, 1);
Sbit (P0_0 , 0x80, 0);

Sfr (P1 , 0x90);

Sbit (P1_7 , 0x90, 7);
Sbit (P1_6 , 0x90, 6);
Sbit (P1_5 , 0x90, 5);
Sbit (P1_4 , 0x90, 4);
Sbit (P1_3 , 0x90, 3);
Sbit (P1_2 , 0x90, 2);
Sbit (P1_1 , 0x90, 1);
Sbit (P1_0 , 0x90, 0);

Sfr (P2 , 0xA0);
Sbit (P2_7 , 0xA0, 7);
Sbit (P2_6 , 0xA0, 6);
Sbit (P2_5 , 0xA0, 5);
Sbit (P2_4 , 0xA0, 4);
Sbit (P2_3 , 0xA0, 3);

```

```
Sbit (P2_2 , 0xA0, 2);
Sbit (P2_1 , 0xA0, 1);
Sbit (P2_0 , 0xA0, 0);
```

```
Sfr (P3 , 0xB0);
```

```
Sbit (P3_7 , 0xB0, 7);
Sbit (P3_6 , 0xB0, 6);
Sbit (P3_5 , 0xB0, 5);
Sbit (P3_4 , 0xB0, 4);
Sbit (P3_3 , 0xB0, 3);
Sbit (P3_2 , 0xB0, 2);
Sbit (P3_1 , 0xB0, 1);
Sbit (P3_0 , 0xB0, 0);
```

```
Sbit (RD , 0xB0, 7);
Sbit (WR , 0xB0, 6);
Sbit (T1 , 0xB0, 5);
Sbit (T0 , 0xB0, 4);
Sbit (INT1 , 0xB0, 3);
Sbit (INT0 , 0xB0, 2);
Sbit (TXD , 0xB0, 1);
Sbit (RXD , 0xB0, 0);
```

```
Sfr (P4 , 0xC0);
```

```
Sbit (P4_7 , 0xC0, 7);
Sbit (P4_6 , 0xC0, 6);
Sbit (P4_5 , 0xC0, 5);
Sbit (P4_4 , 0xC0, 4);
Sbit (P4_3 , 0xC0, 3);
Sbit (P4_2 , 0xC0, 2);
Sbit (P4_1 , 0xC0, 1);
Sbit (P4_0 , 0xC0, 0);
```

```
Sfr (P5 , 0xE8);
```

```
Sbit (P5_7 , 0xE8, 7);
Sbit (P5_6 , 0xE8, 6);
Sbit (P5_5 , 0xE8, 5);
Sbit (P5_4 , 0xE8, 4);
Sbit (P5_3 , 0xE8, 3);
Sbit (P5_2 , 0xE8, 2);
Sbit (P5_1 , 0xE8, 1);
Sbit (P5_0 , 0xE8, 0);
```

```
Sfr (PSW , 0xD0);
```

```
Sbit (CY , 0xD0 , 7);
Sbit (AC , 0xD0 , 6);
```

```

Sbit (F0 , 0xD0 , 5);
Sbit (RS1 , 0xD0 , 4);
Sbit (RS0 , 0xD0 , 3);
Sbit (OV , 0xD0 , 2);
Sbit (UD , 0xD0 , 1);
Sbit (P , 0xD0 , 0);

Sfr (ACC , 0xE0);
Sfr (B , 0xF0);
Sfr (SP , 0x81);
Sfr (DPL , 0x82);
Sfr (DPH , 0x83);

Sfr (PCON , 0x87);
Sfr (CKCON0 , 0x8F);
Sfr (CKCON1 , 0xAF);

/*----- TIMERS registers -----*/
Sfr (TCON , 0x88);
Sbit (TF1 , 0x88, 7);
Sbit (TR1 , 0x88, 6);
Sbit (TF0 , 0x88, 5);
Sbit (TR0 , 0x88, 4);
Sbit (IE1 , 0x88, 3);
Sbit (IT1 , 0x88, 2);
Sbit (IE0 , 0x88, 1);
Sbit (IT0 , 0x88, 0);

Sfr (TMOD , 0x89);

Sfr (T2CON , 0xC8);
Sbit (TF2 , 0xC8, 7);
Sbit (EXF2 , 0xC8, 6);
Sbit (RCLK , 0xC8, 5);
Sbit (TCLK , 0xC8, 4);
Sbit (EXEN2 , 0xC8, 3);
Sbit (TR2 , 0xC8, 2);
Sbit (C_T2 , 0xC8, 1);
Sbit (CP_RL2 , 0xC8, 0);

Sfr (T2MOD , 0xC9);
Sfr (TL0 , 0x8A);
Sfr (TL1 , 0x8B);
Sfr (TL2 , 0xCC);
Sfr (TH0 , 0x8C);
Sfr (TH1 , 0x8D);
Sfr (TH2 , 0xCD);
Sfr (RCAP2L , 0xCA);
Sfr (RCAP2H , 0xCB);
Sfr (WDTRST , 0xA6);

```

```
Sfr (WDTPRG , 0xA7);
```

```
/*----- UART registers -----*/
```

```
Sfr (SCON , 0x98);  
Sbit (SM0 , 0x98, 7);  
Sbit (FE , 0x98, 7);  
Sbit (SM1 , 0x98, 6);  
Sbit (SM2 , 0x98, 5);  
Sbit (REN , 0x98, 4);  
Sbit (TB8 , 0x98, 3);  
Sbit (RB8 , 0x98, 2);  
Sbit (TI , 0x98, 1);  
Sbit (RI , 0x98, 0);
```

```
Sfr (SBUF , 0x99);  
Sfr (SADEN , 0xB9);  
Sfr (SADDR , 0xA9);
```

```
/*----- Internal Baud Rate Generator -----*/
```

```
Sfr (BRL , 0x9A);  
Sfr (BDRCON , 0x9B);
```

```
/*----- IT registers -----*/
```

```
Sfr (IEN0 , 0xA8);  
Sfr (IEN1 , 0xB1);  
Sfr (IPH0 , 0xB7);  
Sfr (IPH1 , 0xB3);  
Sfr (IPL0 , 0xB8);  
Sfr (IPL1 , 0xB2);
```

```
/* IEN0 */
```

```
Sbit (EA , 0xA8, 7);  
Sbit (EC , 0xA8, 6);  
Sbit (ET2 , 0xA8, 5);  
Sbit (ES , 0xA8, 4);  
Sbit (ET1 , 0xA8, 3);  
Sbit (EX1 , 0xA8, 2);  
Sbit (ET0 , 0xA8, 1);  
Sbit (EX0 , 0xA8, 0);
```

```
/*----- PCA registers -----*/
```

```
Sfr (CCON , 0xD8);  
Sfr (CMOD , 0xD9);  
Sfr (CH , 0xF9);
```

```

Sfr (CL , 0xE9);
Sfr (CCAP0H , 0xFA);
Sfr (CCAP0L , 0xEA);
Sfr (CCAPM0 , 0xDA);
Sfr (CCAP1H , 0xFB);
Sfr (CCAP1L , 0xEB);
Sfr (CCAPM1 , 0xDB);
Sfr (CCAP2H , 0xFC);
Sfr (CCAP2L , 0xEC);
Sfr (CCAPM2 , 0xDC);
Sfr (CCAP3H , 0xFD);
Sfr (CCAP3L , 0xED);
Sfr (CCAPM3 , 0xDD);
Sfr (CCAP4H , 0xFE);
Sfr (CCAP4L , 0xEE);
Sfr (CCAPM4 , 0xDE);
/* CCON */
Sbit (CF , 0xD8, 7);
Sbit (CR , 0xD8, 6);

Sbit (CCF4 , 0xD8, 4);
Sbit (CCF3 , 0xD8, 3);
Sbit (CCF2 , 0xD8, 2);
Sbit (CCF1 , 0xD8, 1);
Sbit (CCF0 , 0xD8, 0);

/*----- T W I registers -----*/
Sfr (SSCON , 0x93);
Sfr (SSCS , 0x94);
Sfr (SSDAT , 0x95);
Sfr (SSADR , 0x96);
Sfr (PI2, 0xF8);
Sbit (PI2_1 , 0xF8, 1);
Sbit (PI2_0 , 0xF8, 0);

/*----- OSC control registers -----*/
Sfr (CKSEL , 0x85 );
Sfr (OSCCON , 0x86 );
Sfr (CKRL , 0x97 );

/*----- Keyboard control registers -----*/
Sfr (KBLS , 0x9C );
Sfr (KBE , 0x9D );
Sfr (KBF , 0x9E );
/*----- SPI -----*/
Sfr (SPCON, 0xC3 );
Sfr (SPSTA, 0xC4 );
Sfr (SPDAT, 0xC5 );

```

```
/*----- Misc -----*/  
Sfr( AUXR , 0x8E);  
Sfr ( AUXR1, 0xA2);  
Sfr ( FCON, 0xD1);
```

```
/*----- E data -----*/  
  
Sfr ( EECON, 0xD2 );
```

3. Assembler 51 Examples

3.1 Program

```

$INCLUDE    (reg_c51.INC)

TWI_data DATA 10H;
slave_adr DATA 11H;
rw BIT 20H;                                /* 0=write, 1=read */
b_TWI_busy BIT 21H;

org 000h
ljmp begin
org 43h
ljmp twi_it

;/**
; * FUNCTION_PURPOSE:this function setup TWI in master mode and sends data to
; slave.
; * FUNCTION_INPUTS:void
; * FUNCTION_OUTPUTS:void
; */
org 0100h

begin:

ORL SCON,#40h;                             /* enable TWI */
SETB EA;                                   /* interrupt enable */
ORL IEN1,#02h;                             /* enable TWI interrupt */

CLR b_TWI_busy
loop:                                       /* end less */

JB b_TWI_busy,end_if
MOV ACC,SCON
JB ACC.4,end_if
    SETB b_TWI_busy;                       /* flag busy =1 */
    MOV TWI_data,#55h;                     /* data example to send */
    MOV slave_adr,#01h;                    /* slave adresse example */
    CLR rw;                                /* 0=write */
    MOV SSDAT,#00h;                        /* clear buffer before sending data */
    ORL SCON,#20h;                          /* TWI start sending */
end_if:

JMP loop

```

```

;/**
; * FUNCTION_PURPOSE:TWI interrupt, task witch process the different status
; * of TWI
; * FUNCTION_INPUTS:void
; * FUNCTION_OUTPUTS:void
; */
twi_it:
MOV R7,SSCS

;/* TWI status tasking */

CJNE R7,#00h,end_case_00;      /* A start condition has been sent */
                                /* SLR+R/W are transmitted, ACK bit received */
CLR b_TWI_busy;                /* TWI is free */
JMP end_switch
end_case_00:

CJNE R7,#08h,end_case_08;      /* A start condition has been sent */
                                /* SLR+R/W are transmitted, ACK bit received */
ANL SCON,#~20h;                /* clear start condition */
/* send slave adress and read/write bit */
MOV ACC,slave_adr
RL A
MOV C,rw
MOV ACC.0,C
MOV SSDAT,A

ORL SCON,#04h;                  /* set AA */
JMP end_switch
end_case_08:

CJNE R7,#10h,end_case_10; /* A repeated start condition has been sent */
                                /* SLR+R/W are transmitted, ACK bit received */
ANL SCON,#~20h;                /* clear start condition */
/* send slave adress and read/write bit */
MOV ACC,slave_adr
RL A
MOV C,rw
MOV ACC.0,C
MOV SSDAT,A

ORL SCON,#04h;                  /* set AA */
JMP end_switch
end_case_10:

```

```

CJNE R7,#18h,end_case_18; /* SLR+W was transmitted, ACK bit received */
MOV SSDAT,TWI_data;      /* Transmit data byte, ACK bit received */
ORL SCON,#04h;            /* set AA */
JMP end_switch
end_case_18:

CJNE R7,#20h,end_case_20; /*SLR+W was transmitted,NOT ACK bit received*/
ORL SCON,#10h;            /* Transmit STOP */
CLR b_TWI_busy;          /* TWI is free */
JMP end_switch
end_case_20:

CJNE R7,#28h,end_case_28; /* DATA was transmitted, ACK bit received */
ORL SCON,#10h;            /* send STOP */
CLR b_TWI_busy;          /* TWI is free */
JMP end_switch
end_case_28:

CJNE R7,#30h,end_case_30; /* DATA was transmitted, NOT ACK bit received*/
ORL SCON,#10h;            /* Transmit STOP */
CLR b_TWI_busy;          /* TWI is free */
JMP end_switch
end_case_30:

CJNE R7,#38h,end_case_38; /* Arbitration lost in SLA+W or DATA. */
ORL SCON,#10h;            /* Transmit STOP */
CLR b_TWI_busy;          /* TWI is free */
JMP end_switch
end_case_38:

end_switch:
ANL SCON,#~08h;           /* clear flag */
RETI

end

```

3.2 SFR Register Definition

\$SAVE
\$NOLIST

```

P0      DATA      80H
TCONDATA88H
;--- TCON Bits ---
TF1     BIT        8FH
TR1     BIT        8EH
TF0     BIT        8DH
TR0     BIT        8CH
IE1     BIT        8BH
IT1     BIT        8AH
IE0     BIT        89H
IT0     BIT        88H

P1      DATA      90H

SCON     DATA      98H
;--- SCON Bits ----
SM0     BIT        9FH
SM1     BIT        9EH
SM2     BIT        9DH
REN     BIT        9CH
TB8     BIT        9BH
RB8     BIT        9AH
TI      BIT        99H
RI      BIT        98H

P2      DATA      0A0H
IEN0     DATA      0A8H
;--- IEN0 Bits -----
EA      BIT0AFH
EC      BIT0AEH
ET2     BIT0ADH
ES      BIT0ACH
ET1     BIT0ABH
EX1     BIT0AAH
ET0     BIT0A9H
EX0     BIT0A8H

P3      DATA      0B0H
;--- P3 Bits -----
RD      BIT        0B7H
WR      BIT        0B6H
T1      BIT        0B5H
T0      BIT        0B4H
INT1     BIT        0B3H

```



```
INT0    BIT    0B2H
TXD     BIT    0B1H
RXD     BIT    0B0H
```

```
P4      DATA  0C0H
P5      DATA  0E8H
```

```
IPL0DATA0B8H
;--- IPL0 Bits -----
PPCL    BIT0BEH
PT2L    BIT0BDH
PSL     BIT0BCH
PT1L    BIT0BBH
PX1L    BIT0BAH
PT0L    BIT0B9H
PX0L    BIT0B8H
```

```
T2CON   DATA  0C8H
;--- T2CON bits ----
TF2     BIT    0CFH
EXF2    BIT    0CEH
RCLK    BIT    0CDH
TCLK    BIT    0CCH
EXEN2   BIT    0CBH
TR2     BIT    0CAH
C_T2    BIT    0C9H
CP_RL2  BIT    0C8H
```

```
PSW     DATA  0D0H
;--- PSW bits -----
CY      BIT    0D7H
AC      BIT    0D6H
F0      BIT    0D5H
RS1     BIT    0D4H
RS0     BIT    0D3H
OV      BIT    0D2H
P       BIT    0D0H
```

```
CCONDATA0D8H
;--- CCON bits -----
CF      BIT    0DFH
CR      BIT    0DEH
CCF4    BIT    0DCH
CCF3    BIT    0DBH
CCF2    BIT    0DAH
CCF1    BIT    0D9H
CCF0    BIT    0D8H
```

ACC	DATA	0E0H
B	DATA	0F0H

SP	DATA	81H
DPL	DATA	82H
DPH	DATA	83H
PCON	DATA	87H

TMOD	DATA	89H
TL0	DATA	8AH
TL1	DATA	8BH
TH0	DATA	8CH
TH1	DATA	8DH
AUXRDATA08EH		
CKCON0DATA08Fh		

SBUF	DATA	99H
;-- Baud Rate generator		
BRL	DATA09AH	
BDRCON	DATA 09BH	
;--- Keyboard		
KBLSDATA09CH		
KBEDATA09DH		
KBFDATA09EH		

;--- Watchdog timer		
WDRSTDATA0A6H		
WDTPRG	DATA0A7H	

SADDRDATA0A9H		
CKCON1DATA0AFH		

IEN1DATA0B1H		
IPL1DATA0B2H		
IPH1DATA0B3H		
IPH0DATA0B7H		

SADENDATA0B9H		
---------------	--	--

```
T2MODDATA      0C9h
RCAP2L  DATA   0CAH
RCAP2H  DATA   0CBH
TL2     DATA   0CCH
TH2     DATA   0CDH
```

```
CMODDATA0D9H
CCAPM0DATA0DAH
CCAPM1DATA0DBH
CCAPM2DATA0DCH
CCAPM3DATA0DDH
CCAPM4DATA0DEH
```

```
CHDATA0F9H
CCAP0HDATA0FAH
CCAP1HDATA0FBH
CCAP2HDATA0FCH
CCAP3HDATA0FDH
CCAP4HDATA0FEH
```

```
CLDATA0E9H
CCAP0LDATA0EAH
CCAP1LDATA0EBH
CCAP2LDATA0ECH
CCAP3LDATA0EDH
CCAP4LDATA0EEH
```

```
; SPI
SPCON    DATA   0C3H
SPSTA    DATA   0C4H
SPDAT    DATA   0C5H
```

```
; TWI
PI2DATA   0F8h
SSCONDATA093H
SSCSDATA094H
SSDATDATA095H
SSADRDATA096H
PI2_OBIT0F8H
PI2_1BIT0F9H
```

```
; Clock Control
OSCONDATA086H
CKSELDATA085H
CKRLDATA097H
```

```
;MISC
AUXR1DATA0A2H
```

```
; Flash control  
FCON DATA 0D1H
```

```
;EEData  
EECONDATA0D2H
```

```
$RESTORE
```



Atmel Corporation

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

Microcontrollers

2325 Orchard Parkway
San Jose, CA 95131
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
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
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

e-mail

literature@atmel.com

Web Site

<http://www.atmel.com>

Disclaimer: Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard warranty which is detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use as critical components in life support devices or systems.

©Atmel Corporation 2004. All rights reserved. Atmel, the Atmel logo, and combinations thereof are registered trademarks of Atmel Corporation or its subsidiaries. Windows® Windows 98™, Windows XP™, and Windows 2000™ are trademarks and/or registered trademark of Microsoft Corporation. Other terms and product names in this document may be the trademarks of others.



Printed on recycled paper.