



# ST72260G, ST72262G, ST72264G

## 8-BIT MCU WITH FLASH OR ROM MEMORY, ADC, TWO 16-BIT TIMERS, I<sup>2</sup>C, SPI, SCI INTERFACES

### ■ Memories

- 4 K or 8 Kbytes Program memory: ROM or Single voltage extended Flash (XFlash) with read-out protection write protection and In-Circuit Programming and In-Application Programming (ICP and IAP). 10K write/erase cycles guaranteed, data retention: 20 years at 55°C.
- 256 bytes RAM

### ■ Clock, Reset and Supply Management

- Enhanced reset system
- Enhanced low voltage supply supervisor (LVD) with 3 programmable levels and auxiliary voltage detector (AVD) with interrupt capability for implementing safe power-down procedures
- Clock sources: crystal/ceramic resonator oscillators, internal RC oscillator, clock security system and bypass for external clock
- PLL for 2x frequency multiplication
- Clock-out capability
- 4 Power Saving Modes: Halt, Active Halt, Wait and Slow

### ■ Interrupt Management

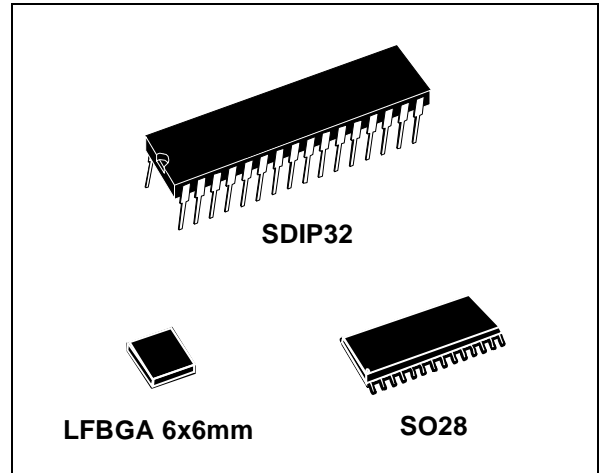
- Nested interrupt controller
- 10 interrupt vectors plus TRAP and RESET
- 22 external interrupt lines (on 2 vectors)

### ■ 22 I/O Ports

- 22 multifunctional bidirectional I/O lines
- 20 alternate function lines
- 8 high sink outputs

### ■ 4 Timers

- Main Clock Controller with Real time base and Clock-out capabilities
- Configurable watchdog timer



- Two 16-bit timers with: 2 input captures, 2 output compares, external clock input on one timer, PWM and Pulse generator modes

### ■ 3 Communications Interfaces

- SPI synchronous serial interface
- I<sup>2</sup>C multimaster interface
- SCI asynchronous serial interface (LIN compatible)

### ■ 1 Analog peripheral

- 10-bit ADC with 6 input channels

### ■ Instruction Set

- 8-bit data manipulation
- 63 basic instructions
- 17 main addressing modes
- 8 x 8 unsigned multiply instruction

### ■ Development Tools

- Full hardware/software development package

### Device Summary

Features	ST72260G1	ST72262G1	ST72262G2	ST72264G1	ST72264G2
Program memory - bytes	4K	4K	8K	4K	8K
RAM (stack) - bytes	256 (128)				
Peripherals	Watchdog timer, RTC, Two 16-bit timers, SPI	Watchdog timer, RTC Two 16-bit timers, SPI, ADC		Watchdog timer, RTC Two 16-bit timers, SPI, SCI, I <sup>2</sup> C, ADC	
Operating Supply	2.4 V to 5.5 V				
CPU Frequency	Up to 8 MHz (with oscillator up to 16 MHz) PLL 4/8 Mhz				
Operating Temperature	-40° C to +85° C				0° C to +70° C
Packages	SO28 / SDIP32				LFBGA

Rev. 1.7

---

# Table of Contents

---

<b>ST72260G, ST72262G, ST72264G</b> .....	<b>1</b>
<b>1 INTRODUCTION</b> .....	<b>5</b>
<b>2 PIN DESCRIPTION</b> .....	<b>6</b>
<b>3 REGISTER &amp; MEMORY MAP</b> .....	<b>10</b>
<b>4 FLASH PROGRAM MEMORY</b> .....	<b>14</b>
4.1 INTRODUCTION .....	14
4.2 MAIN FEATURES .....	14
4.3 PROGRAMMING MODES .....	14
4.4 ICC INTERFACE .....	15
4.5 MEMORY PROTECTION .....	16
4.6 REGISTER DESCRIPTION .....	16
<b>5 CENTRAL PROCESSING UNIT</b> .....	<b>17</b>
5.1 INTRODUCTION .....	17
5.2 MAIN FEATURES .....	17
5.3 CPU REGISTERS .....	17
<b>6 SUPPLY, RESET AND CLOCK MANAGEMENT</b> .....	<b>20</b>
6.1 PHASE LOCKED LOOP .....	20
6.2 MULTI-OSCILLATOR (MO) .....	21
6.3 RESET SEQUENCE MANAGER (RSM) .....	22
6.4 SYSTEM INTEGRITY MANAGEMENT (SI) .....	24
<b>7 INTERRUPTS</b> .....	<b>28</b>
7.1 INTRODUCTION .....	28
7.2 MASKING AND PROCESSING FLOW .....	28
7.3 INTERRUPTS AND LOW POWER MODES .....	30
7.4 CONCURRENT & NESTED MANAGEMENT .....	30
7.5 INTERRUPT REGISTER DESCRIPTION .....	31
<b>8 POWER SAVING MODES</b> .....	<b>33</b>
8.1 INTRODUCTION .....	33
8.2 SLOW MODE .....	33
8.3 WAIT MODE .....	34
8.4 ACTIVE-HALT AND HALT MODES .....	35
8.5 HALT MODE .....	36
<b>9 I/O PORTS</b> .....	<b>38</b>
9.1 INTRODUCTION .....	38
9.2 FUNCTIONAL DESCRIPTION .....	38
9.3 I/O PORT IMPLEMENTATION .....	41
9.4 UNUSED I/O PINS .....	41
9.5 LOW POWER MODES .....	41
9.6 INTERRUPTS .....	41
9.7 DEVICE-SPECIFIC I/O PORT CONFIGURATION .....	42

---

# Table of Contents

---

9.8	I/O PORT REGISTER DESCRIPTION	43
<b>10</b>	<b>MISCELLANEOUS REGISTERS</b>	<b>45</b>
10.1	I/O PORT INTERRUPT SENSITIVITY	45
10.2	I/O PORT ALTERNATE FUNCTIONS	45
10.3	MISCELLANEOUS REGISTER DESCRIPTION	46
<b>11</b>	<b>ON-CHIP PERIPHERALS</b>	<b>48</b>
11.1	WATCHDOG TIMER (WDG)	48
11.2	MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK (MCC/RTC)	53
11.3	16-BIT TIMER	55
11.4	SERIAL PERIPHERAL INTERFACE (SPI)	74
11.5	SERIAL COMMUNICATIONS INTERFACE (SCI)	85
11.6	I2C BUS INTERFACE (I2C)	99
11.7	10-BIT A/D CONVERTER (ADC)	112
<b>12</b>	<b>INSTRUCTION SET</b>	<b>116</b>
12.1	CPU ADDRESSING MODES	116
12.2	INSTRUCTION GROUPS	119
<b>13</b>	<b>ELECTRICAL CHARACTERISTICS</b>	<b>122</b>
13.1	PARAMETER CONDITIONS	122
13.2	ABSOLUTE MAXIMUM RATINGS	123
13.3	OPERATING CONDITIONS	124
13.4	SUPPLY CURRENT CHARACTERISTICS	127
13.5	CLOCK AND TIMING CHARACTERISTICS	130
13.6	MEMORY CHARACTERISTICS	135
13.7	EMC CHARACTERISTICS	136
13.8	I/O PORT PIN CHARACTERISTICS	141
13.9	CONTROL PIN CHARACTERISTICS	147
13.10	TIMER PERIPHERAL CHARACTERISTICS	148
13.11	COMMUNICATION INTERFACE CHARACTERISTICS	149
13.12	10-BIT ADC CHARACTERISTICS	152
<b>14</b>	<b>PACKAGE CHARACTERISTICS</b>	<b>154</b>
14.1	PACKAGE MECHANICAL DATA	154
14.2	THERMAL CHARACTERISTICS	155
14.3	SOLDERING AND GLUEABILITY INFORMATION	156
<b>15</b>	<b>DEVICE CONFIGURATION AND ORDERING INFORMATION</b>	<b>157</b>
15.1	OPTION BYTES	157
15.2	DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE	159
15.3	DEVELOPMENT TOOLS	161
15.4	ST7 APPLICATION NOTES	163
<b>16</b>	<b>SUMMARY OF CHANGES</b>	<b>165</b>
	<b>ERRATA SHEET</b>	<b>166</b>

<b>17 SILICON IDENTIFICATION</b> .....	<b>166</b>
<b>18 REFERENCE SPECIFICATION</b> .....	<b>166</b>
<b>19 SILICON LIMITATIONS</b> .....	<b>166</b>
19.1 EXECUTION OF BTJX INSTRUCTION .....	166
19.2 I/O PORT B AND C CONFIGURATION .....	166
19.3 16-BIT TIMER PWM MODE .....	167
19.4 SPI MULTIMASTER MODE .....	167
19.5 MINIMUM OPERATING VOLTAGE .....	167
19.6 CSS FUNCTION .....	167
19.7 INTERNAL AND EXTERNAL RC OSCILLATOR WITH LVD .....	167
19.8 EXTERNAL CLOCK WITH PLL .....	167
19.9 HALT MODE POWER CONSUMPTION WITH ADC ON .....	168
19.10 ACTIVE HALT WAKE-UP BY EXTERNAL INTERRUPT .....	168
19.11 A/D CONVERTER ACCURACY FOR FIRST CONVERSION .....	168
19.12 NEGATIVE INJECTION IMPACT ON ADC ACCURACY .....	168
19.13 ADC CONVERSION SPURIOUS RESULTS .....	168
19.14 FUNCTIONAL EMS .....	169
<b>20 DEVICE MARKING</b> .....	<b>169</b>
<b>21 ERRATA SHEET REVISION HISTORY</b> .....	<b>170</b>

To obtain the most recent version of this datasheet,  
please check at [www.st.com>products>technical literature>datasheet](http://www.st.com/products/technical_literature/datasheet)  
Please note that an errata sheet can be found at the end of this document on page 166.

## 1 INTRODUCTION

The ST72260G, ST72262G and ST72264G devices are members of the ST7 microcontroller family. They can be grouped as follows :

- ST72264G devices are designed for mid-range applications with ADC, I<sup>2</sup>C and SCI interface capabilities.
- ST72262G devices target the same range of applications but without I<sup>2</sup>C interface or SCI.
- ST72260G devices are for applications that do not need ADC, I<sup>2</sup>C peripherals or SCI.

All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

The ST72F260G, ST72F262G, and ST72F264G versions feature single-voltage FLASH memory

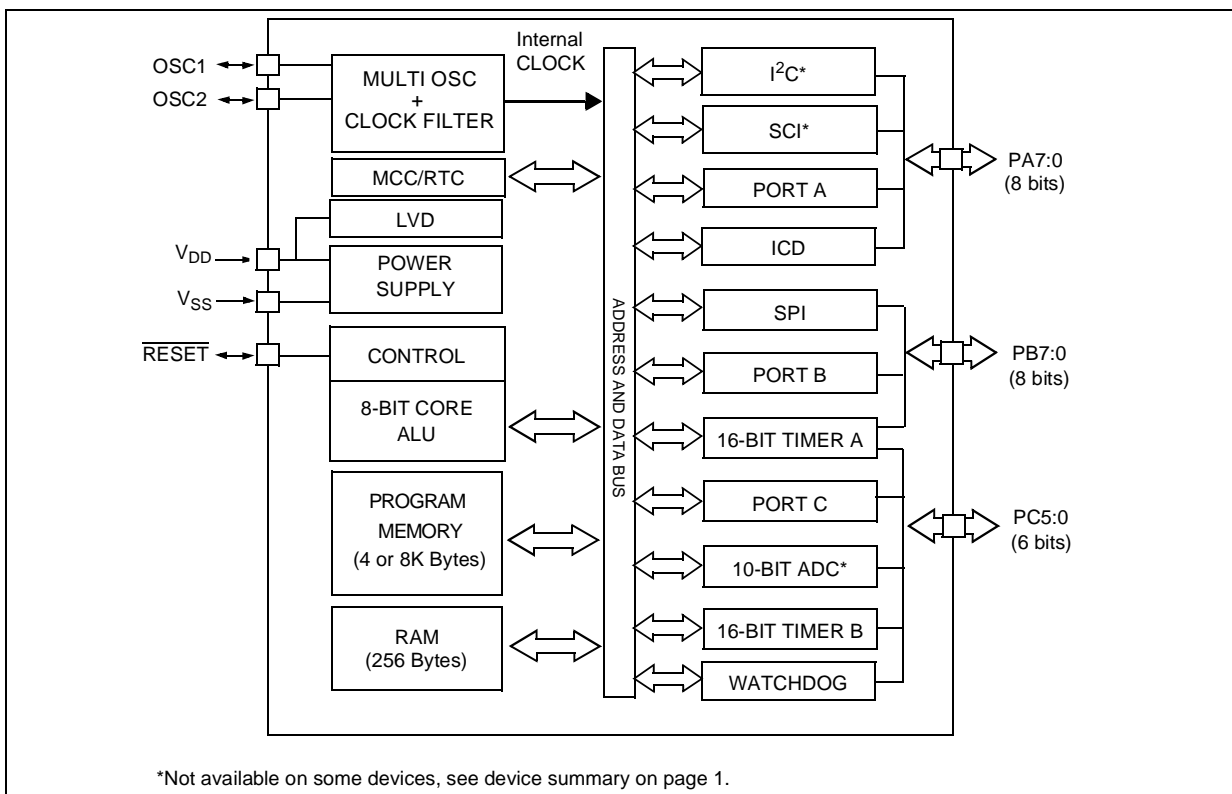
with byte-by-byte In-Circuit Programming (ICP) capabilities.

Under software control, all devices can be placed in WAIT, SLOW, Active-HALT or HALT mode, reducing power consumption when the application is in idle or stand-by state.

The enhanced instruction set and addressing modes of the ST7 offer both power and flexibility to software developers, enabling the design of highly efficient and compact application code. In addition to standard 8-bit data management, all ST7 microcontrollers feature true bit manipulation, 8x8 unsigned multiplication and indirect addressing modes.

For easy reference, all parametric data is located in Section 13 on page 122.

**Figure 1. General Block Diagram**



## 2 PIN DESCRIPTION

Figure 2. 28-Pin SO Package Pinout

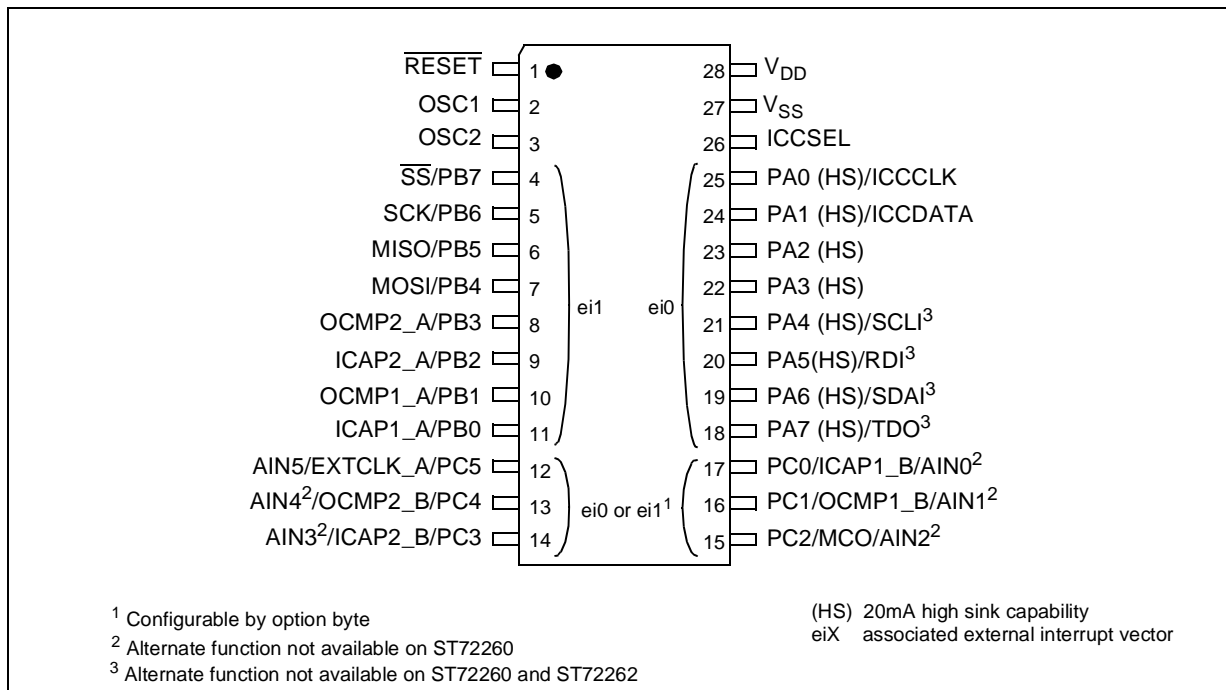


Figure 3. 32-Pin SDIP Package Pinout

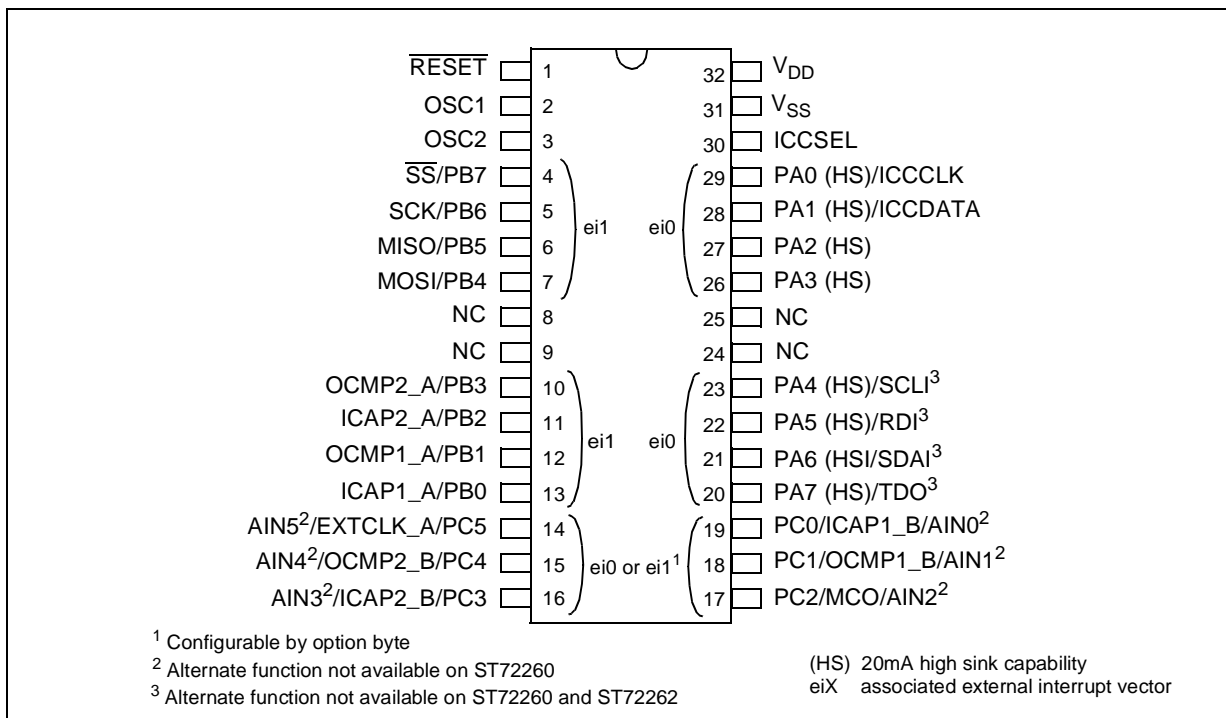
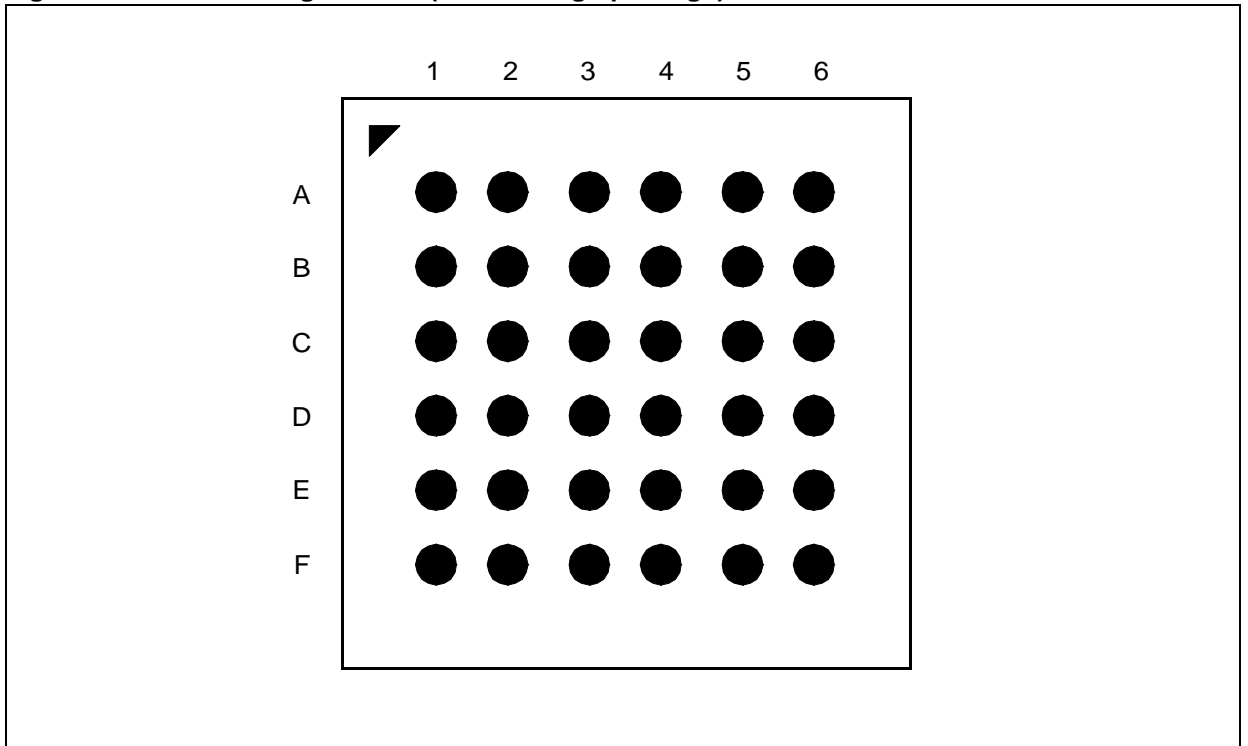


Figure 4. TFBGA Package Pinout (view through package)



**PIN DESCRIPTION** (Cont'd)

For external pin connection guidelines, refer to Section 13 "ELECTRICAL CHARACTERISTICS" on page 122.

**Legend / Abbreviations for Table 1:**

Type: I = input, O = output, S = supply

Input level: A = Dedicated analog input

In/Output level: C<sub>T</sub>= CMOS 0.3 V<sub>DD</sub>/0.7 V<sub>DD</sub> with input trigger

Output level: HS = 20 mA high sink (on N-buffer only)

Port and control configuration:

- Input: float = floating, wpu = weak pull-up, int = interrupt <sup>1)</sup>, ana = analog
- Output: OD = open drain <sup>2)</sup>, PP = push-pull

Refer to Section 9 "I/O PORTS" on page 38 for more details on the software configuration of the I/O ports.

The RESET configuration of each pin is shown in bold. This configuration is valid as long as the device is in reset state.

**Table 1. Device Pin Description**

Pin n°			Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
SDIP32	SO28	BGA			Input	Output	Input				Output			
							float	wpu	int	ana	OD	PP		
1	1	A3	<b>RESET</b>	I/O	C <sub>T</sub>		X				X		Top priority non maskable interrupt (active low)	
2	2	C4	OSC1 <sup>3)</sup>	I									External clock input or Resonator oscillator inverter input or resistor input for RC oscillator	
3	3	B3	OSC2 <sup>3)</sup>	O									Resonator oscillator inverter output or capacitor input for RC oscillator	
4	4	A2	PB7/ <b>SS</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B7</b>	SPI Slave Select (active low)	
5	5	A1	PB6/ <b>SCK</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B6</b>	SPI Serial Clock	
6	6	B1	PB5/ <b>MISO</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B5</b>	SPI Master In/ Slave Out Data	
7	7	B2	PB4/ <b>MOSI</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B4</b>	SPI Master Out / Slave In Data	
8		C1	NC	Not Connected										
9		C2	NC											
		D1	NC											
10	8	C3	PB3/ <b>OCMP2_A</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B3</b>	Timer A Output Compare 2	
11	9	D2	PB2/ <b>ICAP2_A</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B2</b>	Timer A Input Capture 2	
12	10	E1	PB1 / <b>OCMP1_A</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B1</b>	Timer A Output Compare 1	
13	11	F1	PB0 / <b>ICAP1_A</b>	I/O	C <sub>T</sub>	X	ei1			X	X	<b>Port B0</b>	Timer A Input Capture 1	
14	12	F2	PC5/ <b>EXTCLK_A/AIN5</b>	I/O	C <sub>T</sub>	X	ei0/ei1	X	X	X	X	<b>Port C5</b>	Timer A Input Clock or ADC Analog Input 5	
15	13	E2	PC4/ <b>OCMP2_B/AIN4</b>	I/O	C <sub>T</sub>	X	ei0/ei1	X	X	X	X	<b>Port C4</b>	Timer B Output Compare 2 or ADC Analog Input 4	
16	14	F3	PC3/ <b>ICAP2_B/AIN3</b>	I/O	C <sub>T</sub>	X	ei0/ei1	X	X	X	X	<b>Port C3</b>	Timer B Input Capture 2 or ADC Analog Input 3	



Pin n°			Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
SDIP32	SO28	BGA			Input	Output	Input				Output			
							float	wpu	int	ana	OD	PP		
17	15	E3	PC2/MCO/AIN2	I/O	C <sub>T</sub>	X	ei0/ei1	X	X	X	Port C2	Main clock output (f <sub>CPU</sub> ) or ADC Analog Input 2		
18	16	F4	PC1/OCMP1_B/AIN1	I/O	C <sub>T</sub>	X	ei0/ei1	X	X	X	Port C1	Timer B Output Compare 1 or ADC Analog Input 1		
19	17	D3	PC0/ICAP1_B/AIN0	I/O	C <sub>T</sub>	X	ei0/ei1	X	X	X	Port C0	Timer B Input Capture 1 or ADC Analog Input 0		
20	18	E4	PA7/TDO	I/O	C <sub>T</sub>	HS	X	ei0		X	X	Port A7	SCI output	
21	19	F5	PA6/SDAI	I/O	C <sub>T</sub>	HS	X		ei0	T		Port A6	I <sup>2</sup> C DATA	
22	20	F6	PA5 /RDI	I/O	C <sub>T</sub>	HS	X		ei0	X	X	Port A5	SCI input	
23	21	E6	PA4/SCLI	I/O	C <sub>T</sub>	HS	X		ei0	T		Port A4	I <sup>2</sup> C CLOCK	
24		E5	NC	Not Connected										
25		D6	NC											
		D5	NC											
26	22	C6	PA3	I/O	C <sub>T</sub>	HS	X	ei0		X	X	Port A3		
27	23	D4	PA2	I/O	C <sub>T</sub>	HS	X	ei0		X	X	Port A2		
		C5	NC	Not Connected										
		B6	NC											
28	24	A6	PA1/ICCDATA	I/O	C <sub>T</sub>	HS	X	ei0		X	X	Port A1	In Circuit Communication Data	
29	25	A5	PA0/ICCLK	I/O	C <sub>T</sub>	HS	X	ei0		X	X	Port A0	In Circuit Communication Clock	
30	26	B5	ICCSEL	I	C <sub>T</sub>		X						ICC mode pin, must be tied low	
31	27	A4	V <sub>SS</sub>	S									Ground	
32	28	B4	V <sub>DD</sub>	S									Main power supply	

**Notes:**

1. In the interrupt input column, "eiX" defines the associated external interrupt vector. If the weak pull-up column (wpu) is merged with the interrupt column (int), then the I/O configuration is a pull-up interrupt input, otherwise the configuration is a floating interrupt input. Port C is mapped to ei0 or ei1 by option byte.
2. In the open drain output column, "T" defines a true open drain I/O (P-Buffer and protection diode to V<sub>DD</sub> are not implemented). See Section 9 "I/O PORTS" on page 38 for more details.
3. OSC1 and OSC2 pins connect a crystal or ceramic resonator, an external RC, or an external source to the on-chip oscillator see Section 2 "PIN DESCRIPTION" on page 6 and Section 6.2 "MULTI-OSCILLATOR (MO)" on page 21 for more details.

### 3 REGISTER & MEMORY MAP

As shown in Figure 5, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 128 bytes of register location, 256 bytes of RAM and up to 8 Kbytes of user program memory. The RAM space includes up to 128 bytes for the stack from 0100h to 017Fh.

The highest address bytes contain the user reset and interrupt vectors.

The Flash memory contains two sectors (see Figure 5) mapped in the upper part of the ST7 ad-

ressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

The size of Flash Sector 0 and other device options are configurable by Option byte (refer to Section 15.1 on page 157).

**IMPORTANT:** Memory locations marked as “Reserved” must never be accessed. Accessing a reserved area can have unpredictable effects on the device.

Figure 5. Memory Map

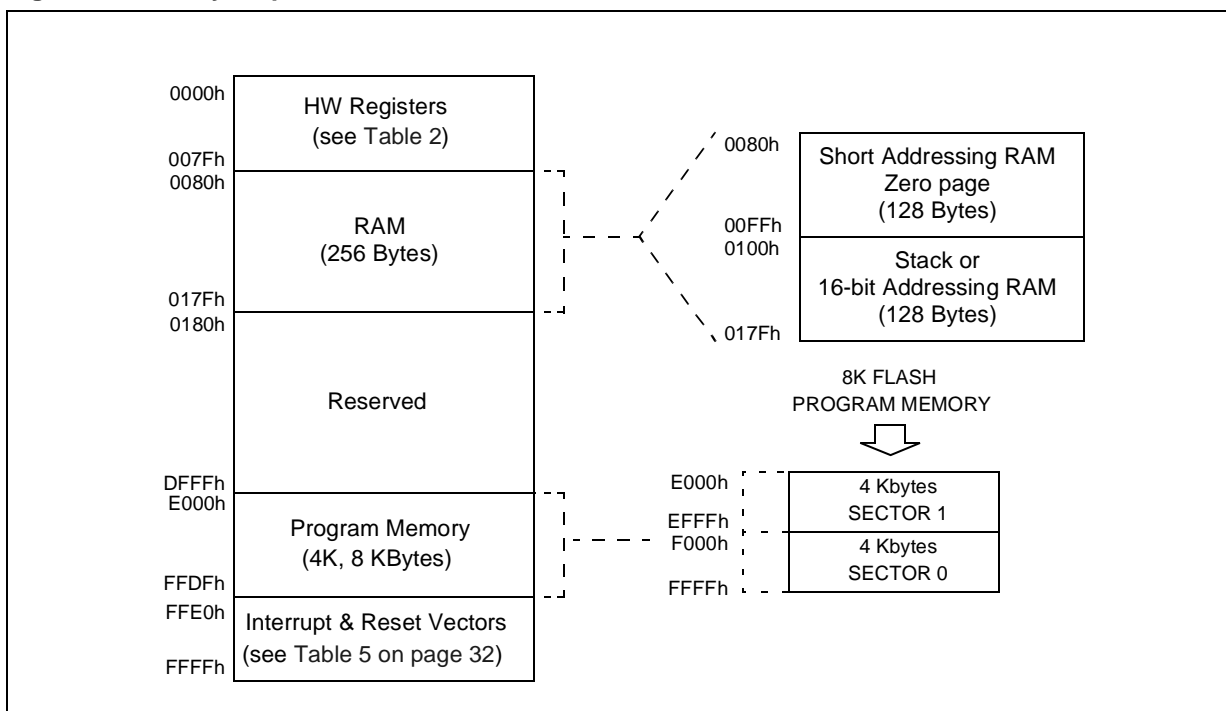


Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h 0001h 0002h	Port C	PCDR	Port C Data Register	xx000000h <sup>1)</sup>	R/W <sup>2)</sup>
		PCDDR	Port C Data Direction Register	00h	R/W <sup>2)</sup>
		PCOR	Port C Option Register	00h	R/W <sup>2)</sup>
0003h	Reserved (1 Byte)				
0004h 0005h 0006h	Port B	PBDR	Port B Data Register	00h <sup>1)</sup>	R/W
		PBDDR	Port B Data Direction Register	00h	R/W
		PBOR	Port B Option Register	00h	R/W.
0007h	Reserved (1 Byte)				
0008h 0009h 000Ah	Port A	PADR	Port A Data Register	00h <sup>1)</sup>	R/W
		PADDR	Port A Data Direction Register	00h	R/W
		PAOR	Port A Option Register	00h	R/W
000Bh to 001Bh	Reserved (17 Bytes)				
001Ch 001Dh 001Eh 001Fh	ITC	ISPR0	Interrupt software priority register0	FFh	R/W
		ISPR1	Interrupt software priority register1	FFh	R/W
		ISPR2	Interrupt software priority register2	FFh	R/W
		ISPR3	Interrupt software priority register3	FFh	R/W
0020h		MISCR1	Miscellaneous register 1	00h	R/W
0021h 0022h 0023h	SPI	SPIDR	SPI Data I/O Register	xxh	R/W
		SPICR	SPI Control Register	0xh	R/W
		SPICSR	SPI Status Register	00h	R/W
0024h	WATCHDOG	WDGCR	Watchdog Control Register	7Fh	R/W
0025h		SICSR	System Integrity Control / Status Register	000x 000x	R/W
0026h	MCC	MCCSR	Main Clock Control / Status Register	00h	R/W
0027h	Reserved (1 Byte)				
0028h 0029h 002Ah 002Bh 002Ch 002Dh 002Eh	I <sup>2</sup> C	I2CCR	I <sup>2</sup> C Control Register	00h	R/W
		I2CSR1	I <sup>2</sup> C Status Register 1	00h	Read Only
		I2CSR2	I <sup>2</sup> C Status Register 2	00h	Read Only
		I2CCCR	I <sup>2</sup> C Clock Control Register	00h	R/W
		I2COAR1	I <sup>2</sup> C Own Address Register 1	00h	R/W
		I2COAR2	I <sup>2</sup> C Own Address Register2	40h	R/W
		I2CDR	I <sup>2</sup> C Data Register	00h	R/W
002Fh 0030h	Reserved (2 Bytes)				

Address	Block	Register Label	Register Name	Reset Status	Remarks
0031h	TIMER A	TACR2	Timer A Control Register 2	00h	R/W
0032h		TACR1	Timer A Control Register 1	00h	R/W
0033h		TASCSR	Timer A Control/Status Register	xxh	R/W
0034h		TAIC1HR	Timer A Input Capture 1 High Register	xxh	Read Only
0035h		TAIC1LR	Timer A Input Capture 1 Low Register	xxh	Read Only
0036h		TAOC1HR	Timer A Output Compare 1 High Register	80h	R/W
0037h		TAOC1LR	Timer A Output Compare 1 Low Register	00h	R/W
0038h		TACHR	Timer A Counter High Register	FFh	Read Only
0039h		TACLRL	Timer A Counter Low Register	FCh	Read Only
003Ah		TAACHR	Timer A Alternate Counter High Register	FFh	Read Only
003Bh		TAACLRL	Timer A Alternate Counter Low Register	FCh	Read Only
003Ch		TAIC2HR	Timer A Input Capture 2 High Register	xxh	Read Only
003Dh		TAIC2LR	Timer A Input Capture 2 Low Register	xxh	Read Only
003Eh		TAOC2HR	Timer A Output Compare 2 High Register	80h	R/W
003Fh		TAOC2LR	Timer A Output Compare 2 Low Register	00h	R/W
0040h		MISCR2	Miscellaneous register 2	00h	R/W
0041h	TIMER B	TBCR2	Timer B Control Register 2	00h	R/W
0042h		TBCR1	Timer B Control Register 1	00h	R/W
0043h		TBSCSR	Timer B Control/Status Register	xxh	R/W
0044h		TBIC1HR	Timer B Input Capture 1 High Register	xxh	Read Only
0045h		TBIC1LR	Timer B Input Capture 1 Low Register	xxh	Read Only
0046h		TBOC1HR	Timer B Output Compare 1 High Register	80h	R/W
0047h		TBOC1LR	Timer B Output Compare 1 Low Register	00h	R/W
0048h		TBCHR	Timer B Counter High Register	FFh	Read Only
0049h		TBCLRL	Timer B Counter Low Register	FCh	Read Only
004Ah		TBACHR	Timer B Alternate Counter High Register	FFh	Read Only
004Bh		TBACLRL	Timer B Alternate Counter Low Register	FCh	Read Only
004Ch		TBIC2HR	Timer B Input Capture 2 High Register	xxh	Read Only
004Dh		TBIC2LR	Timer B Input Capture 2 Low Register	xxh	Read Only
004Eh		TBOC2HR	Timer B Output Compare 2 High Register	80h	R/W
004Fh		TBOC2LR	Timer B Output Compare 2 Low Register	00h	R/W
0050h	SCI	SCISR	SCI Status Register	C0h	Read Only
0051h		SCIDR	SCI Data Register	xxh	R/W
0052h		SCIBRR	SCI Baud Rate Register	00h	R/W
0053h		SCICR1	SCI Control Register1	x000 0000h	R/W
0054h		SCICR2	SCI Control Register2	00h	R/W
0055h		SCIERPR	SCI Extended Receive Prescaler Register	00h	R/W
0056h		SCIETPR	SCI Extended Transmit Prescaler Register	00h	R/W
0057h to 006Eh	Reserved (24 Bytes)				
006Fh	ADC	ADCDRL	Data Register Low <sup>3)</sup>	00h	Read Only
0070h		ADCDRH	Data Register High <sup>3)</sup>	00h	Read Only
0071h		ADCCSR	Control/Status Register	00h	R/W
0072h	FLASH	FCSR	Flash Control Register	00h	R/W
0073h to 007Fh	Reserved (13 Bytes)				

**Legend:** x=Undefined, R/W=Read/Write

**Notes:**

1. The contents of the I/O port DR registers are readable only in output configuration. In input configuration, the values of the I/O pins are returned instead of the DR register contents.
2. The bits associated with unavailable pins must always keep their reset value.
3. For compatibility with the ST72C254, the ADCDRL and ADCDRH data registers are located with the LSB on the lower address (6Fh) and the MSB on the higher address (70h). As this scheme is not little Endian, the ADC data registers cannot be treated by C programs as an integer, but have to be treated as two char registers.

## 4 FLASH PROGRAM MEMORY

### 4.1 Introduction

The ST7 single voltage extended Flash (XFlash) is a non-volatile memory that can be electrically erased and programmed either on a byte-by-byte basis or up to 32 bytes in parallel.

The XFlash devices can be programmed off-board (plugged in a programming tool) or on-board using In-Circuit Programming or In-Application Programming.

The array matrix organisation allows each sector to be erased and reprogrammed without affecting other sectors.

### 4.2 Main Features

- ICP (In-Circuit Programming)
- IAP (In-Application Programming)
- ICT (In-Circuit Testing) for downloading and executing user application test patterns in RAM
- Sector 0 size configurable by option byte
- Read-out and write protection against piracy

### 4.3 PROGRAMMING MODES

The ST7 can be programmed in three different ways:

- Insertion in a programming tool. In this mode, FLASH sectors 0 and 1 and option byte row can be programmed or erased.
- In-Circuit Programming. In this mode, FLASH sectors 0 and 1 and option byte row can be programmed or erased without removing the device from the application board.
- In-Application Programming. In this mode, sector 1 can be programmed or erased without removing the device from the application

board and while the application is running.

#### 4.3.1 In-Circuit Programming (ICP)

ICP uses a protocol called ICC (In-Circuit Communication) which allows an ST7 plugged on a printed circuit board (PCB) to communicate with an external programming device connected via cable. ICP is performed in three steps:

Switch the ST7 to ICC mode (In-Circuit Communications). This is done by driving a specific signal sequence on the ICCCLK/DATA pins while the RESET pin is pulled low. When the ST7 enters ICC mode, it fetches a specific RESET vector which points to the ST7 System Memory containing the ICC protocol routine. This routine enables the ST7 to receive bytes from the ICC interface.

- Download ICP Driver code in RAM from the ICCDATA pin
- Execute ICP Driver code in RAM to program the FLASH memory

Depending on the ICP Driver code downloaded in RAM, FLASH memory programming can be fully customized (number of bytes to program, program locations, or selection of the serial communication interface for downloading).

#### 4.3.2 In Application Programming (IAP)

This mode uses an IAP Driver program previously programmed in Sector 0 by the user (in ICP mode).

This mode is fully controlled by user software. This allows it to be adapted to the user application, (user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored etc.)

IAP mode can be used to program any memory areas except Sector 0, which is write/erase protected to allow recovery in case errors occur during the programming operation.

## FLASH PROGRAM MEMORY (Cont'd)

### 4.4 ICC interface

ICP needs a minimum of 4 and up to 7 pins to be connected to the programming tool. These pins are:

- $\overline{\text{RESET}}$ : device reset
- $V_{SS}$ : device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input serial data pin
- ICCSEL: ICC selection (not required on devices without ICCSEL pin)
- OSC1: main clock input for external source (not required on devices without OSC1/OSC2 pins)
- $V_{DD}$ : application board power supply (optional, see Note 3)

#### Notes:

1. If the ICCCLK or ICCDATA pins are only used as outputs in the application, no signal isolation is necessary. As soon as the Programming Tool is plugged to the board, even if an ICC session is not in progress, the ICCCLK and ICCDATA pins are not available for the application. If they are used as inputs by the application, isolation such as a serial resistor has to be implemented in case another device forces the signal. Refer to the Programming

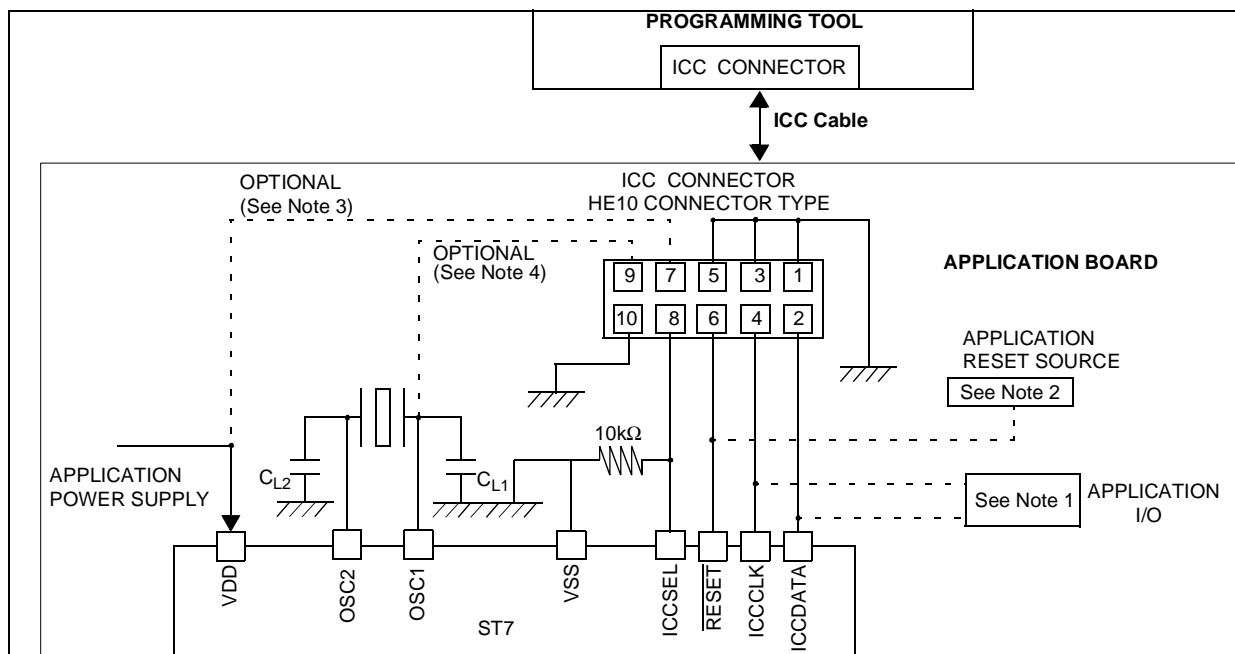
Tool documentation for recommended resistor values.

2. During the ICP session, the programming tool must control the  $\overline{\text{RESET}}$  pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor < 1K). A schottky diode can be used to isolate the application RESET circuit in this case. When using a classical RC network with  $R > 1K$  or a reset management IC with open drain output and pull-up resistor > 1K, no additional components are needed. In all cases the user must ensure that no external reset is generated by the application during the ICC session.

3. The use of Pin 7 of the ICC connector depends on the Programming Tool architecture. This pin must be connected when using most ST Programming Tools (it is used to monitor the application power supply). Please refer to the Programming Tool manual.

4. Pin 9 has to be connected to the OSC1 pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte. ST7 devices with multi-oscillator capability need to have OSC2 grounded in this case.

Figure 6. Typical ICC Interface



**FLASH PROGRAM MEMORY (Cont'd)**

**4.5 Memory Protection**

There are two different types of memory protection: Read Out Protection and Write/Erase Protection which can be applied individually.

**4.5.1 Read out Protection**

Read out protection, when selected, makes it impossible to extract the memory content from the microcontroller, thus preventing piracy.

In flash devices, this protection is removed by re-programming the option. In this case the program memory is automatically erased and the device can be reprogrammed.

Read-out protection selection depends on the device type:

- In Flash devices it is enabled and removed through the FMP\_R bit in the option byte.
- In ROM devices it is enabled by mask option specified in the Option List.

**4.5.2 Flash Write/Erase Protection**

Write/erase protection, when set, makes it impossible to both overwrite and erase program memory. Its purpose is to provide advanced security to applications and prevent any change being made to the memory content.

**Warning:** Once set, Write/erase protection can never be removed. A write-protected flash device is no longer reprogrammable.

Write/erase protection is enabled through the FMP\_W bit in the option byte.

**4.6 Register Description**

**FLASH CONTROL/STATUS REGISTER (FCSR)**

Read/Write

Reset Value: 000 0000 (00h)

1st RASS Key: 0101 0110 (56h)

2nd RASS Key: 1010 1110 (AEh)

					7				0
0	0	0	0	0	OPT	LAT	PGM		

**Note:** This register is reserved for programming using ICP, IAP or other programming methods. It controls the XFlash programming and erasing operations. For details on XFlash programming, refer to the ST7 Flash Programming Reference Manual.

When an EPB or another programming tool is used (in socket or ICP mode), the RASS keys are sent automatically.



## 5 CENTRAL PROCESSING UNIT

### 5.1 INTRODUCTION

This CPU has a full 8-bit architecture and contains six internal registers allowing efficient 8-bit data manipulation.

### 5.2 MAIN FEATURES

- Enable executing 63 basic instructions
- Fast 8-bit by 8-bit multiply
- 17 main addressing modes (with indirect addressing mode)
- Two 8-bit index registers
- 16-bit stack pointer
- Low power HALT and WAIT modes
- Priority maskable hardware interrupts
- Non-maskable software/hardware interrupts

### 5.3 CPU REGISTERS

The 6 CPU registers shown in Figure 7 are not present in the memory mapping and are accessed by specific instructions.

#### Accumulator (A)

The Accumulator is an 8-bit general purpose register used to hold operands and the results of the arithmetic and logic calculations and to manipulate data.

#### Index Registers (X and Y)

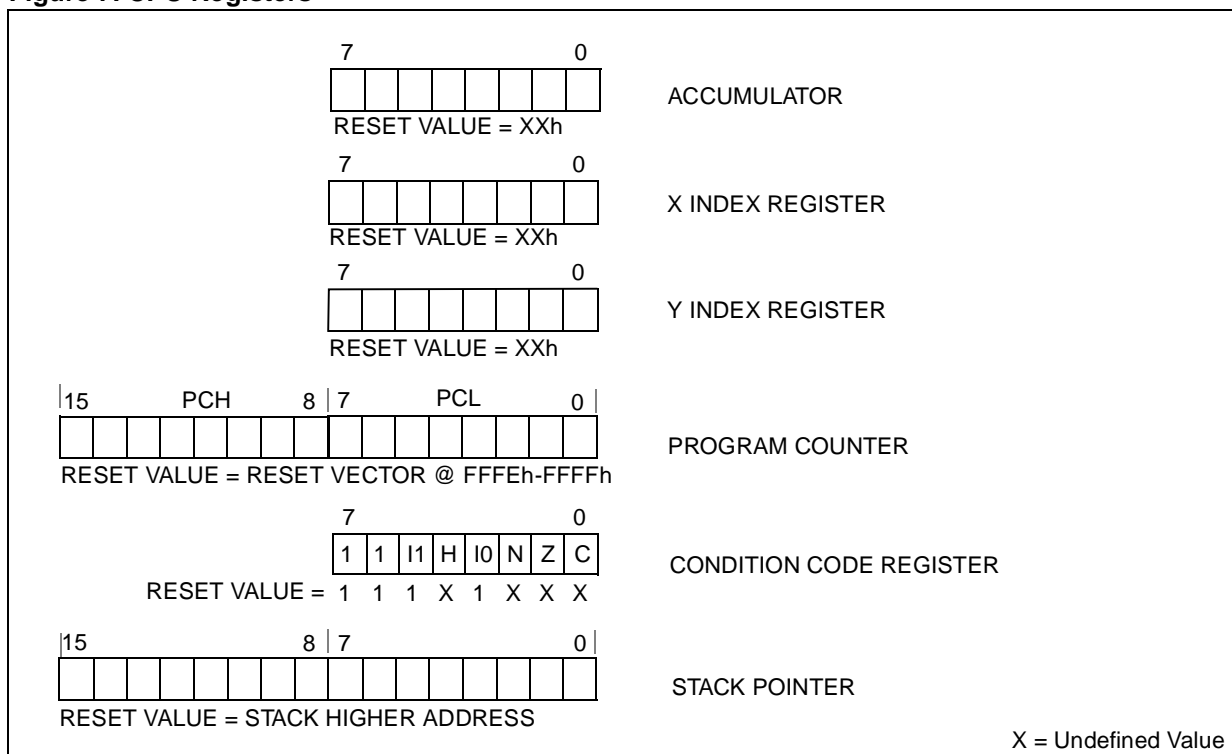
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

The Y register is not affected by the interrupt automatic procedures.

#### Program Counter (PC)

The program counter is a 16-bit register containing the address of the next instruction to be executed by the CPU. It is made of two 8-bit registers PCL (Program Counter Low which is the LSB) and PCH (Program Counter High which is the MSB).

Figure 7. CPU Registers



**CENTRAL PROCESSING UNIT (Cont'd)**

**Condition Code Register (CC)**

Read/Write

Reset Value: 111x1xxx

7							0
1	1	I1	H	I0	N	Z	C

The 8-bit Condition Code register contains the interrupt masks and four flags representative of the result of the instruction just executed. This register can also be handled by the PUSH and POP instructions.

These bits can be individually tested and/or controlled by specific instructions.

**Arithmetic Management Bits**

Bit 4 = **H** *Half carry*.

This bit is set by hardware when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instructions. It is reset by hardware during the same instructions.

- 0: No half carry has occurred.
- 1: A half carry has occurred.

This bit is tested using the JRH or JRNH instruction. The H bit is useful in BCD arithmetic subroutines.

Bit 2 = **N** *Negative*.

This bit is set and cleared by hardware. It is representative of the result sign of the last arithmetic, logical or data manipulation. It's a copy of the result 7<sup>th</sup> bit.

- 0: The result of the last operation is positive or null.
- 1: The result of the last operation is negative (i.e. the most significant bit is a logic 1).

This bit is accessed by the JRMI and JRPL instructions.

Bit 1 = **Z** *Zero*.

This bit is set and cleared by hardware. This bit indicates that the result of the last arithmetic, logical or data manipulation is zero.

- 0: The result of the last operation is different from zero.
- 1: The result of the last operation is zero.

This bit is accessed by the JREQ and JRNE test instructions.

Bit 0 = **C** *Carry/borrow*.

This bit is set and cleared by hardware and software. It indicates an overflow or an underflow has occurred during the last arithmetic operation.

- 0: No overflow or underflow has occurred.
- 1: An overflow or underflow has occurred.

This bit is driven by the SCF and RCF instructions and tested by the JRC and JRNC instructions. It is also affected by the "bit test and branch", shift and rotate instructions.

**Interrupt Management Bits**

Bit 5,3 = **I1, I0** *Interrupt*

The combination of the I1 and I0 bits gives the current interrupt software priority.

Interrupt Software Priority	I1	I0
Level 0 (main)	1	0
Level 1	0	1
Level 2	0	0
Level 3 (= interrupt disable)	1	1

These two bits are set/cleared by hardware when entering in interrupt. The loaded value is given by the corresponding bits in the interrupt software priority registers (IxSPR). They can be also set/cleared by software with the RIM, SIM, IRET, HALT, WFI and PUSH/POP instructions.

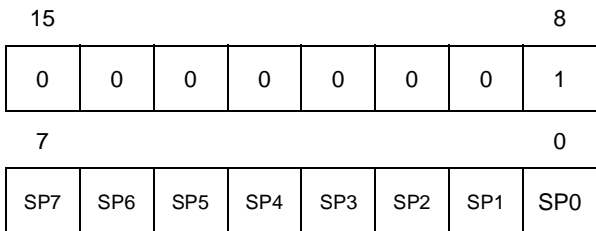
See the interrupt management chapter for more details.

**CENTRAL PROCESSING UNIT (Cont'd)**

**Stack Pointer (SP)**

Read/Write

Reset Value: 01 7Fh



The Stack Pointer is a 16-bit register which is always pointing to the next free location in the stack. It is then decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see Figure 8).

Since the stack is 128 bytes deep, the 8 most significant bits are forced by hardware. Following an MCU Reset, or after a Reset Stack Pointer instruction (RSP), the Stack Pointer contains its reset value (the SP7 to SP0 bits are set) which is the stack higher address.

The least significant byte of the Stack Pointer (called S) can be directly accessed by a LD instruction.

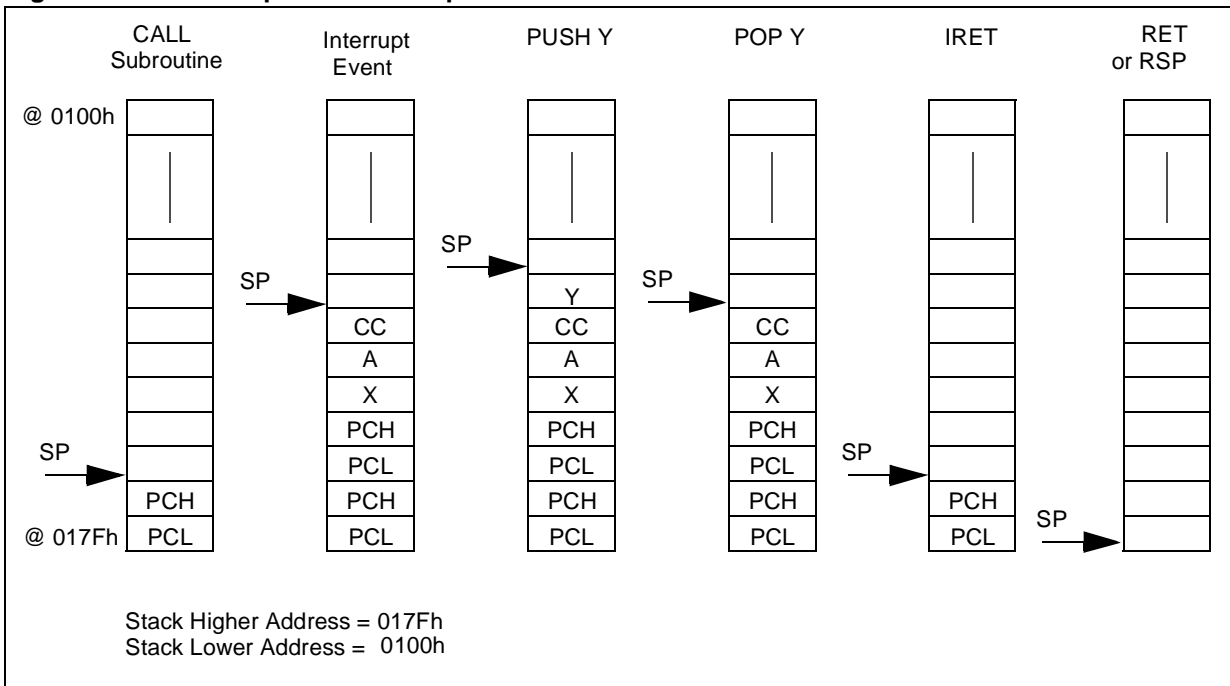
**Note:** When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 8

- When an interrupt is received, the SP is decremented and the context is pushed on the stack.
- On return from interrupt, the SP is incremented and the context is popped from the stack.

A subroutine call occupies two locations and an interrupt five locations in the stack area.

**Figure 8. Stack Manipulation Example**



## 6 SUPPLY, RESET AND CLOCK MANAGEMENT

The device includes a range of utility features for securing the application in critical situations (for example in case of a power brown-out), and reducing the number of external components. An overview is shown in Figure 10.

For more details, refer to dedicated parametric section.

### Main Features

- Optional PLL for multiplying the frequency by 2 (not to be used with internal RC oscillator)
- Reset Sequence Manager (RSM)
- Multi-Oscillator Clock Management (MO)
  - 4 Crystal/Ceramic resonator oscillators
  - 1 Internal RC oscillator
- System Integrity Management (SI)
  - Main supply Low Voltage Detector (LVD)
  - Auxiliary Voltage Detector (AVD) with interrupt capability for monitoring the main supply
  - Clock Security System (CSS) with Clock Filter and Backup Safe Oscillator (enabled by option byte)

### 6.1 PHASE LOCKED LOOP

If the clock frequency input to the PLL is in the 2 to 4 MHz range, the PLL can be used to multiply the frequency by two to obtain an  $f_{OSC2}$  of 4 to 8 MHz.

The PLL is enabled by option byte. If the PLL is disabled, then  $f_{OSC2} = f_{OSC}/2$ .

**Caution:** The PLL is not recommended for applications where timing accuracy is required. See “PLL Characteristics” on page 134.

Figure 9. PLL Block Diagram

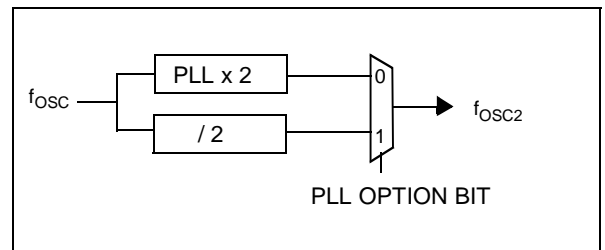
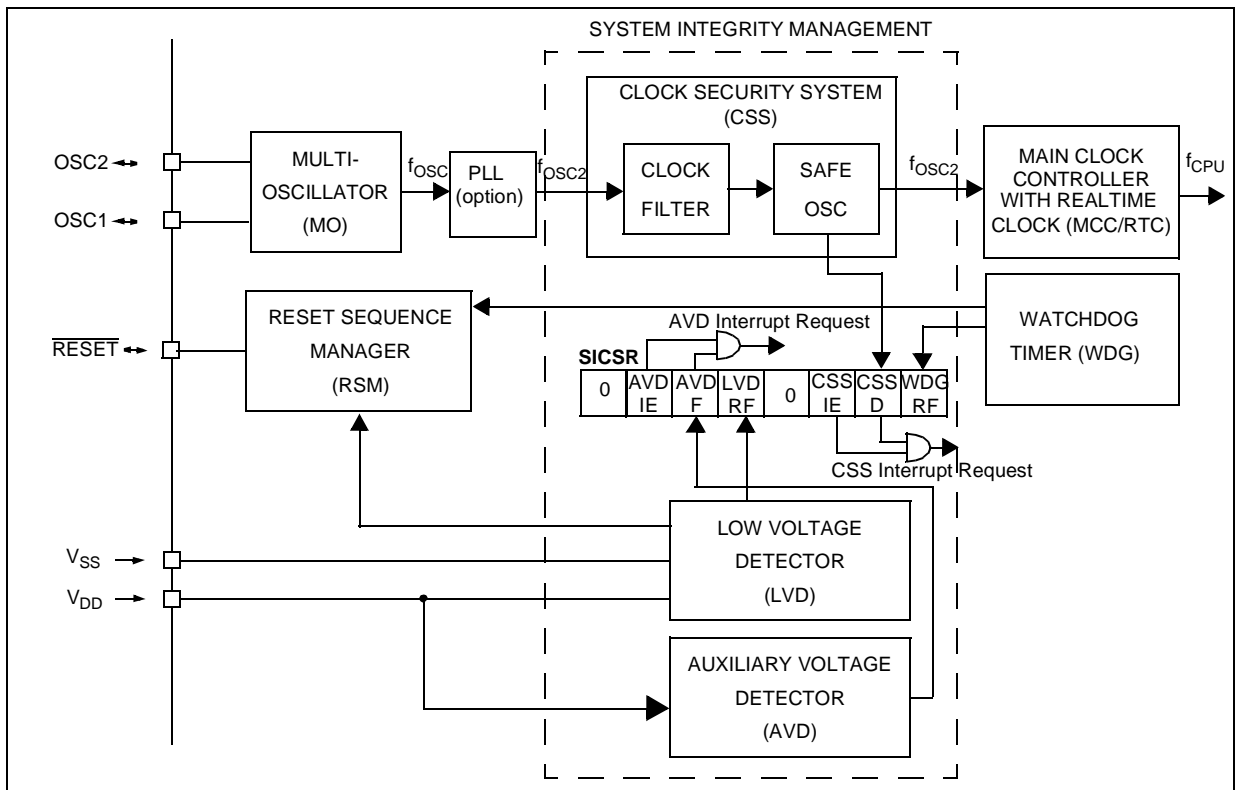


Figure 10. Clock, Reset and Supply Block Diagram



## 6.2 MULTI-OSCILLATOR (MO)

The main clock of the ST7 can be generated by four different source types coming from the multi-oscillator block:

- an external source
- 5 crystal or ceramic resonator oscillators
- an internal high frequency RC oscillator

Each oscillator is optimized for a given frequency range in terms of consumption and is selectable through the option byte. The associated hardware configurations are shown in Table 3. Refer to the electrical characteristics section for more details.

**Caution:** The OSC1 and/or OSC2 pins must not be left unconnected. For the purposes of Failure Mode and Effects Analysis, it should be noted that if the OSC1 and/or OSC2 pins are left unconnected, the ST7 main oscillator may start and, in this configuration, could generate an  $f_{OSC}$  clock frequency in excess of the allowed maximum ( $>16\text{MHz.}$ ), putting the ST7 in an unsafe/undefined state. The product behaviour must therefore be considered undefined when the OSC pins are left unconnected.

### External Clock Source

In this external clock mode, a clock signal (square, sinus or triangle) with  $\sim 50\%$  duty cycle has to drive the OSC1 pin while the OSC2 pin is tied to ground.

### Crystal/Ceramic Oscillators

This family of oscillators has the advantage of producing a very accurate rate on the main clock of the ST7. The selection within a list of 5 oscillators with different frequency ranges has to be done by option byte in order to reduce consumption (refer to Section 15.1 on page 157 for more details on the frequency ranges). In this mode of the multi-oscillator, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. The loading capacitance values must be adjusted according to the selected oscillator.

These oscillators are not stopped during the RESET phase to avoid losing time in the oscillator start-up phase.

### Internal RC Oscillator

This oscillator allows a low cost solution for the main clock of the ST7 using only an internal resistor and capacitor. Internal RC oscillator mode has the drawback of a lower frequency accuracy and should not be used in applications that require accurate timing.

In this mode, the two oscillator pins have to be tied to ground.

**Table 3. ST7 Clock Sources**

	Hardware Configuration
External Clock	
Crystal/Ceramic Resonators	
Internal RC Oscillator	

### 6.3 RESET SEQUENCE MANAGER (RSM)

#### 6.3.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 12:

- External  $\overline{\text{RESET}}$  source pulse
- Internal LVD RESET (Low Voltage Detection)
- Internal WATCHDOG RESET

These sources act on the  $\overline{\text{RESET}}$  pin and it is always kept low during the delay phase.

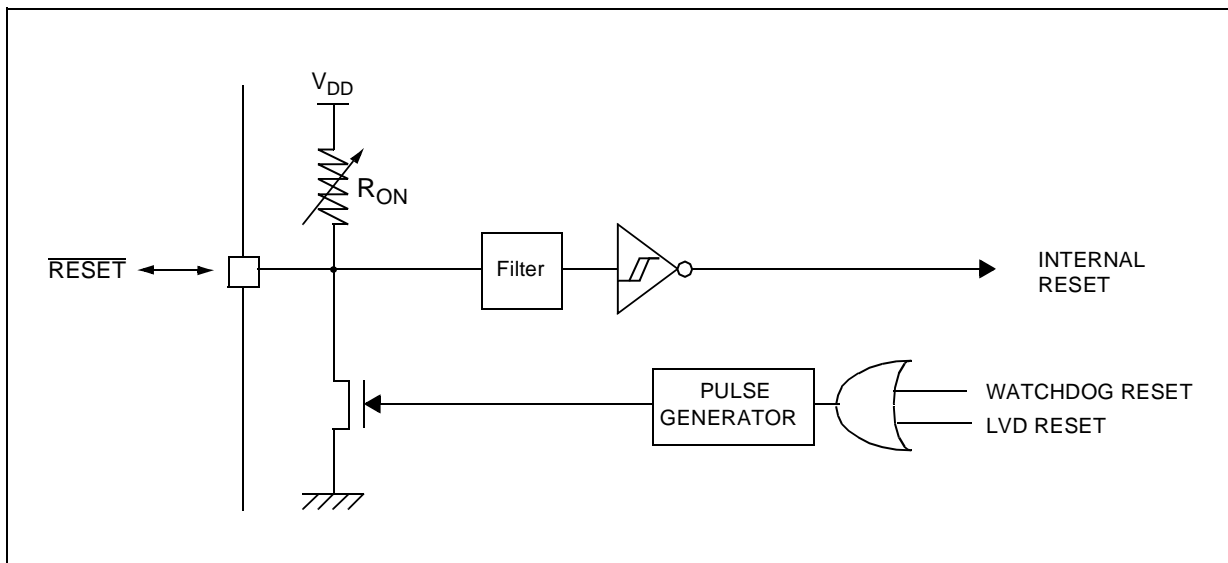
The RESET service routine vector is fixed at addresses FFFEh-FFFFh in the ST7 memory map.

The basic RESET sequence consists of 3 phases as shown in Figure 11:

- Active Phase depending on the RESET source
- 4096 CPU clock cycle delay (selected by option byte)
- RESET vector fetch

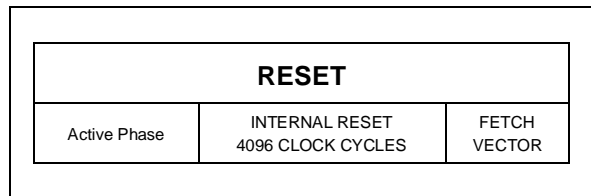
The 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay should be selected by option byte to correspond to the stabilization time of the external oscillator used in the application.

Figure 12. Reset Block Diagram



The RESET vector fetch phase duration is 2 clock cycles.

Figure 11. RESET Sequence Phases



#### 6.3.2 Asynchronous External $\overline{\text{RESET}}$ pin

The  $\overline{\text{RESET}}$  pin is both an input and an open-drain output with integrated  $R_{ON}$  weak pull-up resistor. This pull-up has no fixed value but varies in accordance with the input voltage. It can be pulled low by external circuitry to reset the device. See Electrical Characteristic section for more details.

A RESET signal originating from an external source must have a duration of at least  $t_{h(RSTL)in}$  in order to be recognized (see Figure 13). This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

## RESET SEQUENCE MANAGER (Cont'd)

The  $\overline{\text{RESET}}$  pin is an asynchronous signal which plays a major role in EMS performance. In a noisy environment, it is recommended to follow the guidelines mentioned in the electrical characteristics section.

### 6.3.3 External Power-On RESET

If the LVD is disabled by option byte, to start up the microcontroller correctly, the user must ensure by means of an external reset circuit that the reset signal is held low until  $V_{\text{DD}}$  is over the minimum level specified for the selected  $f_{\text{OSC}}$  frequency.

A proper reset signal for a slow rising  $V_{\text{DD}}$  supply can generally be provided by an external RC network connected to the  $\overline{\text{RESET}}$  pin.

### 6.3.4 Internal Low Voltage Detector (LVD) RESET

Two different RESET sequences caused by the internal LVD circuitry can be distinguished:

- Power-On RESET
- Voltage Drop RESET

The device  $\overline{\text{RESET}}$  pin acts as an output that is pulled low when  $V_{\text{DD}} < V_{\text{IT}+}$  (rising edge) or  $V_{\text{DD}} < V_{\text{IT}-}$  (falling edge) as shown in Figure 13.

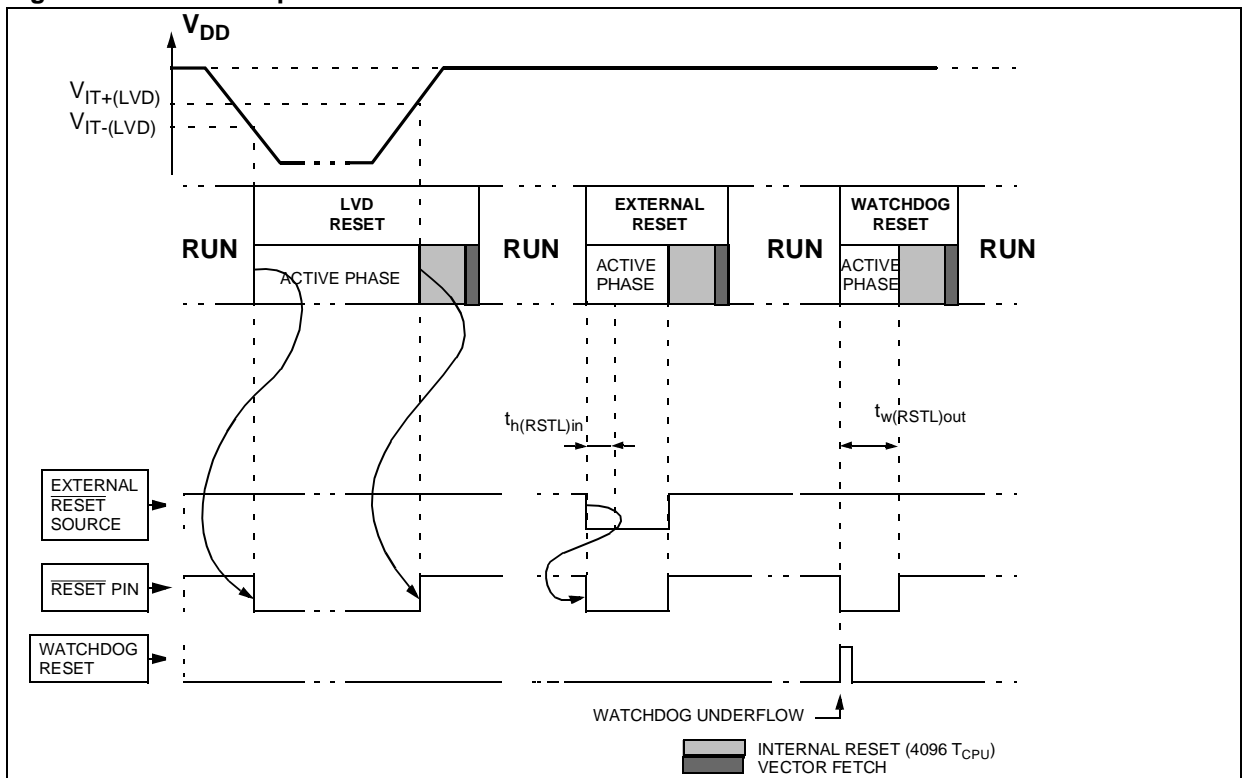
The LVD filters spikes on  $V_{\text{DD}}$  larger than  $t_{\text{g}}(V_{\text{DD}})$  to avoid parasitic resets.

### 6.3.5 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter overflow is shown in Figure 13.

Starting from the Watchdog counter underflow, the device  $\overline{\text{RESET}}$  pin acts as an output that is pulled low during at least  $t_{\text{w}}(\text{RSTL})_{\text{out}}$ .

Figure 13. RESET Sequences



### 6.4 SYSTEM INTEGRITY MANAGEMENT (SI)

The System Integrity Management block contains group the Low voltage Detector (LVD), Auxiliary Voltage Detector (AVD) and Clock Security System (CSS) functions. It is managed by the SICSR register.

#### 6.4.1 Low Voltage Detector (LVD)

The Low Voltage Detector function (LVD) generates a static reset when the  $V_{DD}$  supply voltage is below a  $V_{IT-}$  reference value. This means that it secures the power-up as well as the power-down keeping the ST7 in reset.

The  $V_{IT-}$  reference value for a voltage drop is lower than the  $V_{IT+}$  reference value for power-on in order to avoid a parasitic reset when the MCU starts running and sinks current on the supply (hysteresis).

The LVD Reset circuitry generates a reset when  $V_{DD}$  is below:

- $V_{IT+}$  when  $V_{DD}$  is rising
- $V_{IT-}$  when  $V_{DD}$  is falling

The LVD function is illustrated in Figure 14.

The voltage threshold can be configured by option byte to be low, medium or high.

Provided the minimum  $V_{DD}$  value (guaranteed for the oscillator frequency) is above  $V_{IT-}$ , the MCU can only be in two modes:

- under full software control
- in static safe reset

In these conditions, secure operation is always ensured for the application without the need for external reset hardware.

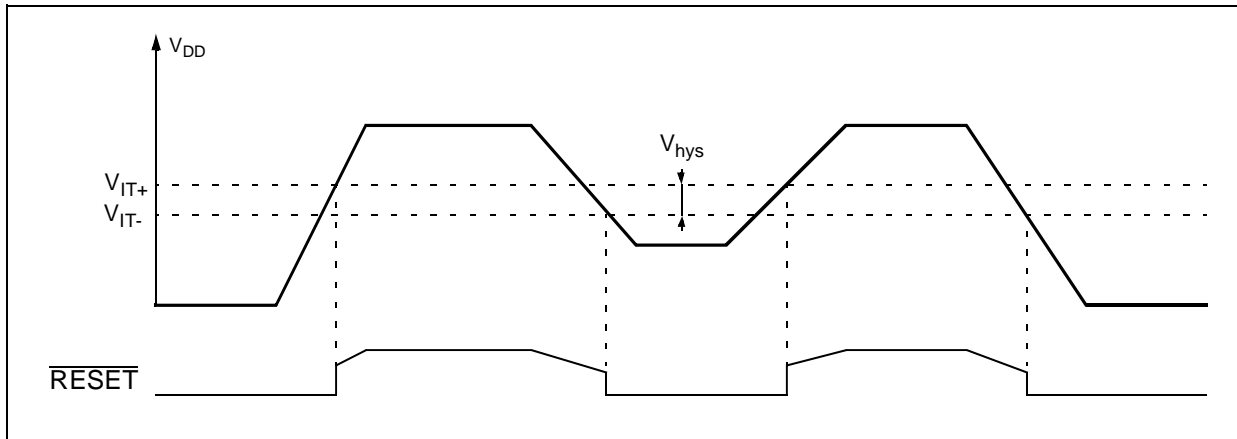
During a Low Voltage Detector Reset, the  $\overline{RESET}$  pin is held low, thus permitting the MCU to reset other devices.

**Notes:**

The LVD allows the device to be used without any external RESET circuitry.

The LVD is an optional function which can be selected by option byte.

Figure 14. Low Voltage Detector vs Reset





## SYSTEM INTEGRITY MANAGEMENT (Cont'd)

### 6.4.2 Auxiliary Voltage Detector (AVD)

The Voltage Detector function (AVD) is based on an analog comparison between a  $V_{IT-}$  and  $V_{IT+}$  reference value and the  $V_{DD}$  main supply. The  $V_{IT-}$  reference value for falling voltage is lower than the  $V_{IT+}$  reference value for rising voltage in order to avoid parasitic detection (hysteresis).

The output of the AVD comparator is directly readable by the application software through a real time status bit (VDF) in the SICSR register. This bit is read only.

**Caution:** The AVD functions only if the LVD is enabled through the option byte.

#### 6.4.2.1 Monitoring the $V_{DD}$ Main Supply

The AVD voltage threshold value is relative to the selected LVD threshold configured by option byte (see Section 15.1 on page 157).

If the AVD interrupt is enabled, an interrupt is generated when the voltage crosses the  $V_{IT+(AVD)}$  or  $V_{IT-(AVD)}$  threshold (AVDF bit toggles).

In the case of a drop in voltage, the AVD interrupt acts as an early warning, allowing software to shut down safely before the LVD resets the microcontroller. See Figure 15.

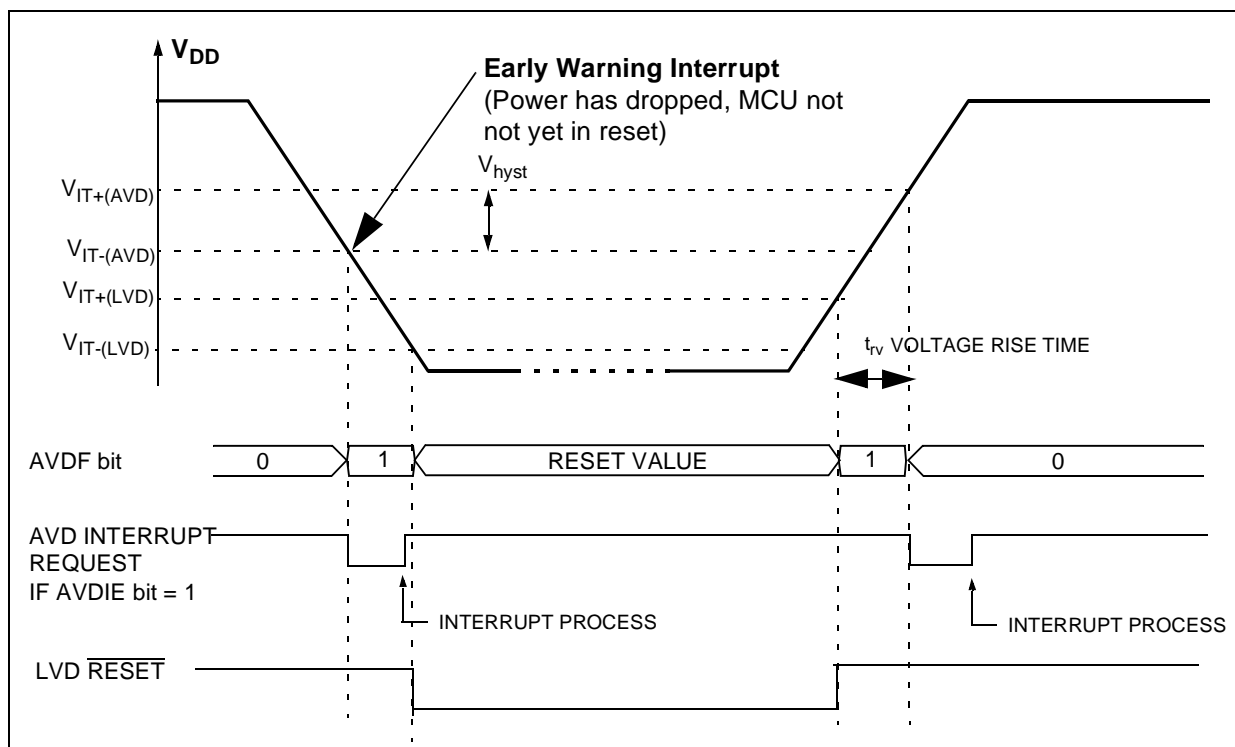
The interrupt on the rising edge is used to inform the application that the  $V_{DD}$  warning state is over.

If the voltage rise time  $t_{rv}$  is less than 256 or 4096 CPU cycles (depending on the reset delay selected by option byte), no AVD interrupt will be generated when  $V_{IT+(AVD)}$  is reached.

If  $t_{rv}$  is greater than 256 or 4096 cycles then:

- If the AVD interrupt is enabled before the  $V_{IT+(AVD)}$  threshold is reached, then 2 AVD interrupts will be received: the first when the AVDIE bit is set, and the second when the threshold is reached.
- If the AVD interrupt is enabled after the  $V_{IT+(AVD)}$  threshold is reached then only one AVD interrupt will occur.

Figure 15. Using the AVD to Monitor  $V_{DD}$



**SYSTEM INTEGRITY MANAGEMENT (Cont'd)**

**6.4.3 Clock Security System (CSS)**

The Clock Security System (CSS) protects the ST7 against breakdowns, spikes and overfrequencies occurring on the main clock source ( $f_{OSC}$ ). It is based on a clock filter and a clock detection control with an internal safe oscillator ( $f_{SFOSC}$ ).

**6.4.3.1 Clock Filter Control**

The PLL has an integrated glitch filtering capability making it possible to protect the internal clock from overfrequencies created by individual spikes. This feature is available only when the PLL is enabled. If glitches occur on  $f_{OSC}$  (for example, due to loose connection or noise), the CSS filters these automatically, so the internal CPU frequency ( $f_{CPU}$ ) continues deliver a glitch-free signal (see Figure 16).

**6.4.3.2 Clock detection Control**

If the clock signal disappears (due to a broken or disconnected resonator...), the safe oscillator delivers a low frequency clock signal ( $f_{SFOSC}$ ) which allows the ST7 to perform some rescue operations.

Automatically, the ST7 clock source switches back from the safe oscillator ( $f_{SFOSC}$ ) if the main clock source ( $f_{OSC}$ ) recovers.

When the internal clock ( $f_{CPU}$ ) is driven by the safe oscillator ( $f_{SFOSC}$ ), the application software is notified by hardware setting the CSSD bit in the SICSR register. An interrupt can be generated if the

CSSIE bit has been previously set.

These two bits are described in the SICSR register description.

**6.4.4 Low Power Modes**

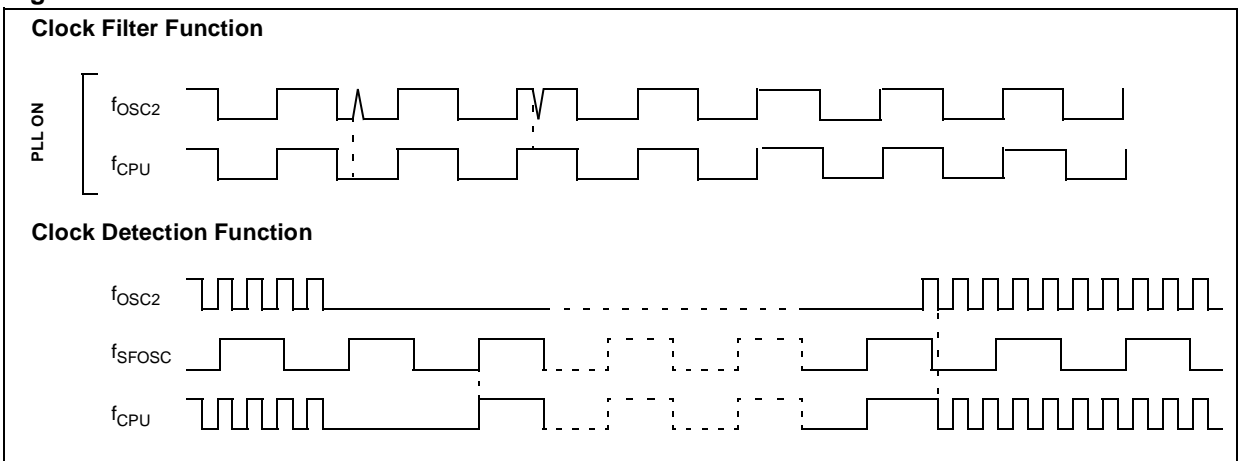
Mode	Description
WAIT	No effect on SI. CSS and AVD interrupts cause the device to exit from Wait mode.
HALT	The SICSR register is frozen. The CSS (including the safe oscillator) is disabled until HALT mode is exited. The previous CSS configuration resumes when the MCU is woken up by an interrupt with "exit from HALT mode" capability or from the counter reset value when the MCU is woken up by a RESET.

**6.4.4.1 Interrupts**

The CSS or AVD interrupt events generate an interrupt if the corresponding Enable Control Bit (CSSIE or AVDIE) is set and the interrupt mask in the CC register is reset (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
CSS event detection (safe oscillator activated as main clock)	CSSD	CSSIE	Yes	No
AVD event	AVDF	AVDIE	Yes	No

**Figure 16. Clock Filter Function**



**SYSTEM INTEGRITY MANAGEMENT (Cont'd)****6.4.5 Register Description****SYSTEM INTEGRITY (SI) CONTROL/STATUS REGISTER (SICSR)**

Read/Write

Reset Value: 000x 000x (00h)

7							0
0	AVD IE	AVD F	LVD RF	0	CSS IE	CSS D	WDG RF

Bit 7 = Reserved, always read as 0.

**Bit 6 = AVDIE Voltage Detector interrupt enable**

This bit is set and cleared by software. It enables an interrupt to be generated when the AVDF flag changes (toggles). The pending interrupt information is automatically cleared when software enters the AVD interrupt routine.

0: AVD interrupt disabled

1: AVD interrupt enabled

**Bit 5 = AVDF Voltage Detector flag**

This read-only bit is set and cleared by hardware. If the AVDIE bit is set, an interrupt request is generated when the AVDF bit changes value.

0:  $V_{DD}$  over  $V_{IT+(AVD)}$  threshold1:  $V_{DD}$  under  $V_{IT-(AVD)}$  threshold**Bit 4 = LVDRF LVD reset flag**

This bit indicates that the last Reset was generated by the LVD block. It is set by hardware (LVD reset) and cleared by software (writing zero). See WDGRF flag description for more details. When the LVD is disabled by OPTION BYTE, the LVDRF bit value is undefined.

Bit 3 = Reserved, must be kept cleared.

**Bit 2 = CSSIE Clock security syst interrupt enable**

This bit enables the interrupt when a disturbance is detected by the Clock Security System (CSSD

bit set). It is set and cleared by software.

0: Clock security system interrupt disabled

1: Clock security system interrupt enabled

When the CSS is disabled by OPTION BYTE, the CSSIE bit has no effect.

**Bit 1 = CSSD Clock security system detection**

This bit indicates that the safe oscillator of the Clock Security System block has been selected by hardware due to a disturbance on the main clock signal ( $f_{OSC}$ ). It is set by hardware and cleared by reading the SICSR register when the original oscillator recovers.

0: Safe oscillator is not active

1: Safe oscillator has been activated

When the CSS is disabled by OPTION BYTE, the CSSD bit value is forced to 0.

**Bit 0 = WDGRF Watchdog reset flag**

This bit indicates that the last Reset was generated by the Watchdog peripheral. It is set by hardware (watchdog reset) and cleared by software (writing zero) or an LVD Reset (to ensure a stable cleared state of the WDGRF flag when CPU starts).

Combined with the LVDRF flag information, the flag description is given by the following table.

RESET Sources	LVDRF	WDGRF
External RESET pin	0	0
Watchdog	0	1
LVD	1	X

**Application Notes**

The LVDRF flag is not cleared when another RESET type occurs (external or watchdog), the LVDRF flag remains set to keep trace of the original failure. In this case, a watchdog reset can be detected by software while an external reset can not.

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0025h	SICSR Reset Value	0	AVDIE 0	AVDF 0	LVDRF x	0	CSSIE 0	CSSD 0	WDGRF x

## 7 INTERRUPTS

### 7.1 INTRODUCTION

The ST7 enhanced interrupt management provides the following features:

- Hardware interrupts
- Software interrupt (TRAP)
- Nested or concurrent interrupt management with flexible interrupt priority and level management:
  - Up to 4 software programmable nesting levels
  - Up to 16 interrupt vectors fixed by hardware
  - 2 non-maskable events: RESET and TRAP

This interrupt management is based on:

- Bit 5 and bit 3 of the CPU CC register (I1:0),
- Interrupt software priority registers (ISPRx),
- Fixed interrupt vector addresses located at the high addresses of the memory map (FFE0h to FFFFh) sorted by hardware priority order.

This enhanced interrupt controller guarantees full upward compatibility with the standard (not nested) ST7 interrupt controller.

### 7.2 MASKING AND PROCESSING FLOW

The interrupt masking is managed by the I1 and I0 bits of the CC register and the ISPRx registers which give the interrupt software priority level of each interrupt vector (see Table 4). The processing flow is shown in Figure 17

When an interrupt request has to be serviced:

- Normal processing is suspended at the end of the current instruction execution.
- The PC, X, A and CC registers are saved onto the stack.
- I1 and I0 bits of CC register are set according to the corresponding values in the ISPRx registers of the serviced interrupt vector.
- The PC is then loaded with the interrupt vector of the interrupt to service and the first instruction of the interrupt service routine is fetched (refer to "Interrupt Mapping" table for vector addresses).

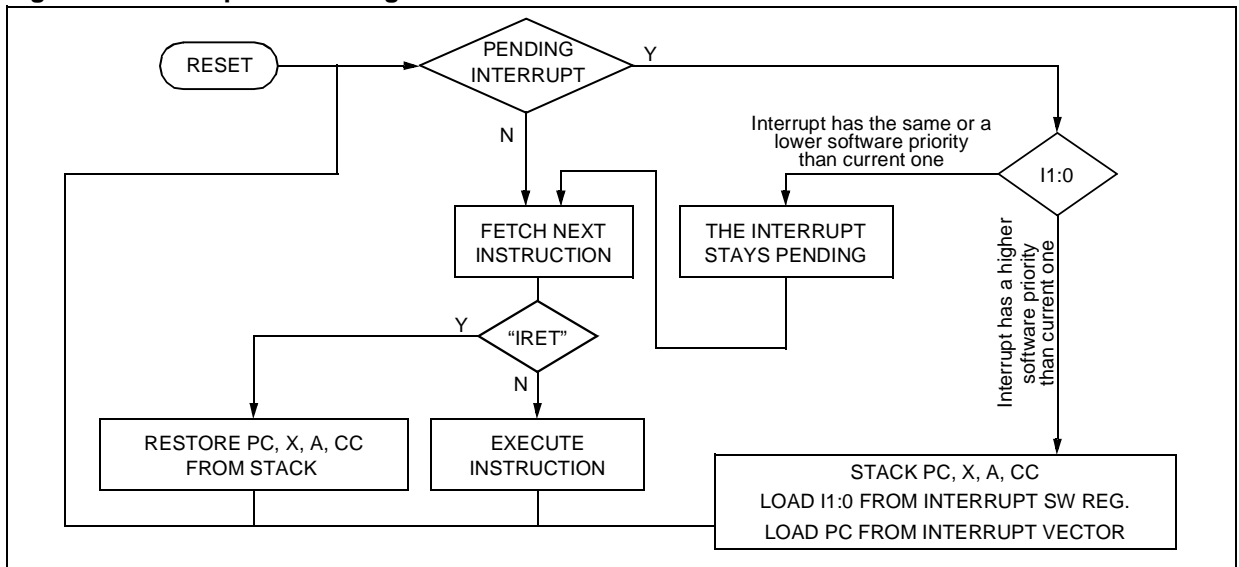
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

**Note:** As a consequence of the IRET instruction, the I1 and I0 bits will be restored from the stack and the program in the previous level will resume.

**Table 4. Interrupt Software Priority Levels**

Interrupt software priority	Level	I1	I0
Level 0 (main)	Low ↓ High	1	0
Level 1		0	1
Level 2		0	0
Level 3 (= interrupt disable)		1	1

**Figure 17. Interrupt Processing Flowchart**



## INTERRUPTS (Cont'd)

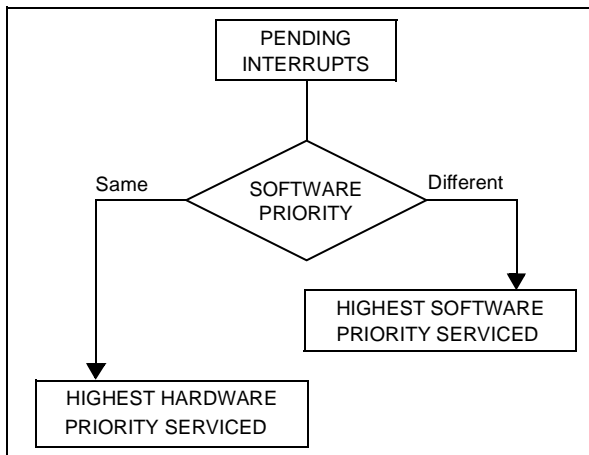
### Servicing Pending Interrupts

As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

- the highest software priority interrupt is serviced,
- if several interrupts have the same software priority then the interrupt with the highest hardware priority is serviced first.

Figure 18 describes this decision process.

**Figure 18. Priority Decision Process**



When an interrupt request is not serviced immediately, it is latched and then processed when its software priority combined with the hardware priority becomes the highest one.

**Note 1:** The hardware priority is exclusive while the software one is not. This allows the previous process to succeed with only one interrupt.

**Note 2:** RESET and TRAP are non-maskable and they can be considered as having the highest software priority in the decision process.

### Different Interrupt Vector Sources

Two interrupt source types are managed by the ST7 interrupt controller: the non-maskable type (RESET and TRAP) and the maskable type (external or from internal peripherals).

### Non-Maskable Sources

These sources are processed regardless of the state of the I1 and I0 bits of the CC register (see Figure 17). After stacking the PC, X, A and CC registers (except for RESET), the corresponding vector is loaded in the PC register and the I1 and I0 bits of the CC are set to disable interrupts (level

3). These sources allow the processor to exit HALT mode.

#### ■ TRAP (Non Maskable Software Interrupt)

This software interrupt is serviced when the TRAP instruction is executed. It will be serviced according to the flowchart on Figure 17 as a TLI.

#### ■ RESET

The RESET source has the highest priority in the ST7. This means that the first current routine has the highest software priority (level 3) and the highest hardware priority.

See the RESET chapter for more details.

### Maskable Sources

Maskable interrupt vector sources can be serviced if the corresponding interrupt is enabled and if its own interrupt software priority (in ISPRx registers) is higher than the one currently being serviced (I1 and I0 in CC register). If any of these two conditions is false, the interrupt is latched and thus remains pending.

#### ■ External Interrupts

External interrupts allow the processor to exit from HALT low power mode.

External interrupt sensitivity is software selectable through the Miscellaneous registers (MISCRx).

External interrupt triggered on edge will be latched and the interrupt request automatically cleared upon entering the interrupt service routine.

If several input pins of a group connected to the same interrupt vector request an interrupt simultaneously, the interrupt vector will be serviced. Software can read the pin levels to identify which pin(s) are the source of the interrupt.

If several input pins are selected simultaneously as interrupt source, these are logically NANDed. For this reason if one of the interrupt pins is tied low, it masks the other ones.

#### ■ Peripheral Interrupts

Usually the peripheral interrupts cause the MCU to exit from HALT mode except those mentioned in the "Interrupt Mapping" table.

A peripheral interrupt occurs when a specific flag is set in the peripheral status registers and if the corresponding enable bit is set in the peripheral control register.

The general sequence for clearing an interrupt is based on an access to the status register followed by a read or write to an associated register.

**Note:** The clearing sequence resets the internal latch. A pending interrupt (i.e. waiting for being serviced) will therefore be lost if the clear sequence is executed.

INTERRUPTS (Cont'd)

7.3 INTERRUPTS AND LOW POWER MODES

All interrupts allow the processor to exit the WAIT low power mode. On the contrary, only external and other specified interrupts allow the processor to exit the HALT modes (see column "Exit from HALT" in "Interrupt Mapping" table). When several pending interrupts are present while exiting HALT mode, the first one serviced can only be an interrupt with exit from HALT mode capability and it is selected through the same decision process shown in Figure 18.

**Note:** If an interrupt, that is not able to Exit from HALT mode, is pending with the highest priority when exiting HALT mode, this interrupt is serviced after the first one serviced.

7.4 CONCURRENT & NESTED MANAGEMENT

The following Figure 19 and Figure 20 show two different interrupt management modes. The first is called concurrent mode and does not allow an interrupt to be interrupted, unlike the nested mode in Figure 20. The interrupt hardware priority is given in this order from the lowest to the highest: MAIN, IT4, IT3, IT2, IT1, IT0. The software priority is given for each interrupt.

**Warning:** A stack overflow may occur without notifying the software of the failure.

**Note:** TLI (Top Level Interrupt) is not available in this product.

Figure 19. Concurrent Interrupt Management

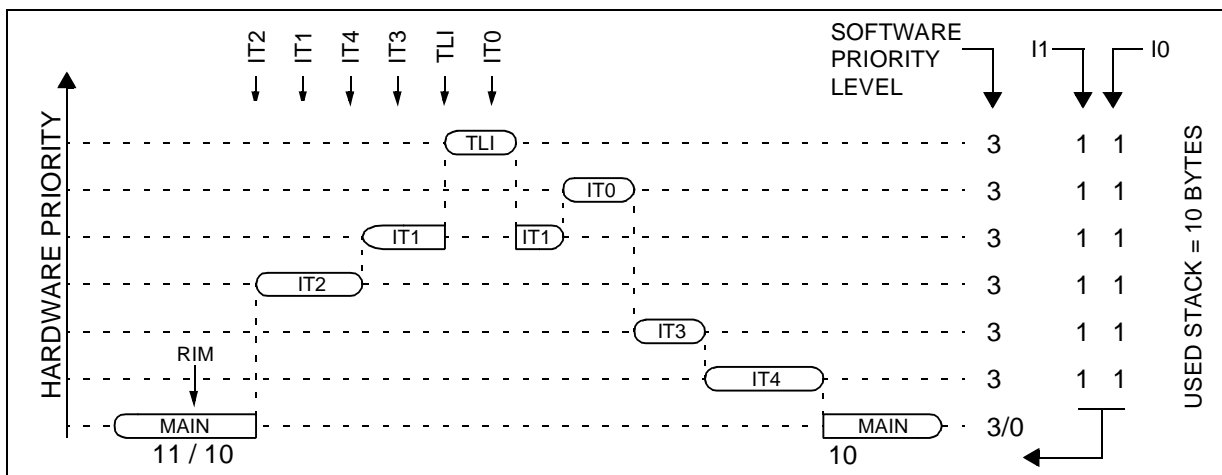
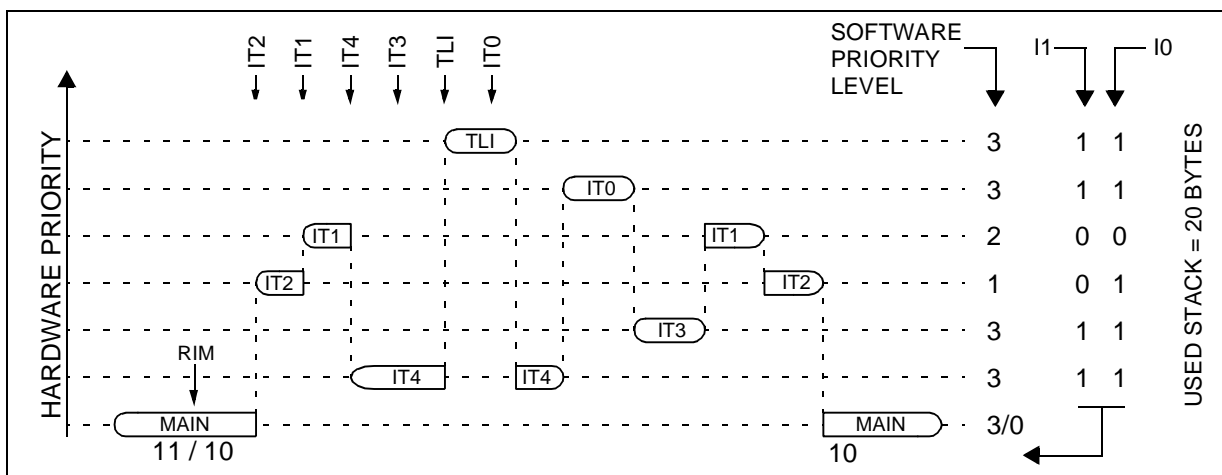


Figure 20. Nested Interrupt Management



## INTERRUPTS (Cont'd)

### 7.5 INTERRUPT REGISTER DESCRIPTION

#### CPU CC REGISTER INTERRUPT BITS

Read/Write

Reset Value: 111x 1010 (xAh)

7							0
1	1	I1	H	I0	N	Z	C

Bit 5, 3 = **I1, I0** *Software Interrupt Priority*

These two bits indicate the current interrupt software priority.

Interrupt Software Priority	Level	I1	I0
Level 0 (main)	Low	1	0
Level 1	↓	0	1
Level 2		0	0
Level 3 (= interrupt disable*)		High	1

These two bits are set/cleared by hardware when entering in interrupt. The loaded value is given by the corresponding bits in the interrupt software priority registers (ISPRx).

They can be also set/cleared by software with the RIM, SIM, HALT, WFI, IRET and PUSH/POP instructions (see "Interrupt Dedicated Instruction Set" table).

**\*Note:** TRAP and RESET events are non maskable sources and can interrupt a level 3 program.

#### INTERRUPT SOFTWARE PRIORITY REGISTERS (ISPRX)

Read/Write (bits 7:4 of **ISPR3** are read only)

Reset Value: 1111 1111 (FFh)

	7						0
ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1	I1_0
ISPR1	I1_7	I0_7	I1_6	I0_6	I1_5	I0_5	I1_4
ISPR2	I1_11	I0_11	I1_10	I0_10	I1_9	I0_9	I1_8
ISPR3	1	1	1	1	I1_13	I0_13	I1_12

These four registers contain the interrupt software priority of each interrupt vector.

– Each interrupt vector (except RESET and TRAP) has corresponding bits in these registers where its own software priority is stored. This correspondence is shown in the following table.

Vector Address	ISPRx Bits
FFFBh-FFFAh	ei0
FFF9h-FFF8h	ei1
...	...
FFE1h-FFE0h	Not used

– Each I1\_x and I0\_x bit value in the ISPRx registers has the same meaning as the I1 and I0 bits in the CC register.

– Level 0 can not be written (I1\_x=1, I0\_x=0). In this case, the previously stored value is kept. (example: previous=CFh, write=64h, result=44h)

The RESET and TRAP vectors have no software priorities. When one is serviced, the I1 and I0 bits of the CC register are both set.

**Caution:** If the I1\_x and I0\_x bits are modified while the interrupt x is executed the following behaviour has to be considered: If the interrupt x is still pending (new interrupt or flag not cleared) and the new software priority is higher than the previous one, the interrupt x is re-entered. Otherwise, the software priority stays unchanged up to the next interrupt request (after the IRET of the interrupt x).

**Table 5. Interrupt Mapping**

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT	Address Vector
	RESET	Reset	N/A	Highest Priority ↓ Lowest Priority	yes	FFFEh-FFFFh
	TRAP	Software Interrupt			no	FFFCh-FFFDh
0	ei0	External Interrupt Port A7..0 (C5..0 <sup>1</sup> )			yes	FFFAh-FFFBh
1	ei1	External Interrupt Port B7..0 (C5..0 <sup>1</sup> )			no	FFF8h-FFF9h
2	CSS	Clock Filter Interrupt	CRSR		no	FFF6h-FFF7h
3	SPI	SPI Peripheral Interrupts	SPISR		yes	FFF4h-FFF5h
4	TIMER A	TIMER A Peripheral Interrupts	TASR		no	FFF2h-FFF3h
5	MCC	Time base interrupt	MCCSR		yes	FFF0h-FFF1h
6	TIMER B	TIMER B Peripheral Interrupts	TBSR		no	FFEEh-FFEFh
7	AVD	Auxiliary Voltage Detector interrupt	SICSR			FFEC h-FFEDh
8		Not used				FFEAh-FFEBh
9		Not used				FFE8h-FFE9h
10	SCI	SCI Peripheral Interrupt	SCISR		no	FFE6h-FFE7h
11	I <sup>2</sup> C	I <sup>2</sup> C Peripheral Interrupt	I2CSRx	no	FFE4h-FFE5h	
12		Not Used			FFE2h-FFE3h	
13		Not Used			FFE0h-FFE1h	

**Note 1.** Configurable by option byte.

**Table 6. Nested Interrupts Register Map and Reset Values**

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
001Ch	ISPR0 Reset Value	SPI		CSS		EI1		EI0	
		I1_3 1	I0_3 1	I1_2 1	I0_2 1	I1_1 1	I0_1 1	I0_1 1	I0_0 1
001Dh	ISPR1 Reset Value	AVD		TIMERB		MCC		TIMERA	
		I1_7 1	I0_7 1	I1_6 1	I0_6 1	I1_5 1	I0_5 1	I1_4 1	I0_4 1
001Eh	ISPR2 Reset Value	I <sup>2</sup> C		SCI		Not Used		Not Used	
		I1_11 1	I0_11 1	I1_10 1	I0_10 1	I1_9 1	I0_9 1	I1_8 1	I0_8 1
001Fh	ISPR3 Reset Value					Not Used		Not Used	
		1	1	1	1	I1_13 1	I0_13 1	I1_12 1	I0_12 1



## 8 POWER SAVING MODES

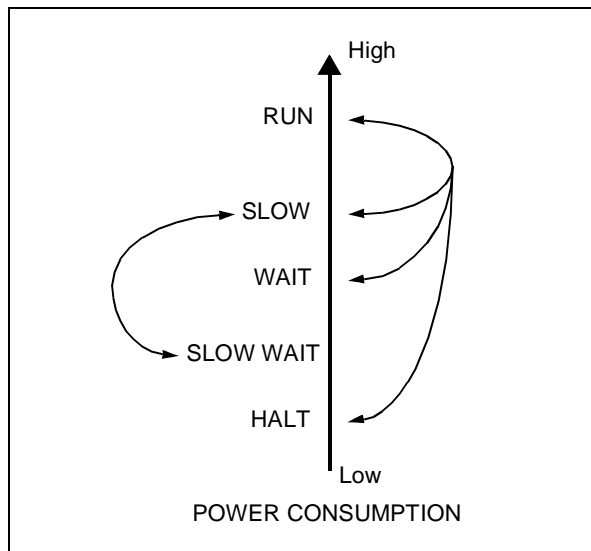
### 8.1 INTRODUCTION

To give a large measure of flexibility to the application in terms of power consumption, three main power saving modes are implemented in the ST7 (see Figure 21).

After a RESET the normal operating mode is selected by default (RUN mode). This mode drives the device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency divided by 2 ( $f_{CPU}$ ).

From Run mode, the different power saving modes may be selected by setting the relevant register bits or by calling the specific ST7 software instruction whose action depends on the oscillator status.

**Figure 21. Power Saving Mode Transitions**



### 8.2 SLOW MODE

This mode has two targets:

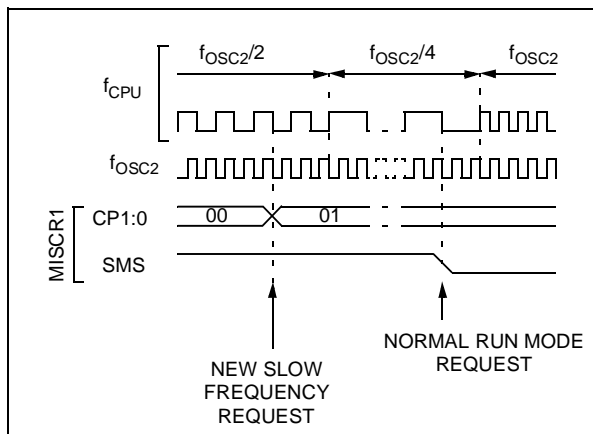
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency ( $f_{CPU}$ ) to the available supply voltage.

SLOW mode is controlled by three bits in the MISR1 register: the SMS bit which enables or disables Slow mode and two CPx bits which select the internal slow frequency ( $f_{CPU}$ ).

In this mode, the oscillator frequency can be divided by 4, 8, 16 or 32 instead of 2 in normal operating mode. The CPU and peripherals are clocked at this lower frequency.

**Note:** SLOW-WAIT mode is activated when entering the WAIT mode while the device is already in SLOW mode.

**Figure 22. SLOW Mode Clock Transitions**



POWER SAVING MODES (Cont'd)

8.3 WAIT MODE

WAIT mode places the MCU in a low power consumption mode by stopping the CPU.

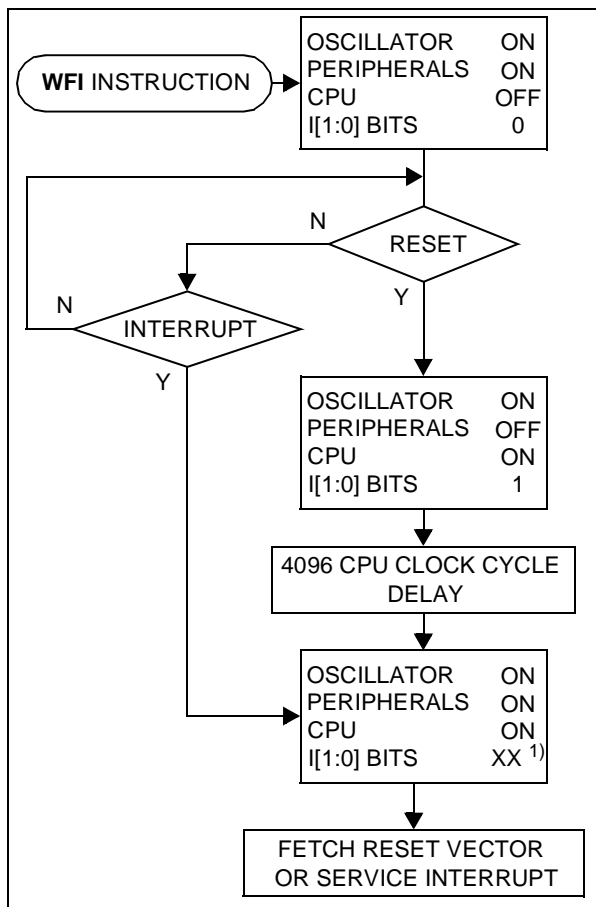
This power saving mode is selected by calling the "WFI" ST7 software instruction.

All peripherals remain active. During WAIT mode, the I[1:0] bits in the CC register are forced to '10b', to enable all interrupts. All other registers and memory remain unchanged. The MCU remains in WAIT mode until an interrupt or Reset occurs, whereupon the Program Counter branches to the starting address of the interrupt or Reset service routine.

The MCU will remain in WAIT mode until a Reset or an Interrupt occurs, causing it to wake up.

Refer to Figure 23.

Figure 23. WAIT Mode Flowchart



Note:

1. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits in the CC register are set during the interrupt routine and cleared when the CC register is popped.

## 8.4 ACTIVE-HALT AND HALT MODES

ACTIVE-HALT and HALT modes are the two lowest power consumption modes of the MCU. They are both entered by executing the 'HALT' instruction. The decision to enter either in ACTIVE-HALT or HALT mode is given by the MCC/RTC interrupt enable flag (OIE bit in MCCR register).

MCCR OIE bit	Power Saving Mode entered when HALT instruction is executed
0	HALT mode
1	ACTIVE-HALT mode

### 8.4.1 ACTIVE-HALT MODE

ACTIVE-HALT mode is the lowest power consumption mode of the MCU with a real time clock available. It is entered by executing the 'HALT' instruction when the OIE bit of the Main Clock Controller Status register (MCCR) is set.

The MCU can exit ACTIVE-HALT mode on reception of either an MCC/RTC interrupt, a specific interrupt (see Table 5, "Interrupt Mapping," on page 32) or a RESET. When exiting ACTIVE-HALT mode by means of an interrupt, no 4096 CPU cycle delay occurs. The CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see Figure 25).

When entering ACTIVE-HALT mode, the I[1:0] bits in the CC register are forced to '10b' to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately.

In ACTIVE-HALT mode, only the main oscillator and its associated counter (MCC/RTC) are running to keep a wake-up time base. All other peripherals are not clocked except those which get their clock supply from another clock generator (such as external or auxiliary oscillator).

The safeguard against staying locked in ACTIVE-HALT mode is provided by the oscillator interrupt.

**Note:** As soon as the interrupt capability of one of the oscillators is selected (MCCR.OIE bit set), entering ACTIVE-HALT mode while the Watchdog is active does not generate a RESET.

This means that the device cannot spend more than a defined delay in this power saving mode.

Figure 24. ACTIVE-HALT Timing Overview

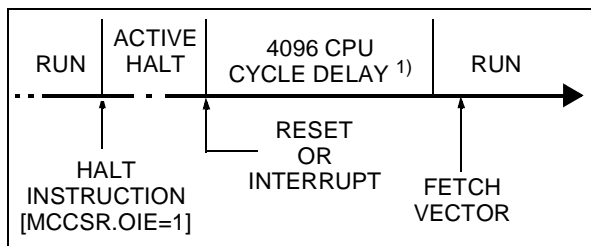
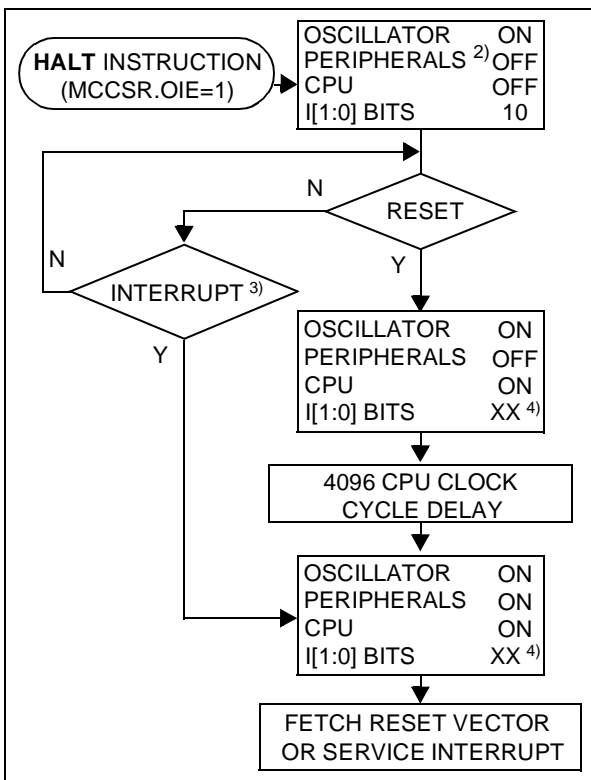


Figure 25. ACTIVE-HALT Mode Flowchart



#### Notes:

1. This delay occurs only if the MCU exits ACTIVE-HALT mode by means of a RESET.
2. Peripheral clocked with an external clock source can still be active.
3. Only the MCC/RTC interrupt and some specific interrupts can exit the MCU from ACTIVE-HALT mode (such as external interrupt). Refer to Table 5, "Interrupt Mapping," on page 32 for more details.
4. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and restored when the CC register is popped.

POWER SAVING MODES (Cont'd)

8.5 HALT MODE

The HALT mode is the lowest power consumption mode of the MCU. It is entered by executing the ST7 HALT instruction (see Figure 27).

The MCU can exit HALT mode on reception of either a specific interrupt (see Table 5, "Interrupt Mapping," on page 32) or a RESET. When exiting HALT mode by means of a RESET or an interrupt, the oscillator is immediately turned on and the 4096 CPU cycle delay is used to stabilize the oscillator. After the start up delay, the CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see Figure 26). When entering HALT mode, the I[1:0] bits in the CC register are forced to '10b' to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes immediately.

In the HALT mode the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. All peripherals are not clocked except the ones which get their clock supply from another clock generator (such as an external or auxiliary oscillator).

The compatibility of Watchdog operation with HALT mode is configured by the "WDGHALT" option bit of the option byte. The HALT instruction when executed while the Watchdog system is enabled, can generate a Watchdog RESET (see Section 15.1 "OPTION BYTES" on page 157 for more details).

Figure 26. HALT Mode Timing Overview

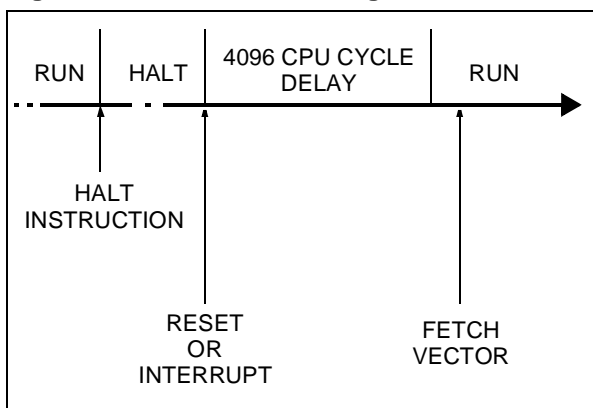
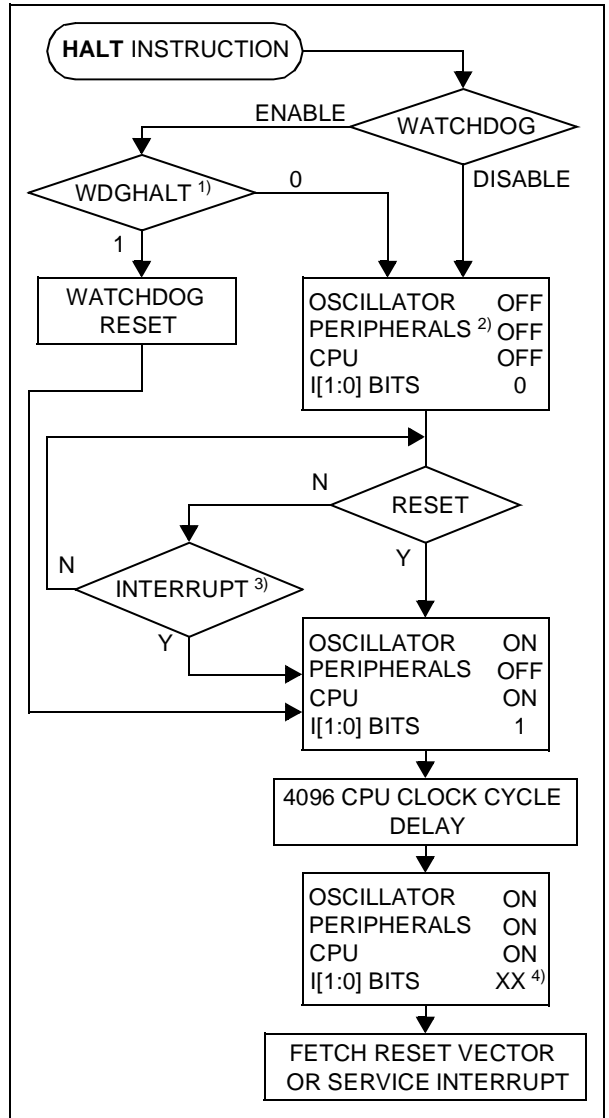


Figure 27. HALT Mode Flowchart



Notes:

1. WDGHALT is an option bit. See option byte section for more details.
2. Peripheral clocked with an external clock source can still be active.
3. Only some specific interrupts can exit the MCU from HALT mode (such as external interrupt). Refer to Table 5, "Interrupt Mapping," on page 32 for more details.
4. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits in the CC register are set during the interrupt routine and cleared when the CC register is popped.

**POWER SAVING MODES** (Cont'd)**8.5.0.1 Halt Mode Recommendations**

- Make sure that an external event is available to wake up the microcontroller from Halt mode.
  - When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as “Input Pull-up with Interrupt” before executing the HALT instruction. The main reason for this is that the I/O may be wrongly configured due to external interference or by an unforeseen logical condition.
  - For the same reason, reinitialize the level sensitivity of each external interrupt as a precautionary measure.
- The opcode for the HALT instruction is 0x8E. To avoid an unexpected HALT instruction due to a program counter failure, it is advised to clear all occurrences of the data value 0x8E from memory. For example, avoid defining a constant in ROM with the value 0x8E.
  - As the HALT instruction clears the interrupt mask in the CC register to allow interrupts, the user may choose to clear all pending interrupt bits before executing the HALT instruction. This avoids entering other peripheral interrupt routines after executing the external interrupt routine corresponding to the wake-up event (reset or external interrupt).

## 9 I/O PORTS

### 9.1 INTRODUCTION

The I/O ports allow data transfer. An I/O port can contain up to 8 pins. Each pin can be programmed independently either as a digital input or digital output. In addition, specific pins may have several other functions. These functions can include external interrupt, alternate signal input/output for on-chip peripherals or analog input.

### 9.2 FUNCTIONAL DESCRIPTION

A Data Register (DR) and a Data Direction Register (DDR) are always associated with each port. The Option Register (OR), which allows input/output options, may or may not be implemented. The following description takes into account the OR register. Refer to the Port Configuration table for device specific information.

An I/O pin is programmed using the corresponding bits in the DDR, DR and OR registers: bit  $x$  corresponding to pin  $x$  of the port.

Figure 28 shows the generic I/O block diagram.

#### 9.2.1 Input Modes

Clearing the DDR $x$  bit selects input mode. In this mode, reading its DR bit returns the digital value from that I/O pin.

If an OR bit is available, different input modes can be configured by software: floating or pull-up. Refer to I/O Port Implementation section for configuration.

#### Notes:

1. Writing to the DR modifies the latch value but does not change the state of the input pin.
2. Do not use read/modify/write instructions (BSET/BRES) to modify the DR register.

#### External Interrupt Function

Depending on the device, setting the OR $x$  bit while in input mode can configure an I/O as an input with interrupt. In this configuration, a signal edge or level input on the I/O generates an interrupt request via the corresponding interrupt vector (eix).

Falling or rising edge sensitivity is programmed independently for each interrupt vector. The External Interrupt Control Register (EICR) or the Miscellaneous Register controls this sensitivity, depending on the device.

A device may have up to 7 external interrupts. Several pins may be tied to one external interrupt vector. Refer to Pin Description to see which ports have external interrupts.

If several I/O interrupt pins on the same interrupt vector are selected simultaneously, they are logically combined. For this reason if one of the interrupt pins is tied low, it may mask the others.

External interrupts are hardware interrupts. Fetching the corresponding interrupt vector automatically clears the request latch. Modifying the sensitivity bits will clear any pending interrupts.

#### 9.2.2 Output Modes

Setting the DDR $x$  bit selects output mode. Writing to the DR bits applies a digital value to the I/O through the latch. Reading the DR bits returns the previously stored value.

If an OR bit is available, different output modes can be selected by software: push-pull or open-drain. Refer to I/O Port Implementation section for configuration.

#### DR Value and Output Pin Status

DR	Push-Pull	Open-Drain
0	$V_{OL}$	$V_{OL}$
1	$V_{OH}$	Floating

#### 9.2.3 Alternate Functions

Many ST7s I/Os have one or more alternate functions. These may include output signals from, or input signals to, on-chip peripherals. The Device Pin Description table describes which peripheral signals can be input/output to which ports.

A signal coming from an on-chip peripheral can be output on an I/O. To do this, enable the on-chip peripheral as an output (enable bit in the peripheral's control register). The peripheral configures the I/O as an output and takes priority over standard I/O programming. The I/O's state is readable by addressing the corresponding I/O data register.

Configuring an I/O as floating enables alternate function input. It is not recommended to configure an I/O as pull-up as this will increase current consumption. Before using an I/O as an alternate input, configure it without interrupt. Otherwise spurious interrupts can occur.

Configure an I/O as input floating for an on-chip peripheral signal which can be input and output.

#### Caution:

I/Os which can be configured as both an analog and digital alternate function need special attention. The user must control the peripherals so that the signals do not arrive at the same time on the same pin. If an external clock is used, only the clock alternate function should be employed on that I/O pin and not the other alternate function.

I/O PORTS (Cont'd)

Figure 28. I/O Port General Block Diagram

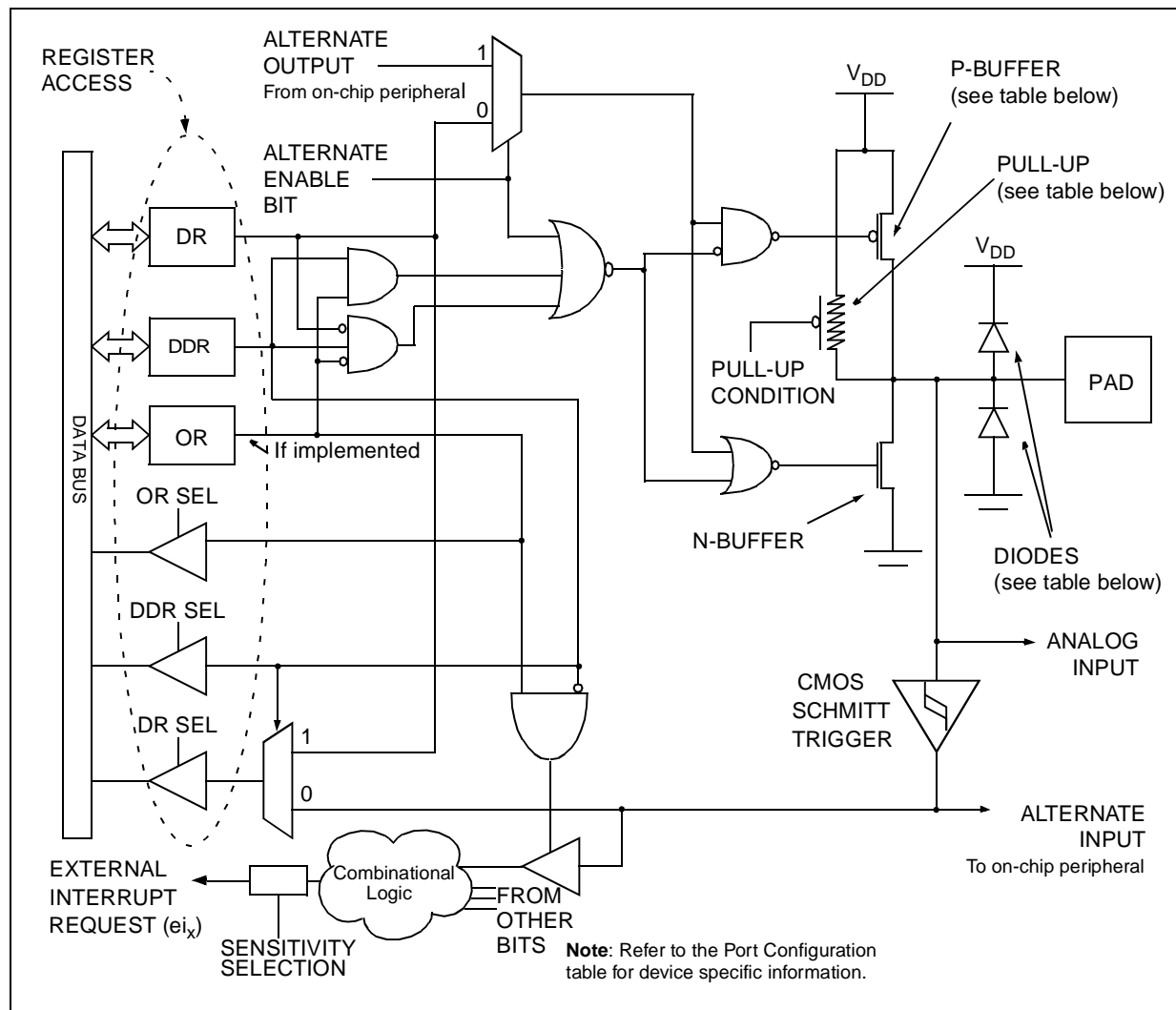


Table 7. I/O Port Mode Options

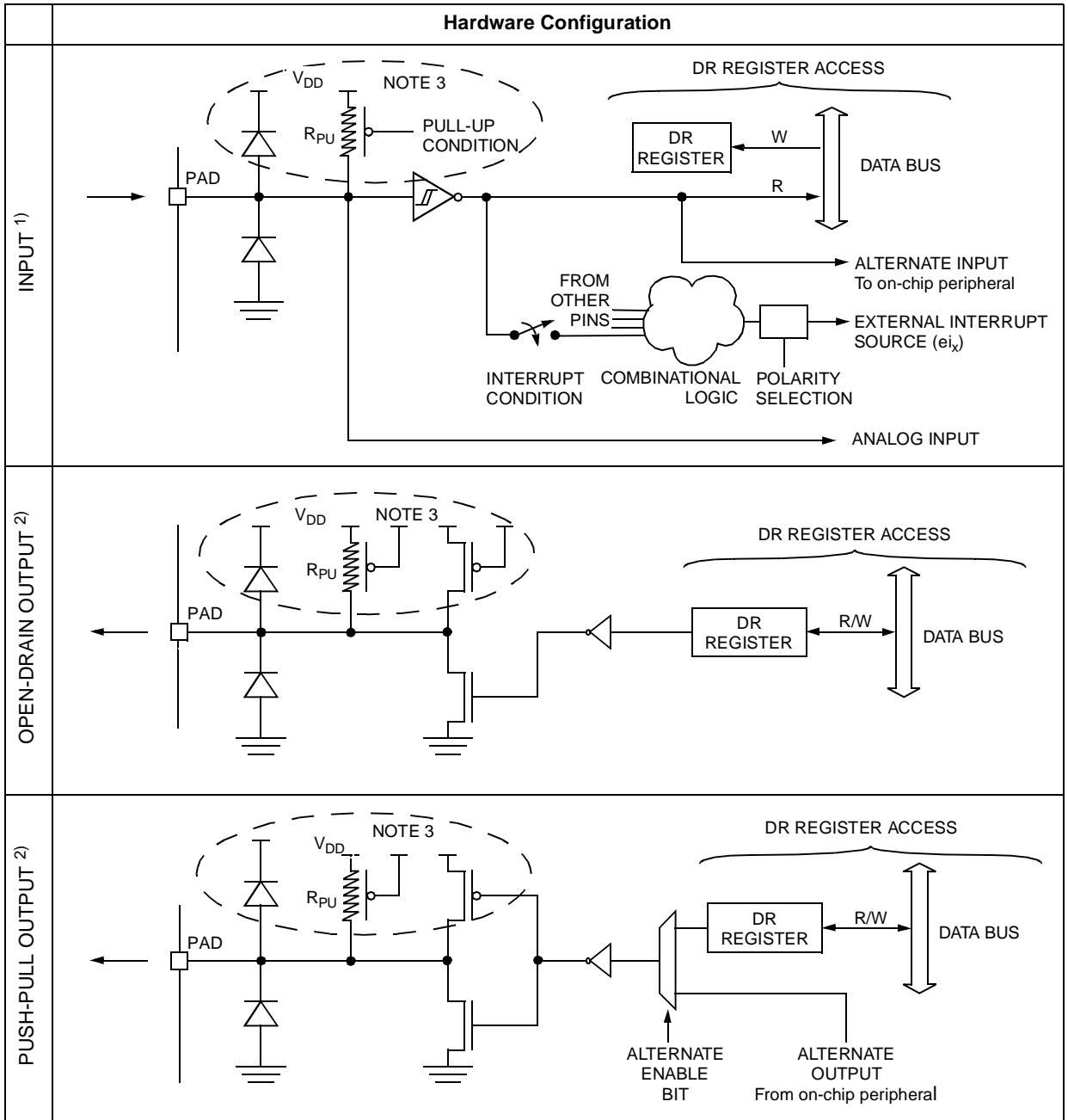
Configuration Mode		Pull-Up	P-Buffer	Diodes	
				to V <sub>DD</sub>	to V <sub>SS</sub>
Input	Floating with/without Interrupt	Off	Off	On	On
	Pull-up with/without Interrupt	On			
Output	Push-pull	Off	On	NI (see note)	On
	Open Drain (logic level)		Off		
	True Open Drain	NI	NI		

**Legend:** NI - not implemented  
 Off - implemented not activated  
 On - implemented and activated

**Note:** The diode to V<sub>DD</sub> is not implemented in the true open drain pads. A local protection between the pad and V<sub>OL</sub> is implemented to protect the device against positive stress.

I/O PORTS (Cont'd)

Table 8. I/O Configurations



Notes:

1. When the I/O port is in input configuration and the associated alternate function is enabled as an output, reading the DR register will read the alternate function output status.
2. When the I/O port is in output configuration and the associated alternate function is enabled as an input, the alternate I/O function reads the pin status given by the DR register content.
3. For true open drain, these elements are not implemented.



**I/O PORTS** (Cont'd)**Analog alternate function**

Configure the I/O as floating input to use an ADC input. The analog multiplexer (controlled by the ADC registers) switches the analog voltage present on the selected pin to the common analog rail, connected to the ADC input.

**Analog Recommendations**

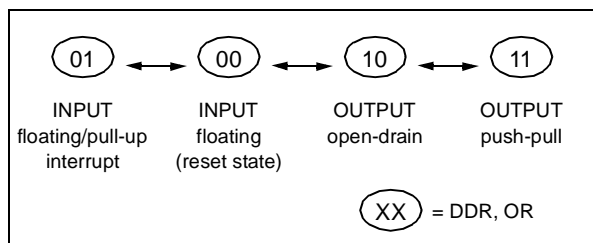
Do not change the voltage level or loading on any I/O while conversion is in progress. Do not have clocking pins located close to a selected analog pin.

**WARNING:** The analog input voltage level must be within the limits stated in the absolute maximum ratings.

**9.3 I/O PORT IMPLEMENTATION**

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific I/O port features such as ADC input or open drain.

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in Figure 29. Other transitions are potentially risky and should be avoided, since they may present unwanted side-effects such as spurious interrupt generation.

**Figure 29. Interrupt I/O Port State Transitions****9.4 UNUSED I/O PINS**

Unused I/O pins must be connected to fixed voltage levels. Refer to Section 13.8.

**9.5 LOW POWER MODES**

Mode	Description
WAIT	No effect on I/O ports. External interrupts cause the device to exit from WAIT mode.
HALT	No effect on I/O ports. External interrupts cause the device to exit from HALT mode.

**9.6 INTERRUPTS**

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and OR registers and if the I bit in the CC register is cleared (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
External interrupt on selected external event	-	DDRx ORx	Yes	Yes

I/O PORTS (Cont'd)

9.7 DEVICE-SPECIFIC I/O PORT CONFIGURATION

The I/O port register configurations are summarised as follows.

**Interrupt Ports**

**PA7, PA5, PA3:0, PB7:0, PC5:0** (with pull-up)

MODE	DDR	OR
floating input	0	0
pull-up interrupt input	0	1
open drain output	1	0
push-pull output	1	1

**True Open Drain Interrupt Ports**

**PA6, PA4** (without pull-up)

MODE	DDR	OR
floating input	0	0
floating interrupt input	0	1
open drain (high sink ports)	1	X

**Table 9. Port Configuration**

Port	Pin Name	Input (DDR = 0)		Output (DDR = 1)		
		OR = 0	OR = 1	OR = 0	OR = 1	High-Sink
Port A	PA7	floating	pull-up interrupt	open drain	push-pull	Yes
	PA6	floating	floating interrupt	true open-drain		
	PA5	floating	pull-up interrupt	open drain	push-pull	
	PA4	floating	floating interrupt	true open-drain		
	PA3:0	floating	pull-up interrupt	open drain	push-pull	
Port B	PB7:0	floating	pull-up interrupt	open drain	push-pull	No
Port C	PC5:0	floating	pull-up interrupt	open drain	push-pull	

## I/O PORTS (Cont'd)

## 9.8 I/O PORT REGISTER DESCRIPTION

**DATA REGISTER (DR)**

Port x Data Register

PxDR with x = A, B or C.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0 = **D[7:0]** *Data register 8 bits.*

The DR register has a specific behaviour according to the selected input/output configuration. Writing the DR register is always taken into account even if the pin is configured as an input; this allows always having the expected level on the pin when toggling to output mode. Reading the DR register returns either the DR register latch content (pin configured as output) or the digital value applied to the I/O pin (pin configured as input).

**DATA DIRECTION REGISTER (DDR)**

Port x Data Direction Register

PxDDR with x = A, B or C.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0

Bits 7:0 = **DD[7:0]** *Data direction register 8 bits.*

The DDR register gives the input/output direction configuration of the pins. Each bit is set and cleared by software.

0: Input mode

1: Output mode

**OPTION REGISTER (OR)**

Port x Option Register

PxOR with x = A, B or C.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
O7	O6	O5	O4	O3	O2	O1	O0

Bits 7:0 = **O[7:0]** *Option register 8 bits.*

For specific I/O pins, this register is not implemented. In this case the DDR register is enough to select the I/O pin configuration.

The OR register allows to distinguish: in input mode if the pull-up with interrupt capability or the basic pull-up configuration is selected, in output mode if the push-pull or open drain configuration is selected.

Each bit is set and cleared by software.

Input mode:

0: Floating input

1: Pull-up input with or without interrupt

Output mode:

0: Output open drain (with P-Buffer unactivated)

1: Output push-pull (when available)

## I/O PORTS (Cont'd)

Table 10. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Reset Value of all I/O port registers		0	0	0	0	0	0	0	0
0000h	PCDR	MSB							LSB
0001h	PCDDR								
0002h	PCOR								
0004h	PBDR	MSB							LSB
0005h	PBDDR								
0006h	PBOR								
0008h	PADR	MSB							LSB
0009h	PADDR								
000Ah	PAOR								

## 10 MISCELLANEOUS REGISTERS

The miscellaneous registers allow control over several different features such as the external interrupts or the I/O alternate functions.

### 10.1 I/O PORT INTERRUPT SENSITIVITY

The external interrupt sensitivity is controlled by the ISxx bits of the Miscellaneous register and the OPTION BYTE. This control allows you to have two fully independent external interrupt source sensitivities with configurable sources (using the EXTIT option bit) as shown in Figure 30 and Figure 31.

Each external interrupt source can be generated on four different events on the pin:

- Falling edge
- Rising edge
- Falling and rising edge
- Falling edge and low level

To guarantee correct functionality, the sensitivity bits in the MISCR1 register must be modified only when the I[1:0] bits in the CC register are set to 1 (interrupt masked). See Section 9.8 "I/O PORT REGISTER DESCRIPTION" on page 43 and Section 10.3 "MISCELLANEOUS REGISTER DESCRIPTION" on page 46 for more details on the programming.

### 10.2 I/O PORT ALTERNATE FUNCTIONS

The MISCR registers manage four I/O port miscellaneous alternate functions:

- Main clock signal ( $f_{CPU}$ ) output on PC2
- SPI pin configuration:
  - $\overline{SS}$  pin internal control to use the PB7 I/O port function while the SPI is active.
  - Master output capability on the MOSI pin (PB4) deactivated while the SPI is active.
  - Slave output capability on the MISO pin (PB5) deactivated while the SPI is active.

These functions are described in detail in the Section 10.3 "MISCELLANEOUS REGISTER DESCRIPTION" on page 46.

Figure 30. Ext. Interrupt Sensitivity (EXTIT=0)

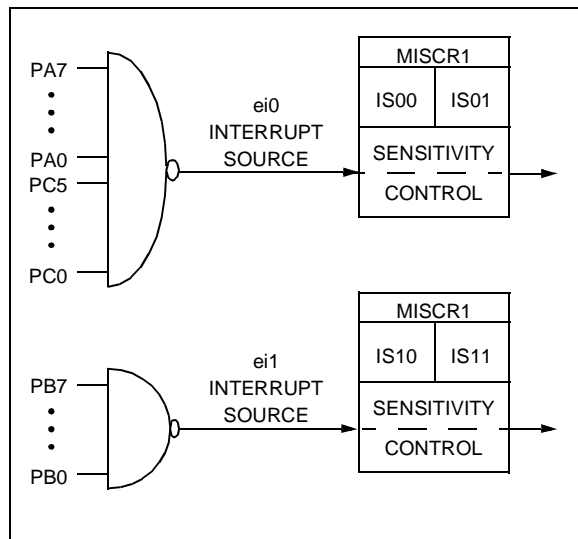
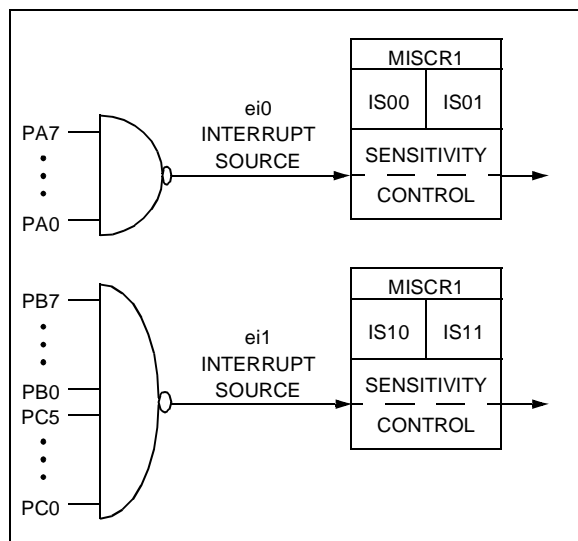


Figure 31. Ext. Interrupt Sensitivity (EXTIT=1)



MISCELLANEOUS REGISTERS (Cont'd)

10.3 MISCELLANEOUS REGISTER DESCRIPTION

MISCELLANEOUS REGISTER 1 (MISCR1)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
IS11	IS10	MCO	IS01	IS00	CP1	CP0	SMS

Bits 7:6 = **IS1[1:0]** *ei1 sensitivity*

The interrupt sensitivity, defined using the IS1[1:0] bits, is applied to the ei1 external interrupts. These two bits can be written only when the I[1:0] bits in the CC register are set to 1 (interrupt masked).

ei1: Port B (C optional)

External Interrupt Sensitivity	IS11	IS10
Falling edge & low level	0	0
Rising edge only	0	1
Falling edge only	1	0
Rising and falling edge	1	1

Bit 5 = **MCO** *Main clock out selection*

This bit enables the MCO alternate function on the PC2 I/O port. It is set and cleared by software.

0: MCO alternate function disabled (I/O pin free for general-purpose I/O)

1: MCO alternate function enabled ( $f_{CPU}$  on I/O port)

Bits 4:3 = **IS0[1:0]** *ei0 sensitivity*

The interrupt sensitivity, defined using the IS0[1:0] bits, is applied to the ei0 external interrupts. These two bits can be written only when the I[1:0] bits in the CC register are set to 1 (interrupt masked).

ei0: Port A (C optional)

External Interrupt Sensitivity	IS01	IS00
Falling edge & low level	0	0
Rising edge only	0	1
Falling edge only	1	0
Rising and falling edge	1	1

Bits 2:1 = **CP[1:0]** *CPU clock prescaler*

These bits select the CPU clock prescaler which is applied in the various slow modes. Their action is conditioned by the setting of the SMS bit. These two bits are set and cleared by software

$f_{CPU}$ in SLOW mode	CP1	CP0
$f_{OSC2} / 2$	0	0
$f_{OSC2} / 4$	1	0
$f_{OSC2} / 8$	0	1
$f_{OSC2} / 16$	1	1

Bit 0 = **SMS** *Slow mode select*

This bit is set and cleared by software.

0: Normal mode.  $f_{CPU} = f_{OSC2}$

1: Slow mode.  $f_{CPU}$  is given by CP1, CP0

See low power consumption mode and MCC chapters for more details.

**MISCELLANEOUS REGISTERS** (Cont'd)**MISCELLANEOUS REGISTER 2 (MISCR2)**

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	MOD	SOD	SSM	SSI

**Caution:** This register has been provided for compatibility with the ST72254 family only. The same bits are available in the SPICSR register. New applications must use the SPICSR register. Do not use both registers, this will cause the SPI to malfunction.

Bits 7:4 = **Reserved** always read as 0

Bits 3 = **MOD SPI Master Output Disable**

This bit is set and cleared by software. When set, it disables the SPI Master (MOSI) output signal.

0: SPI Master Output enabled.

1: SPI Master Output disabled.

Bit 2 = **SOD SPI Slave Output Disable**

This bit is set and cleared by software. When set it disable the SPI Slave (MISO) output signal.

0: SPI Slave Output enabled.

1: SPI Slave Output disabled.

Bit 1 = **SSM  $\overline{SS}$  mode selection**

This bit is set and cleared by software.

0: Normal mode - the level of the SPI  $\overline{SS}$  signal is input from the external  $\overline{SS}$  pin.

1: I/O mode, the level of the SPI  $\overline{SS}$  signal is read from the SSI bit.

Bit 0 = **SSI  $\overline{SS}$  internal mode**

This bit replaces the  $\overline{SS}$  pin of the SPI when the SSM bit is set to 1. (see SPI description). It is set and cleared by software.

**Table 11. Miscellaneous Register Map and Reset Values**

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0020h	<b>MISCR1</b> Reset Value	IS11 0	IS10 0	MCO 0	IS01 0	IS00 0	CP1 0	CP0 0	SMS 0
0040h	<b>MISCR2</b> Reset Value	0	0	0	0	MOD 0	SOD 0	SSM 0	SSI 0

## 11 ON-CHIP PERIPHERALS

### 11.1 WATCHDOG TIMER (WDG)

#### 11.1.1 Introduction

The Watchdog timer is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The Watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the counter's contents before the T6 bit becomes cleared.

#### 11.1.2 Main Features

- Programmable free-running downcounter
- Programmable reset
- Reset (if watchdog activated) when the T6 bit reaches zero
- Optional reset on HALT instruction (configurable by option byte)
- Hardware Watchdog selectable by option byte

#### 11.1.3 Functional Description

The counter value stored in the Watchdog Control register (WDGCR bits T[6:0]), is decremented every 16384  $f_{OSC2}$  cycles (approx.), and the length of the timeout period can be programmed by the user in 64 increments.

If the watchdog is activated (the WDGA bit is set) and when the 7-bit timer (bits T[6:0]) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling low the reset pin for typically 500ns.

The application program must write in the WDGCR register at regular intervals during normal operation to prevent an MCU reset. This downcounter is free-running: it counts down even if the watchdog is disabled. The value to be stored in the WDGCR register must be between FFh and C0h:

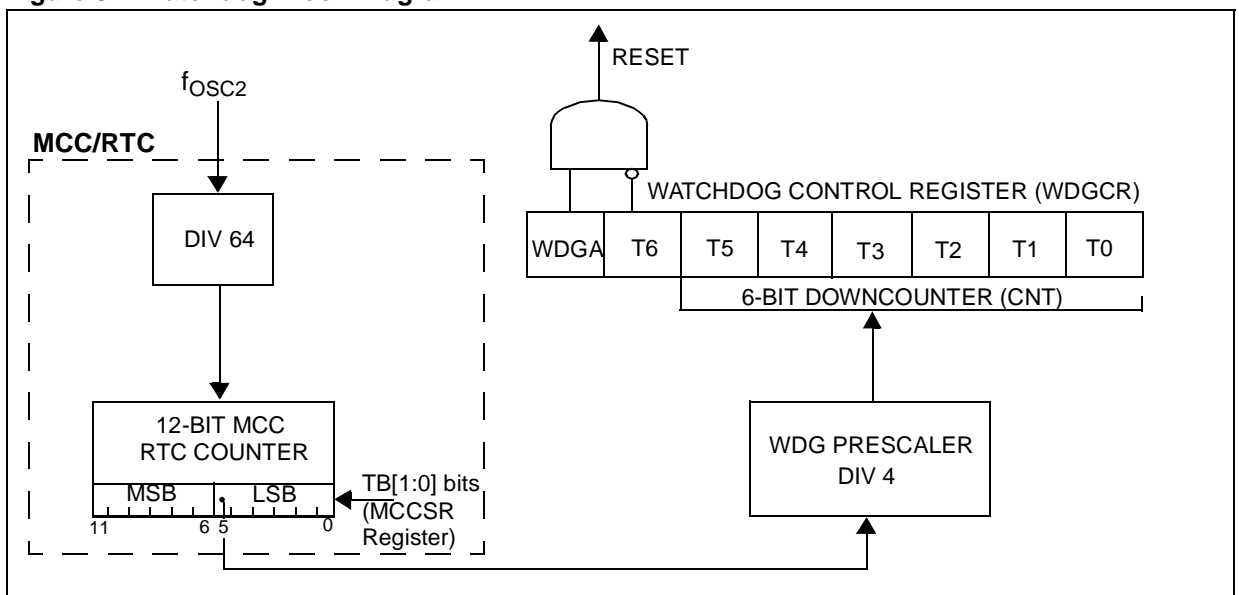
- The WDGA bit is set (watchdog enabled)
- The T6 bit is set to prevent generating an immediate reset
- The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset (see Figure 33. Approximate Timeout Duration). The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WDGCR register (see Figure 34).

Following a reset, the watchdog is disabled. Once activated it cannot be disabled, except by a reset.

The T6 bit can be used to generate a software reset (the WDGA bit is set and the T6 bit is cleared).

If the watchdog is activated, the HALT instruction will generate a Reset.

Figure 32. Watchdog Block Diagram



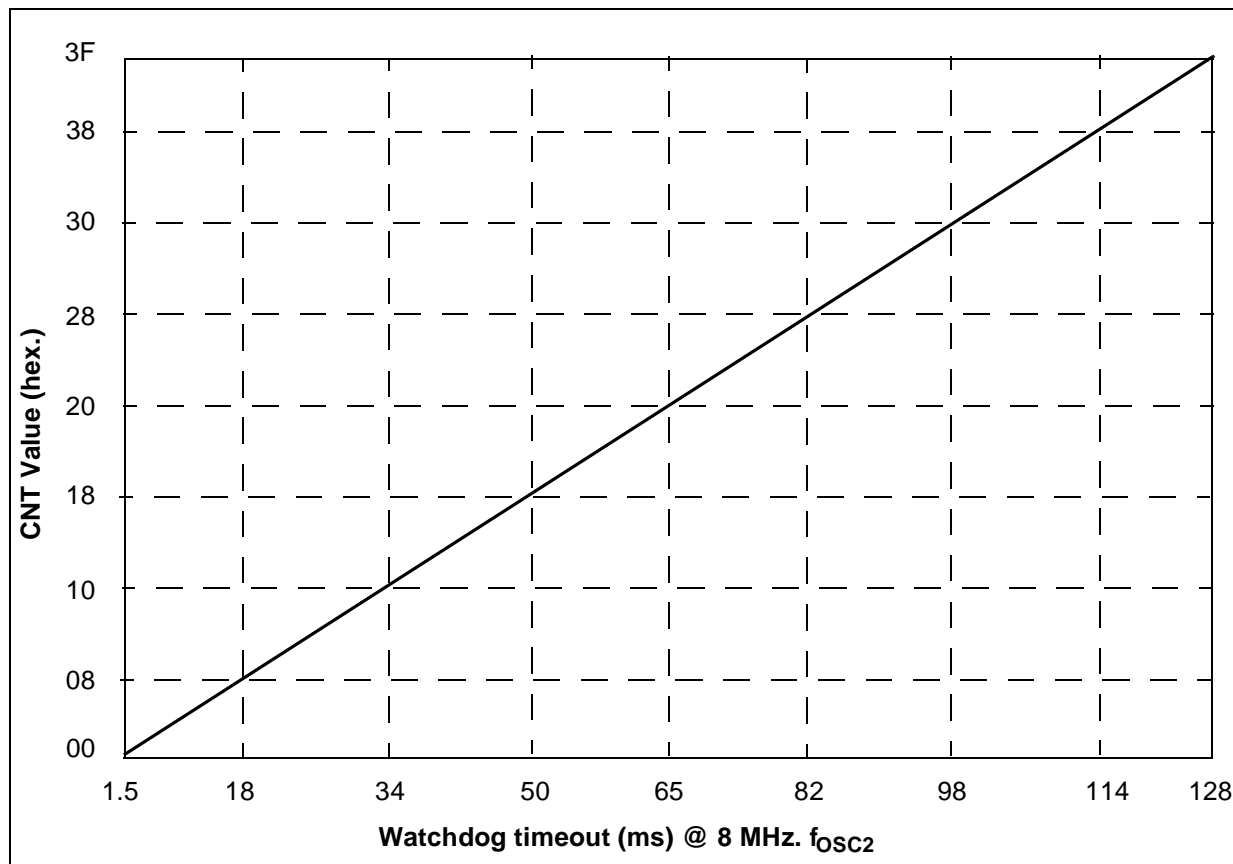


**WATCHDOG TIMER (Cont'd)****11.1.4 How to Program the Watchdog Timeout**

Figure 33 shows the linear relationship between the 6-bit value to be loaded in the Watchdog Counter (CNT) and the resulting timeout duration in milliseconds. This can be used for a quick calculation without taking the timing variations into account. If

more precision is needed, use the formulae in Figure 34.

**Caution:** When writing to the WDGCR register, always write 1 in the T6 bit to avoid generating an immediate reset.

**Figure 33. Approximate Timeout Duration**

**WATCHDOG TIMER (Cont'd)**

**Figure 34. Exact Timeout Duration ( $t_{min}$  and  $t_{max}$ )**

**WHERE:**

$$t_{min0} = (LSB + 128) \times 64 \times t_{OSC2}$$

$$t_{max0} = 16384 \times t_{OSC2}$$

$$t_{OSC2} = 125ns \text{ if } f_{OSC2}=8 \text{ MHz}$$

CNT = Value of T[5:0] bits in the WDGCR register (6 bits)

MSB and LSB are values from the table below depending on the timebase selected by the TB[1:0] bits in the MCCSR register

TB1 Bit (MCCSR Reg.)	TB0 Bit (MCCSR Reg.)	Selected MCCSR Timebase	MSB	LSB
0	0	2ms	4	59
0	1	4ms	8	53
1	0	10ms	20	35
1	1	25ms	49	54

**To calculate the minimum Watchdog Timeout ( $t_{min}$ ):**

**IF**  $CNT < \left\lceil \frac{MSB}{4} \right\rceil$  **THEN**  $t_{min} = t_{min0} + 16384 \times CNT \times t_{osc2}$

**ELSE**  $t_{min} = t_{min0} + \left[ 16384 \times \left( CNT - \left\lceil \frac{4CNT}{MSB} \right\rceil \right) + (192 + LSB) \times 64 \times \left\lceil \frac{4CNT}{MSB} \right\rceil \right] \times t_{osc2}$

**To calculate the maximum Watchdog Timeout ( $t_{max}$ ):**

**IF**  $CNT \leq \left\lceil \frac{MSB}{4} \right\rceil$  **THEN**  $t_{max} = t_{max0} + 16384 \times CNT \times t_{osc2}$

**ELSE**  $t_{max} = t_{max0} + \left[ 16384 \times \left( CNT - \left\lceil \frac{4CNT}{MSB} \right\rceil \right) + (192 + LSB) \times 64 \times \left\lceil \frac{4CNT}{MSB} \right\rceil \right] \times t_{osc2}$

**Note:** In the above formulae, division results must be rounded down to the next integer value.

**Example:**

With 2ms timeout selected in MCCSR register

Value of T[5:0] Bits in WDGCR Register (Hex.)	Min. Watchdog Timeout (ms)	Max. Watchdog Timeout (ms)
	$t_{min}$	$t_{max}$
00	1.496	2.048
3F	128	128.552

## WATCHDOG TIMER (Cont'd)

## 11.1.5 Low Power Modes

Mode	Description		
SLOW	No effect on Watchdog.		
WAIT	No effect on Watchdog.		
HALT	OIE bit in MCSR register	WDGHALT bit in Option Byte	
	0	0	No Watchdog reset is generated. The MCU enters Halt mode. The Watchdog counter is decremented once and then stops counting and is no longer able to generate a watchdog reset until the MCU receives an external interrupt or a reset. If an external interrupt is received, the Watchdog restarts counting after 256 or 4096 CPU clocks. If a reset is generated, the Watchdog is disabled (reset state) unless Hardware Watchdog is selected by option byte. For application recommendations see Section 11.1.7 below.
	0	1	A reset is generated.
	1	x	No reset is generated. The MCU enters Active Halt mode. The Watchdog counter is not decremented. It stop counting. When the MCU receives an oscillator interrupt or external interrupt, the Watchdog restarts counting immediately. When the MCU receives a reset the Watchdog restarts counting after 256 or 4096 CPU clocks.

## 11.1.6 Hardware Watchdog Option

If Hardware Watchdog is selected by option byte, the watchdog is always active and the WDGA bit in the WDGCR is not used. Refer to the Option Byte description.

## 11.1.7 Using Halt Mode with the WDG (WDGHALT option)

The following recommendation applies if Halt mode is used when the watchdog is enabled.

- Before executing the HALT instruction, refresh the WDG counter, to avoid an unexpected WDG reset immediately after waking up the microcontroller.

## 11.1.8 Interrupts

None.

## 11.1.9 Register Description

## CONTROL REGISTER (WDGCR)

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0

Bit 7 = **WDGA** Activation bit.

This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset.

0: Watchdog disabled

1: Watchdog enabled

**Note:** This bit is not used if the hardware watchdog option is enabled by option byte.

Bit 6:0 = **T[6:0]** 7-bit counter (MSB to LSB).

These bits contain the value of the watchdog counter. It is decremented every  $16384 f_{OSC2}$  cycles (approx.). A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

Table 12. Watchdog Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0024h	<b>WDGCR</b> Reset Value	WDGA 0	T6 1	T5 1	T4 1	T3 1	T2 1	T1 1	T0 1

## 11.2 MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK (MCC/RTC)

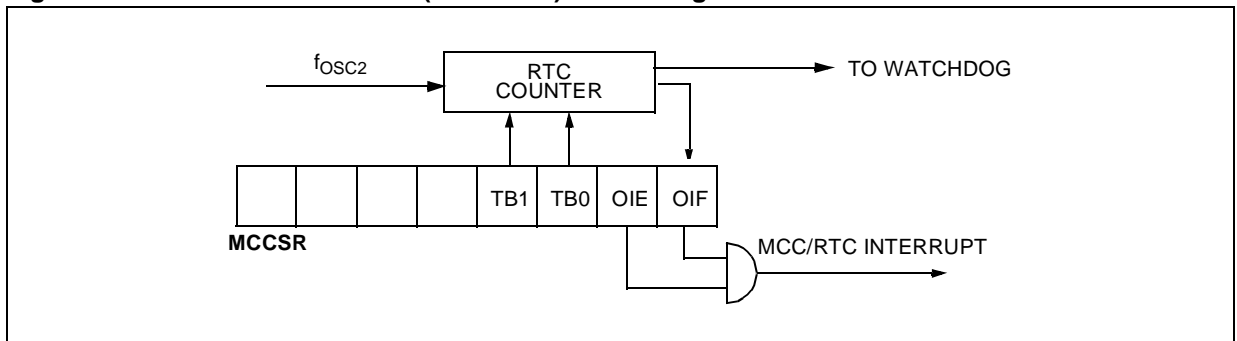
The Main Clock Controller consists of a real time clock timer with interrupt capability

### 11.2.1 Real Time Clock Timer (RTC)

The counter of the real time clock timer allows an interrupt to be generated based on an accurate real time clock. Four different time bases depending directly on  $f_{OSC2}$  are available. The whole functionality is controlled by four bits of the MCC-SR register: TB[1:0], OIE and OIF.

When the RTC interrupt is enabled (OIE bit set), the ST7 enters ACTIVE-HALT mode when the HALT instruction is executed. See Section 8.4 "ACTIVE-HALT AND HALT MODES" on page 35 for more details.

**Figure 35. Main Clock Controller (MCC/RTC) Block Diagram**



**MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK (Cont'd)**

**11.2.2 Low Power Modes**

Mode	Description
WAIT	No effect on MCC/RTC peripheral. MCC/RTC interrupt cause the device to exit from WAIT mode.
ACTIVE-HALT	No effect on MCC/RTC counter (OIE bit is set), the registers are frozen. MCC/RTC interrupt cause the device to exit from ACTIVE-HALT mode.
HALT	MCC/RTC counter and registers are frozen. MCC/RTC operation resumes when the MCU is woken up by an interrupt with "exit from HALT" capability.

**11.2.3 Interrupts**

The MCC/RTC interrupt event generates an interrupt if the OIE bit of the MCCR register is set and the interrupt mask in the CC register is not active (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Time base overflow event	OIF	OIE	Yes	No <sup>1)</sup>

**Note:**

The MCC/RTC interrupt wakes up the MCU from ACTIVE-HALT mode, not from HALT mode.

**11.2.4 Register Description**

**MCC CONTROL/STATUS REGISTER (MCCR)**

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	TB1	TB0	OIE	OIF

**Table 13. Main Clock Controller Register Map and Reset Values**

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0025h	SICSR Reset Value	VDS 0	VDIE 0	VDF 0	LVDRF x	0	CFIE 0	CSSD 0	WDGRF x
0026h	MCCR Reset Value	0	0	0	0	TB1 0	TB0 0	OIE 0	OIF 0

Bit 7:4 = reserved

Bit 3:2 = **TB[1:0]** Time base control

These bits select the programmable divider time base. They are set and cleared by software.

Counter Prescaler	Time Base		TB1	TB0
	f <sub>osc2</sub> =4MHz	f <sub>osc2</sub> =8MHz		
16000	4ms	2ms	0	0
32000	8ms	4ms	0	1
80000	20ms	10ms	1	0
200000	50ms	25ms	1	1

A modification of the time base is taken into account at the end of the current period (previously set) to avoid an unwanted time shift. This allows to use this time base as a real time clock.

Bit 1 = **OIE** Oscillator interrupt enable

This bit set and cleared by software.

0: Oscillator interrupt disabled

1: Oscillator interrupt enabled

This interrupt can be used to exit from ACTIVE-HALT mode.

When this bit is set, calling the ST7 software HALT instruction enters the ACTIVE-HALT power saving mode. MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK (Cont'd)

Bit 0 = **OIF** Oscillator interrupt flag

This bit is set by hardware and cleared by software reading the CSR register. It indicates when set that the main oscillator has reached the selected elapsed time (TB1:0).

0: Timeout not reached

1: Timeout reached

**CAUTION:** The BRES and BSET instructions must not be used on the MCCR register to avoid unintentionally clearing the OIF bit.

## 11.3 16-BIT TIMER

### 11.3.1 Introduction

The timer consists of a 16-bit free-running counter driven by a programmable prescaler.

It may be used for a variety of purposes, including pulse length measurement of up to two input signals (*input capture*) or generation of up to two output waveforms (*output compare* and *PWM*).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the CPU clock prescaler.

Some ST7 devices have two on-chip 16-bit timers. They are completely independent, and do not share any resources. They are synchronized after a MCU reset as long as the timer clock frequencies are not modified.

This description covers one or two 16-bit timers. In ST7 devices with two timers, register names are prefixed with TA (Timer A) or TB (Timer B).

### 11.3.2 Main Features

- Programmable prescaler:  $f_{CPU}$  divided by 2, 4 or 8.
- Overflow status flag and maskable interrupt
- External clock input (must be at least 4 times slower than the CPU clock speed) with the choice of active edge
- 1 or 2 Output Compare functions each with:
  - 2 dedicated 16-bit registers
  - 2 dedicated programmable signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- 1 or 2 Input Capture functions each with:
  - 2 dedicated 16-bit registers
  - 2 dedicated active edge selection signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- Pulse width modulation mode (PWM)
- One pulse mode
- Reduced Power Mode
- 5 alternate functions on I/O ports (ICAP1, ICAP2, OCMP1, OCMP2, EXTCLK)\*

The Block Diagram is shown in Figure 36.

**\*Note:** Some timer pins may not be available (not bonded) in some ST7 devices. Refer to the device pin out description.

When reading an input signal on a non-bonded pin, the value will always be '1'.

### 11.3.3 Functional Description

#### 11.3.3.1 Counter

The main block of the Programmable Timer is a 16-bit free running upcounter and its associated 16-bit registers. The 16-bit registers are made up of two 8-bit registers called high & low.

Counter Register (CR):

- Counter High Register (CHR) is the most significant byte (MS Byte).
- Counter Low Register (CLR) is the least significant byte (LS Byte).

Alternate Counter Register (ACR)

- Alternate Counter High Register (ACHR) is the most significant byte (MS Byte).
- Alternate Counter Low Register (ACLR) is the least significant byte (LS Byte).

These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (Timer overflow flag), located in the Status register, (SR), (see note at the end of paragraph titled 16-bit read sequence).

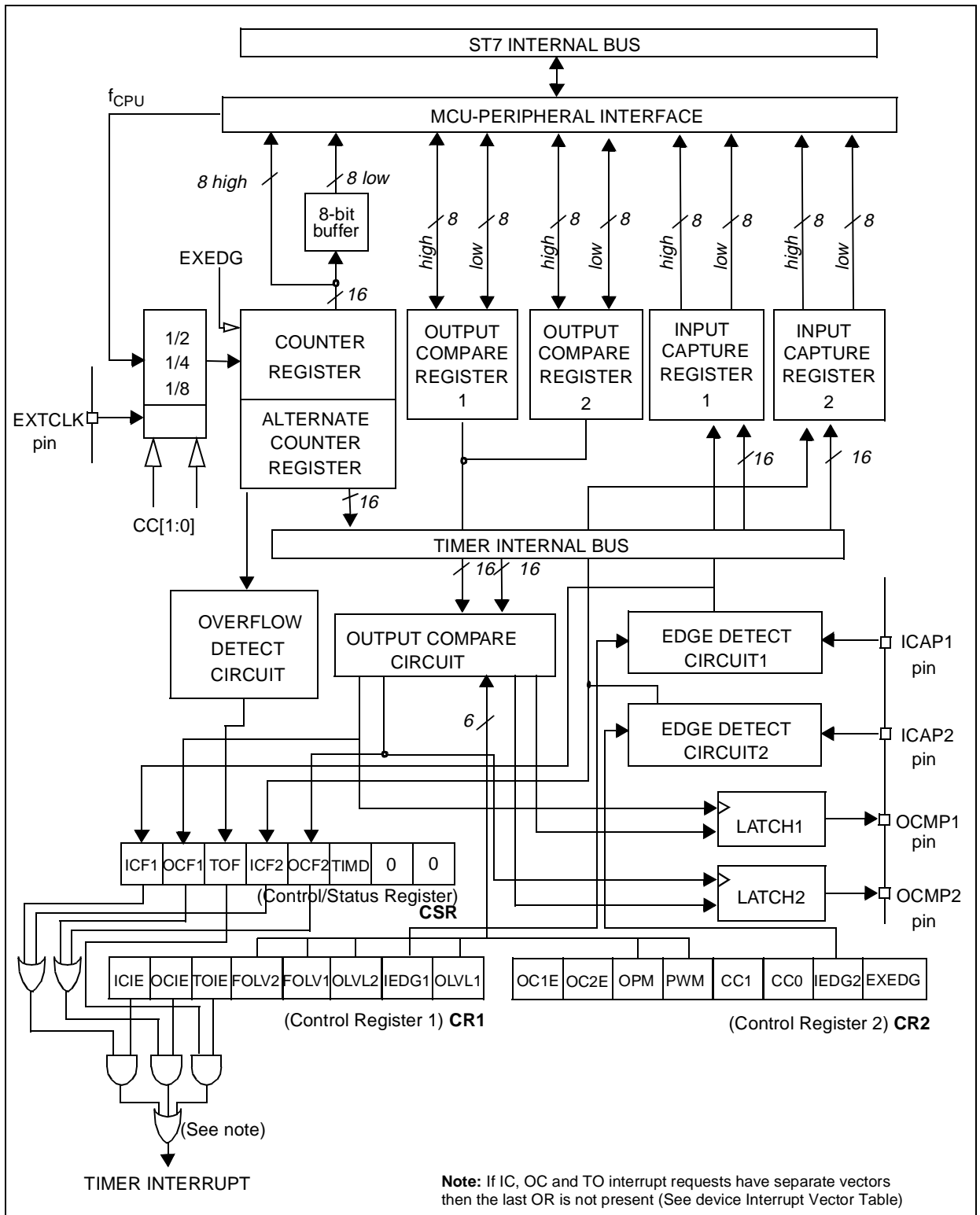
Writing in the CLR register or ACLR register resets the free running counter to the FFFCh value. Both counters have a reset value of FFFCh (this is the only value which is reloaded in the 16-bit timer). The reset value of both counters is also FFFCh in One Pulse mode and PWM mode.

The timer clock depends on the clock control bits of the CR2 register, as illustrated in Table 14 Clock Control Bits. The value in the counter register repeats every 131072, 262144 or 524288 CPU clock cycles depending on the CC[1:0] bits.

The timer frequency can be  $f_{CPU}/2$ ,  $f_{CPU}/4$ ,  $f_{CPU}/8$  or an external frequency.

16-BIT TIMER (Cont'd)

Figure 36. Timer Block Diagram

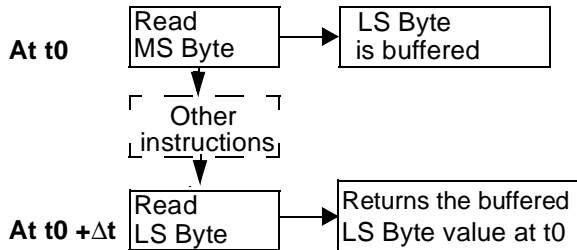




**16-BIT TIMER** (Cont'd)

**16-bit read sequence:** (from either the Counter Register or the Alternate Counter Register).

*Beginning of the sequence*



*Sequence completed*

The user must read the MS Byte first, then the LS Byte value is buffered automatically.

This buffered value remains unchanged until the 16-bit read sequence is completed, even if the user reads the MS Byte several times.

After a complete reading sequence, if only the CLR register or ACLR register are read, they return the LS Byte of the count value at the time of the read.

Whatever the timer mode used (input capture, output compare, one pulse mode or PWM mode) an overflow occurs when the counter rolls over from FFFFh to 0000h then:

- The TOF bit of the SR register is set.
- A timer interrupt is generated if:
  - TOIE bit of the CR1 register is set and
  - I bit of the CC register is cleared.

If one of these conditions is false, the interrupt remains pending to be issued as soon as they are both true.

Clearing the overflow interrupt request is done in two steps:

1. Reading the SR register while the TOF bit is set.
2. An access (read or write) to the CLR register.

**Notes:** The TOF bit is not cleared by accesses to ACLR register. The advantage of accessing the ACLR register rather than the CLR register is that it allows simultaneous use of the overflow function and reading the free running counter at random times (for example, to measure elapsed time) without the risk of clearing the TOF bit erroneously.

The timer is not affected by WAIT mode.

In HALT mode, the counter stops counting until the mode is exited. Counting then resumes from the previous count (MCU awakened by an interrupt) or from the reset count (MCU awakened by a Reset).

**11.3.3.2 External Clock**

The external clock (where available) is selected if CC0=1 and CC1=1 in the CR2 register.

The status of the EXEDG bit in the CR2 register determines the type of level transition on the external clock pin EXTCLK that will trigger the free running counter.

The counter is synchronized with the falling edge of the internal CPU clock.

A minimum of four falling edges of the CPU clock must occur between two consecutive active edges of the external clock; thus the external clock frequency must be less than a quarter of the CPU clock frequency.

16-BIT TIMER (Cont'd)

Figure 37. Counter Timing Diagram, internal clock divided by 2

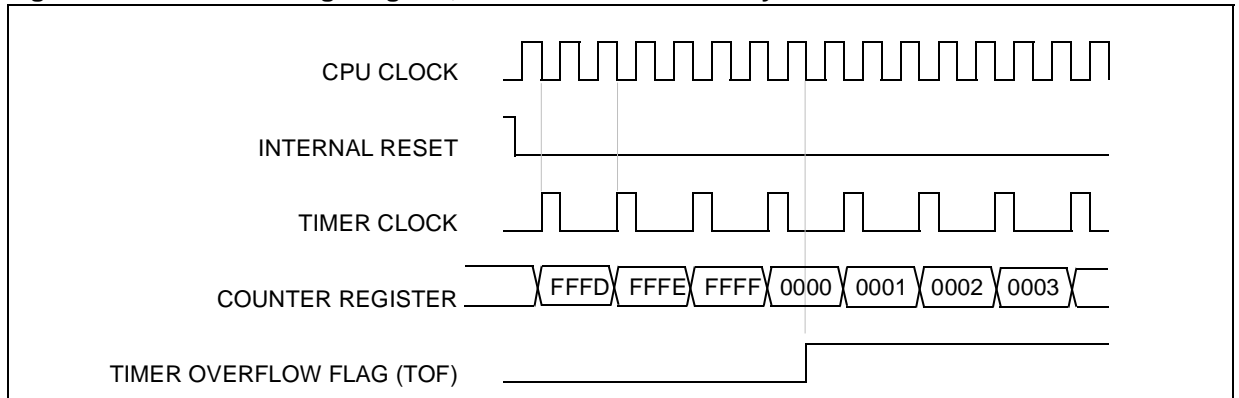


Figure 38. Counter Timing Diagram, internal clock divided by 4

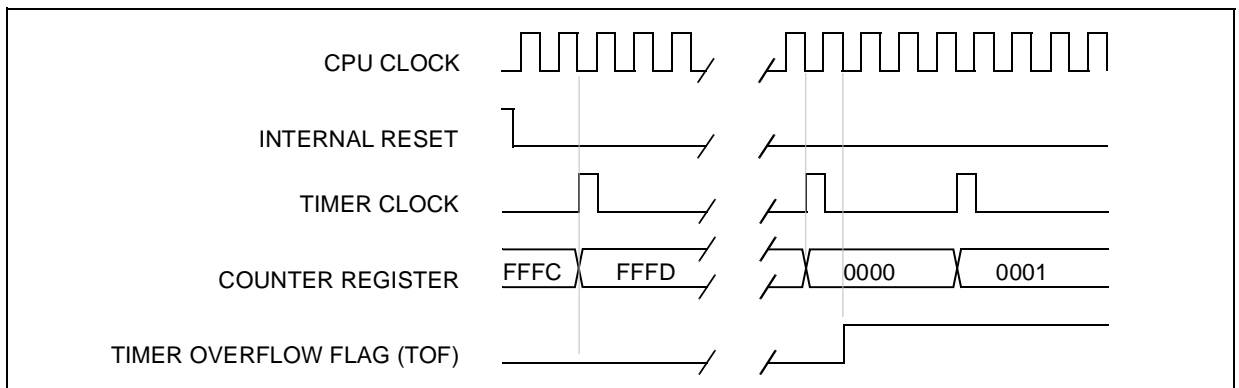
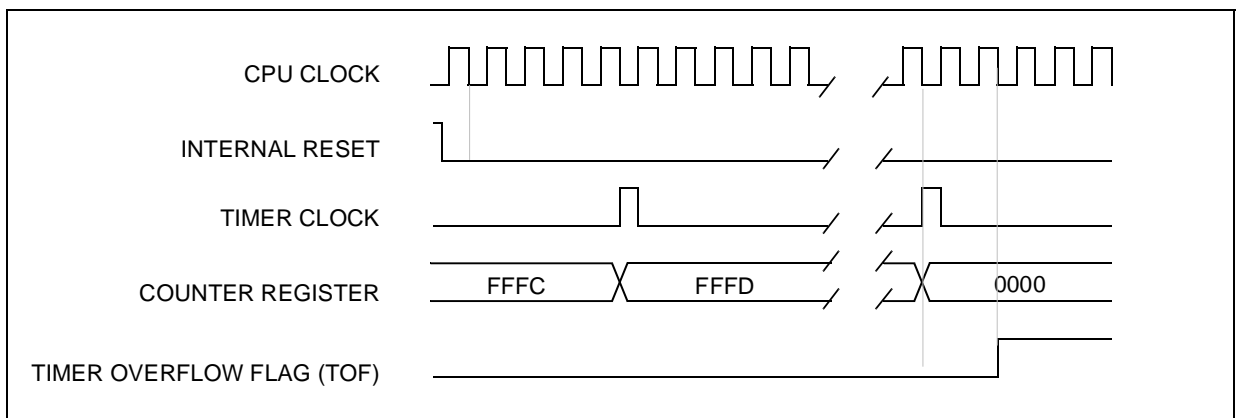


Figure 39. Counter Timing Diagram, internal clock divided by 8

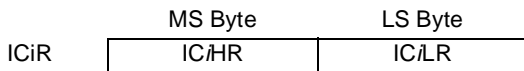


**Note:** The MCU is in reset state when the internal reset signal is high, when it is low the MCU is running.

**16-BIT TIMER (Cont'd)****11.3.3.3 Input Capture**

In this section, the index,  $i$ , may be 1 or 2 because there are 2 input capture functions in the 16-bit timer.

The two 16-bit input capture registers (IC1R and IC2R) are used to latch the value of the free running counter after a transition is detected on the ICAP $i$  pin (see figure 5).



IC $i$ R register is a read-only register.

The active transition is software programmable through the IEDG $i$  bit of Control Registers (CR).

Timing resolution is one count of the free running counter: ( $f_{CPU}/CC[1:0]$ ).

**Procedure:**

To use the input capture function select the following in the CR2 register:

- Select the timer clock (CC[1:0]) (see Table 14 Clock Control Bits).
- Select the edge of the active transition on the ICAP2 pin with the IEDG2 bit (the ICAP2 pin must be configured as floating input or input with pull-up without interrupt if this configuration is available).

And select the following in the CR1 register:

- Set the ICIE bit to generate an interrupt after an input capture coming from either the ICAP1 pin or the ICAP2 pin
- Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1 pin must be configured as floating input or input with pull-up without interrupt if this configuration is available).

When an input capture occurs:

- ICF $i$  bit is set.
- The IC $i$ R register contains the value of the free running counter on the active transition on the ICAP $i$  pin (see Figure 41).
- A timer interrupt is generated if the ICIE bit is set and the I bit is cleared in the CC register. Otherwise, the interrupt remains pending until both conditions become true.

Clearing the Input Capture interrupt request (i.e. clearing the ICF $i$  bit) is done in two steps:

1. Reading the SR register while the ICF $i$  bit is set.
2. An access (read or write) to the IC $i$ LR register.

**Notes:**

1. After reading the IC $i$ HR register, transfer of input capture data is inhibited and ICF $i$  will never be set until the IC $i$ LR register is also read.
2. The IC $i$ R register contains the free running counter value which corresponds to the most recent input capture.
3. The 2 input capture functions can be used together even if the timer also uses the 2 output compare functions.
4. In One pulse Mode and PWM mode only Input Capture 2 can be used.
5. The alternate inputs (ICAP1 & ICAP2) are always directly connected to the timer. So any transitions on these pins activates the input capture function.  
Moreover if one of the ICAP $i$  pins is configured as an input and the second one as an output, an interrupt can be generated if the user toggles the output pin and if the ICIE bit is set. This can be avoided if the input capture function  $i$  is disabled by reading the IC $i$ HR (see note 1).
6. The TOF bit can be used with interrupt generation in order to measure events that go beyond the timer range (FFFFh).

16-BIT TIMER (Cont'd)

Figure 40. Input Capture Block Diagram

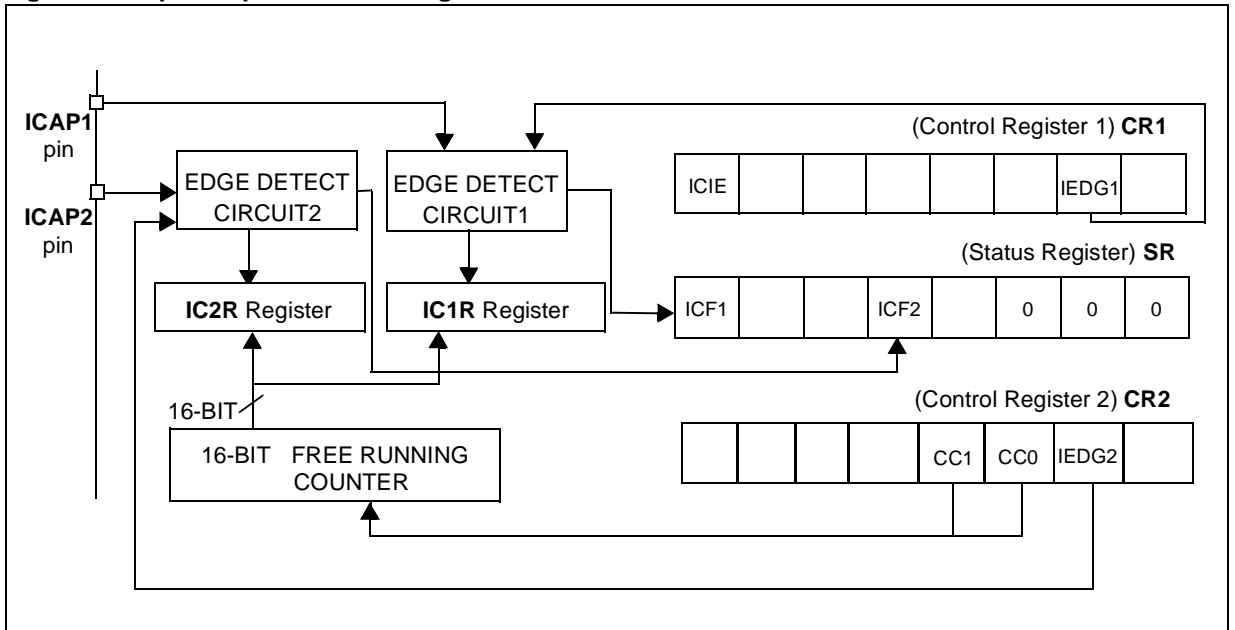
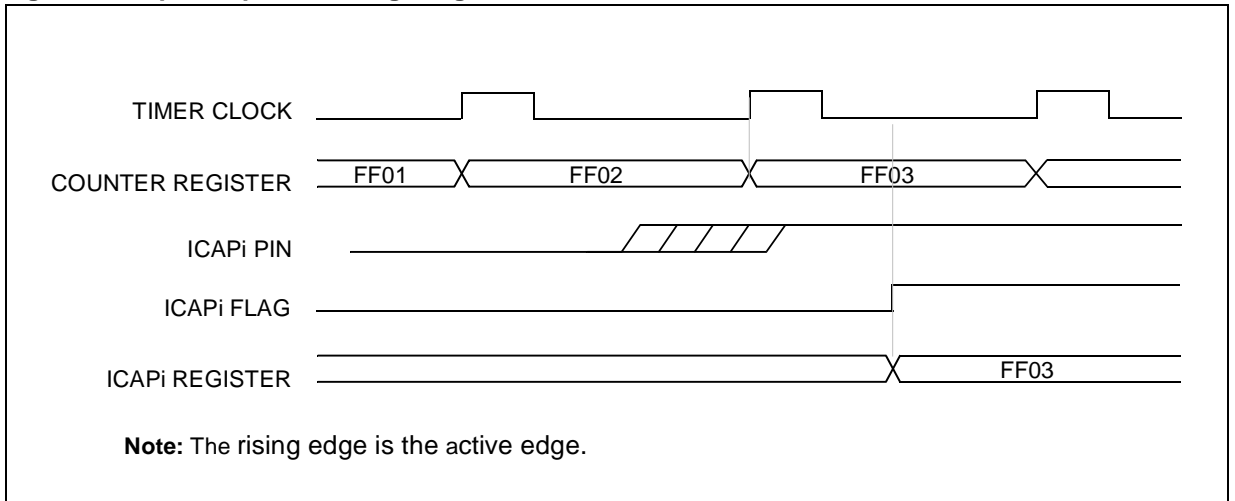


Figure 41. Input Capture Timing Diagram



**16-BIT TIMER** (Cont'd)**11.3.3.4 Output Compare**

In this section, the index,  $i$ , may be 1 or 2 because there are 2 output compare functions in the 16-bit timer.

This function can be used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the Output Compare register and the free running counter, the output compare function:

- Assigns pins with a programmable value if the OC $\bar{E}$  bit is set
- Sets a flag in the status register
- Generates an interrupt if enabled

Two 16-bit registers Output Compare Register 1 (OC1R) and Output Compare Register 2 (OC2R) contain the value to be compared to the counter register each timer clock cycle.

	MS Byte	LS Byte
OC $\bar{R}$	OC $\bar{H}R$	OC $\bar{L}R$

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OC $\bar{R}$  value to 8000h.

Timing resolution is one count of the free running counter: ( $f_{CPU}/CC[1:0]$ ).

**Procedure:**

To use the output compare function, select the following in the CR2 register:

- Set the OC $\bar{E}$  bit if an output is needed then the OCMP $i$  pin is dedicated to the output compare  $i$  signal.
- Select the timer clock (CC[1:0]) (see Table 14 Clock Control Bits).

And select the following in the CR1 register:

- Select the OLVL $i$  bit to applied to the OCMP $i$  pins after the match occurs.
- Set the OCIE bit to generate an interrupt if it is needed.

When a match is found between OCR $i$  register and CR register:

- OCF $i$  bit is set.

- The OCMP $i$  pin takes OLVL $i$  bit value (OCMP $i$  pin latch is forced low during reset).
- A timer interrupt is generated if the OCIE bit is set in the CR1 register and the I bit is cleared in the CC register (CC).

The OC $\bar{R}$  register value required for a specific timing application can be calculated using the following formula:

$$\Delta OC\bar{R} = \frac{\Delta t * f_{CPU}}{PRESC}$$

Where:

- $\Delta t$  = Output compare period (in seconds)
- $f_{CPU}$  = CPU clock frequency (in hertz)
- PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 14 Clock Control Bits)

If the timer clock is an external clock, the formula is:

$$\Delta OC\bar{R} = \Delta t * f_{EXT}$$

Where:

- $\Delta t$  = Output compare period (in seconds)
- $f_{EXT}$  = External timer clock frequency (in hertz)

Clearing the output compare interrupt request (i.e. clearing the OCF $i$  bit) is done by:

1. Reading the SR register while the OCF $i$  bit is set.
2. An access (read or write) to the OC $\bar{L}R$  register.

The following procedure is recommended to prevent the OCF $i$  bit from being set between the time it is read and the write to the OC $\bar{R}$  register:

- Write to the OC $\bar{H}R$  register (further compares are inhibited).
- Read the SR register (first step of the clearance of the OCF $i$  bit, which may be already set).
- Write to the OC $\bar{L}R$  register (enables the output compare function and clears the OCF $i$  bit).

**16-BIT TIMER (Cont'd)**

**Notes:**

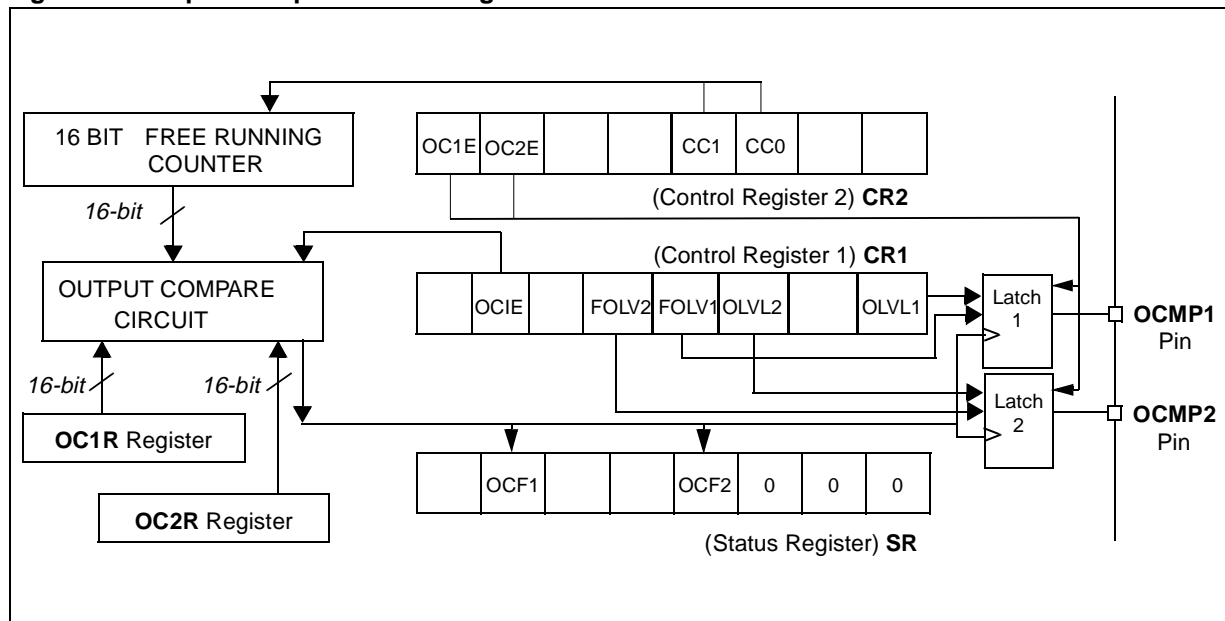
1. After a processor write cycle to the OCiHR register, the output compare function is inhibited until the OCiLR register is also written.
2. If the OCiE bit is not set, the OCMPi pin is a general I/O port and the OLVLi bit will not appear when a match is found but an interrupt could be generated if the OCIE bit is set.
3. When the timer clock is  $f_{CPU}/2$ , OCFi and OCMPi are set while the counter value equals the OCiR register value (see Figure 43 on page 63). This behaviour is the same in OPM or PWM mode.  
When the timer clock is  $f_{CPU}/4$ ,  $f_{CPU}/8$  or in external clock mode, OCFi and OCMPi are set while the counter value equals the OCiR register value plus 1 (see Figure 44 on page 63).
4. The output compare functions can be used both for generating external events on the OCMPi pins even if the input capture mode is also used.
5. The value in the 16-bit OCiR register and the OLVi bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout.

**Forced Compare Output capability**

When the FOLVi bit is set by software, the OLVLi bit is copied to the OCMPi pin. The OLVi bit has to be toggled in order to toggle the OCMPi pin when it is enabled (OCiE bit=1). The OCFi bit is then not set by hardware, and thus no interrupt request is generated.

The FOLVLi bits have no effect in both one pulse mode and PWM mode.

**Figure 42. Output Compare Block Diagram**



16-BIT TIMER (Cont'd)

Figure 43. Output Compare Timing Diagram,  $f_{TIMER} = f_{CPU}/2$

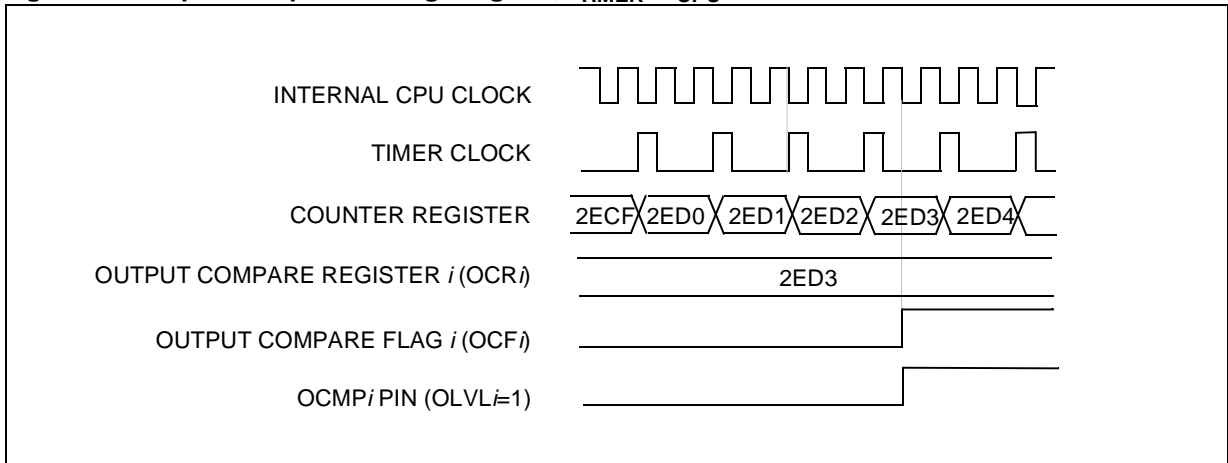
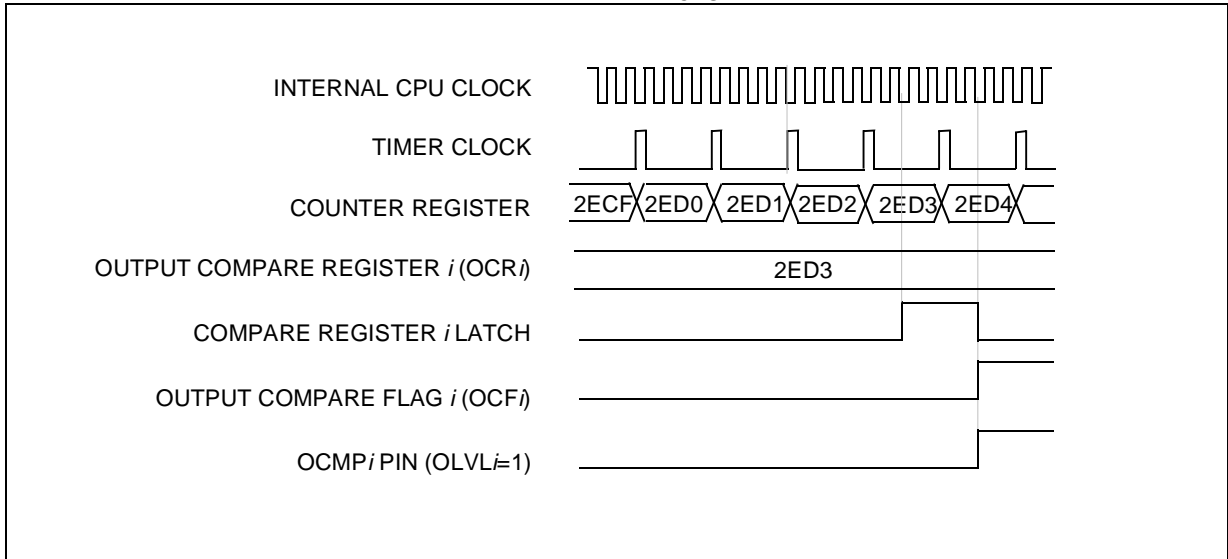


Figure 44. Output Compare Timing Diagram,  $f_{TIMER} = f_{CPU}/4$



**16-BIT TIMER (Cont'd)**

**11.3.3.5 One Pulse Mode**

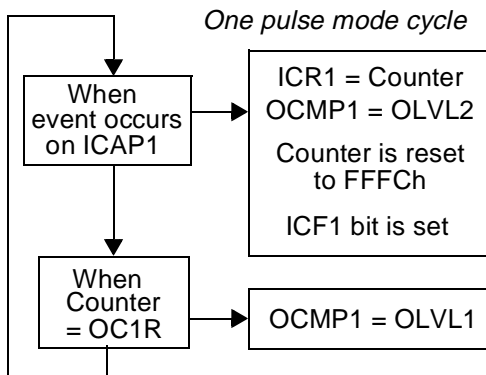
One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

The one pulse mode uses the Input Capture1 function and the Output Compare1 function.

**Procedure:**

To use one pulse mode:

1. Load the OC1R register with the value corresponding to the length of the pulse (see the formula in the opposite column).
2. Select the following in the CR1 register:
  - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after the pulse.
  - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin during the pulse.
  - Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1 pin must be configured as floating input).
3. Select the following in the CR2 register:
  - Set the OC1E bit, the OCMP1 pin is then dedicated to the Output Compare 1 function.
  - Set the OPM bit.
  - Select the timer clock CC[1:0] (see Table 14 Clock Control Bits).



Then, on a valid event on the ICAP1 pin, the counter is initialized to FFFCh and OLVL2 bit is loaded on the OCMP1 pin, the ICF1 bit is set and the value FFFDh is loaded in the IC1R register.

Because the ICF1 bit is set when an active edge occurs, an interrupt can be generated if the ICIE bit is set.

Clearing the Input Capture interrupt request (i.e. clearing the ICFi bit) is done in two steps:

1. Reading the SR register while the ICFi bit is set.
2. An access (read or write) to the ICiLR register.

The OC1R register value required for a specific timing application can be calculated using the following formula:

$$OCiR \text{ Value} = \frac{t * f_{CPU}}{PRESC} - 5$$

Where:

t = Pulse period (in seconds)

f<sub>CPU</sub> = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on the CC[1:0] bits, see Table 14 Clock Control Bits)

If the timer clock is an external clock the formula is:

$$OCiR = t * f_{EXT} - 5$$

Where:

t = Pulse period (in seconds)

f<sub>EXT</sub> = External timer clock frequency (in hertz)

When the value of the counter is equal to the value of the contents of the OC1R register, the OLVL1 bit is output on the OCMP1 pin, (See Figure 45).

**Notes:**

1. The OCF1 bit cannot be set by hardware in one pulse mode but the OCF2 bit can generate an Output Compare interrupt.
2. When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.
3. If OLVL1=OLVL2 a continuous signal will be seen on the OCMP1 pin.
4. The ICAP1 pin can not be used to perform input capture. The ICAP2 pin can be used to perform input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each time a valid edge occurs on the ICAP1 pin and ICF1 can also generates interrupt if ICIE is set.
5. When one pulse mode is used OC1R is dedicated to this mode. Nevertheless OC2R and OCF2 can be used to indicate a period of time has been elapsed but cannot generate an output waveform because the level OLVL2 is dedicated to the one pulse mode.



16-BIT TIMER (Cont'd)

Figure 45. One Pulse Mode Timing Example

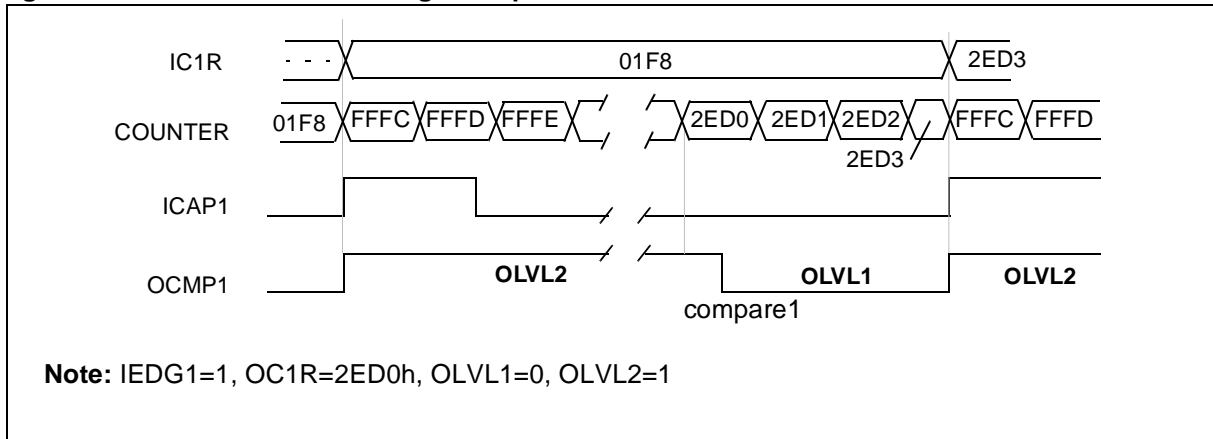
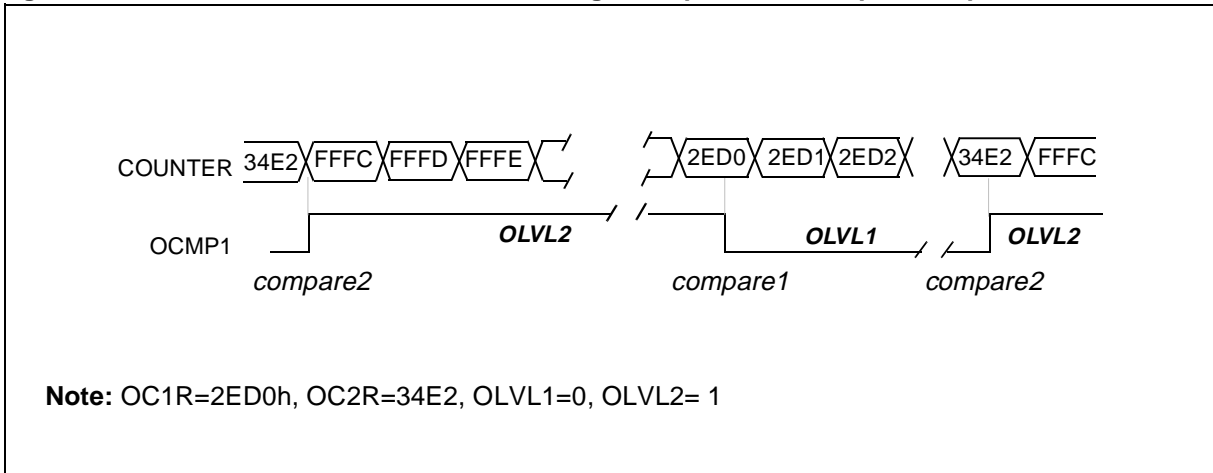


Figure 46. Pulse Width Modulation Mode Timing Example with 2 Output Compare Functions



**Note:** On timers with only 1 Output Compare register, a fixed frequency PWM signal can be generated using the output compare and the counter overflow to define the pulse length.

**16-BIT TIMER (Cont'd)**

**11.3.3.6 Pulse Width Modulation Mode**

Pulse Width Modulation (PWM) mode enables the generation of a signal with a frequency and pulse length determined by the value of the OC1R and OC2R registers.

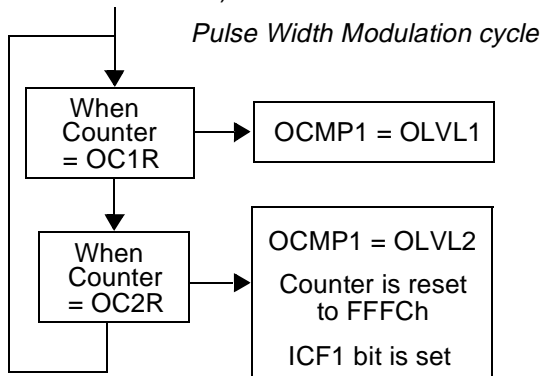
Pulse Width Modulation mode uses the complete Output Compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

In PWM mode, double buffering is implemented on the output compare registers. Any new values written in the OC1R and OC2R registers are taken into account only at the end of the PWM period (OC2) to avoid spikes on the PWM output pin (OCMP1).

**Procedure**

To use pulse width modulation mode:

1. Load the OC2R register with the value corresponding to the period of the signal using the formula in the opposite column.
2. Load the OC1R register with the value corresponding to the period of the pulse if (OLVL1=0 and OLVL2=1) using the formula in the opposite column.
3. Select the following in the CR1 register:
  - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with the OC1R register.
  - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with the OC2R register.
4. Select the following in the CR2 register:
  - Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.
  - Set the PWM bit.
  - Select the timer clock (CC[1:0]) (see Table 14 Clock Control Bits).



If OLVL1=1 and OLVL2=0 the length of the positive pulse is the difference between the OC2R and OC1R registers.

If OLVL1=OLVL2 a continuous signal will be seen on the OCMP1 pin.

The OC/R register value required for a specific timing application can be calculated using the following formula:

$$OC/R \text{ Value} = \frac{t * f_{CPU}}{PRESC} - 5$$

Where:

- t = Signal or pulse period (in seconds)
- f<sub>CPU</sub> = CPU clock frequency (in hertz)
- PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 14 Clock Control Bits)

If the timer clock is an external clock the formula is:

$$OC/R = t * f_{EXT} - 5$$

Where:

- t = Signal or pulse period (in seconds)
- f<sub>EXT</sub> = External timer clock frequency (in hertz)

The Output Compare 2 event causes the counter to be initialized to FFFCh (See Figure 46)

**Notes:**

1. After a write instruction to the OC/R register, the output compare function is inhibited until the OC/LR register is also written.
2. The OCF1 and OCF2 bits cannot be set by hardware in PWM mode therefore the Output Compare interrupt is inhibited.
3. The ICF1 bit is set by hardware when the counter reaches the OC2R value and can produce a timer interrupt if the ICIE bit is set and the I bit is cleared.
4. In PWM mode the ICAP1 pin can not be used to perform input capture because it is disconnected to the timer. The ICAP2 pin can be used to perform input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each period and ICF1 can also generates interrupt if ICIE is set.
5. When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.

**16-BIT TIMER** (Cont'd)**11.3.4 Low Power Modes**

Mode	Description
WAIT	No effect on 16-bit Timer. Timer interrupts cause the device to exit from WAIT mode.
HALT	16-bit Timer registers are frozen. In HALT mode, the counter stops counting until Halt mode is exited. Counting resumes from the previous count when the MCU is woken up by an interrupt with "exit from HALT mode" capability or from the counter reset value when the MCU is woken up by a RESET. If an input capture event occurs on the ICAP <i>i</i> pin, the input capture detection circuitry is armed. Consequently, when the MCU is woken up by an interrupt with "exit from HALT mode" capability, the ICF <i>i</i> bit is set, and the counter value present when exiting from HALT mode is captured into the IC <i>i</i> R register.

**11.3.5 Interrupts**

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Input Capture 1 event/Counter reset in PWM mode	ICF1	ICIE	Yes	No
Input Capture 2 event	ICF2		Yes	No
Output Compare 1 event (not available in PWM mode)	OCF1	OCIE	Yes	No
Output Compare 2 event (not available in PWM mode)	OCF2		Yes	No
Timer Overflow event	TOF	TOIE	Yes	No

**Note:** The 16-bit Timer interrupt events are connected to the same interrupt vector (see Interrupts chapter). These events generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

**11.3.6 Summary of Timer modes**

MODES	TIMER RESOURCES			
	Input Capture 1	Input Capture 2	Output Compare 1	Output Compare 2
Input Capture (1 and/or 2)	Yes	Yes	Yes	Yes
Output Compare (1 and/or 2)	Yes	Yes	Yes	Yes
One Pulse Mode	No	Not Recommended <sup>1)</sup>	No	Partially <sup>2)</sup>
PWM Mode	No	Not Recommended <sup>3)</sup>	No	No

1) See note 4 in Section 11.3.3.5 "One Pulse Mode" on page 64

2) See note 5 in Section 11.3.3.5 "One Pulse Mode" on page 64

3) See note 4 in Section 11.3.3.6 "Pulse Width Modulation Mode" on page 66

**16-BIT TIMER** (Cont'd)

**11.3.7 Register Description**

Each Timer is associated with three control and status registers, and with six pairs of data registers (16-bit values) relating to the two input captures, the two output compares, the counter and the alternate counter.

**CONTROL REGISTER 1 (CR1)**

Read/Write

Reset Value: 0000 0000 (00h)

7							0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1

Bit 7 = **ICIE** *Input Capture Interrupt Enable*.  
 0: Interrupt is inhibited.  
 1: A timer interrupt is generated whenever the ICF1 or ICF2 bit of the SR register is set.

Bit 6 = **OCIE** *Output Compare Interrupt Enable*.  
 0: Interrupt is inhibited.  
 1: A timer interrupt is generated whenever the OCF1 or OCF2 bit of the SR register is set.

Bit 5 = **TOIE** *Timer Overflow Interrupt Enable*.  
 0: Interrupt is inhibited.  
 1: A timer interrupt is enabled whenever the TOF bit of the SR register is set.

Bit 4 = **FOLV2** *Forced Output Compare 2*.  
 This bit is set and cleared by software.  
 0: No effect on the OCMP2 pin.  
 1: Forces the OLVL2 bit to be copied to the OCMP2 pin, if the OC2E bit is set and even if there is no successful comparison.

Bit 3 = **FOLV1** *Forced Output Compare 1*.  
 This bit is set and cleared by software.  
 0: No effect on the OCMP1 pin.  
 1: Forces OLVL1 to be copied to the OCMP1 pin, if the OC1E bit is set and even if there is no successful comparison.

Bit 2 = **OLVL2** *Output Level 2*.  
 This bit is copied to the OCMP2 pin whenever a successful comparison occurs with the OC2R register and OCxE is set in the CR2 register. This value is copied to the OCMP1 pin in One Pulse Mode and Pulse Width Modulation mode.

Bit 1 = **IEDG1** *Input Edge 1*.  
 This bit determines which type of level transition on the ICAP1 pin will trigger the capture.  
 0: A falling edge triggers the capture.  
 1: A rising edge triggers the capture.

Bit 0 = **OLVL1** *Output Level 1*.  
 The OLVL1 bit is copied to the OCMP1 pin whenever a successful comparison occurs with the OC1R register and the OC1E bit is set in the CR2 register.

**16-BIT TIMER** (Cont'd)**CONTROL REGISTER 2 (CR2)**

Read/Write

Reset Value: 0000 0000 (00h)

7							0
OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG

Bit 7 = **OC1E** *Output Compare 1 Pin Enable*.

This bit is used only to output the signal from the timer on the OCMP1 pin (OLV1 in Output Compare mode, both OLV1 and OLV2 in PWM and one-pulse mode). Whatever the value of the OC1E bit, the Output Compare 1 function of the timer remains active.

0: OCMP1 pin alternate function disabled (I/O pin free for general-purpose I/O).

1: OCMP1 pin alternate function enabled.

Bit 6 = **OC2E** *Output Compare 2 Pin Enable*.

This bit is used only to output the signal from the timer on the OCMP2 pin (OLV2 in Output Compare mode). Whatever the value of the OC2E bit, the Output Compare 2 function of the timer remains active.

0: OCMP2 pin alternate function disabled (I/O pin free for general-purpose I/O).

1: OCMP2 pin alternate function enabled.

Bit 5 = **OPM** *One Pulse Mode*.

0: One Pulse Mode is not active.

1: One Pulse Mode is active, the ICAP1 pin can be used to trigger one pulse on the OCMP1 pin; the active transition is given by the IEDG1 bit. The length of the generated pulse depends on the contents of the OC1R register.

Bit 4 = **PWM** *Pulse Width Modulation*.

0: PWM mode is not active.

1: PWM mode is active, the OCMP1 pin outputs a programmable cyclic signal; the length of the pulse depends on the value of OC1R register; the period depends on the value of OC2R register.

Bit 3, 2 = **CC[1:0]** *Clock Control*.

The timer clock mode depends on these bits:

**Table 14. Clock Control Bits**

Timer Clock	CC1	CC0
$f_{\text{CPU}} / 4$	0	0
$f_{\text{CPU}} / 2$	0	1
$f_{\text{CPU}} / 8$	1	0
External Clock (where available)	1	1

**Note:** If the external clock pin is not available, programming the external clock configuration stops the counter.

Bit 1 = **IEDG2** *Input Edge 2*.

This bit determines which type of level transition on the ICAP2 pin will trigger the capture.

0: A falling edge triggers the capture.

1: A rising edge triggers the capture.

Bit 0 = **EXEDG** *External Clock Edge*.

This bit determines which type of level transition on the external clock pin EXTCLK will trigger the counter register.

0: A falling edge triggers the counter register.

1: A rising edge triggers the counter register.

**16-BIT TIMER** (Cont'd)

**CONTROL/STATUS REGISTER (CSR)**

Read Only (except bit 2 R/W)

Reset Value: xxxx x0xx (xxh)

7						0	0
ICF1	OCF1	TOF	ICF2	OCF2	TIMD	0	0

Bit 7 = **ICF1** *Input Capture Flag 1.*

0: No input capture (reset value).

1: An input capture has occurred on the ICAP1 pin or the counter has reached the OC2R value in PWM mode. To clear this bit, first read the SR register, then read or write the low byte of the IC1R (IC1LR) register.

Bit 6 = **OCF1** *Output Compare Flag 1.*

0: No match (reset value).

1: The content of the free running counter has matched the content of the OC1R register. To clear this bit, first read the SR register, then read or write the low byte of the OC1R (OC1LR) register.

Bit 5 = **TOF** *Timer Overflow Flag.*

0: No timer overflow (reset value).

1: The free running counter rolled over from FFFFh to 0000h. To clear this bit, first read the SR register, then read or write the low byte of the CR (CLR) register.

**Note:** Reading or writing the ACLR register does not clear TOF.

Bit 4 = **ICF2** *Input Capture Flag 2.*

0: No input capture (reset value).

1: An input capture has occurred on the ICAP2 pin. To clear this bit, first read the SR register, then read or write the low byte of the IC2R (IC2LR) register.

Bit 3 = **OCF2** *Output Compare Flag 2.*

0: No match (reset value).

1: The content of the free running counter has matched the content of the OC2R register. To clear this bit, first read the SR register, then read or write the low byte of the OC2R (OC2LR) register.

Bit 2 = **TIMD** *Timer disable.*

This bit is set and cleared by software. When set, it freezes the timer prescaler and counter and disabled the output functions (OCMP1 and OCMP2 pins) to reduce power consumption. Access to the timer registers is still available, allowing the timer configuration to be changed, or the counter reset, while it is disabled.

0: Timer enabled

1: Timer prescaler, counter and outputs disabled

Bits 1:0 = Reserved, must be kept cleared.

**16-BIT TIMER (Cont'd)****INPUT CAPTURE 1 HIGH REGISTER (IC1HR)**

Read Only

Reset Value: Undefined

This is an 8-bit read only register that contains the high part of the counter value (transferred by the input capture 1 event).

**OUTPUT COMPARE 1 HIGH REGISTER (OC1HR)**

Read/Write

Reset Value: 1000 0000 (80h)

This is an 8-bit register that contains the high part of the value to be compared to the CHR register.

**INPUT CAPTURE 1 LOW REGISTER (IC1LR)**

Read Only

Reset Value: Undefined

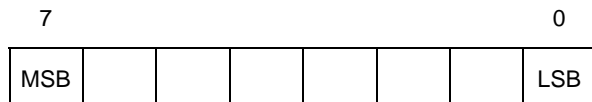
This is an 8-bit read only register that contains the low part of the counter value (transferred by the input capture 1 event).

**OUTPUT COMPARE 1 LOW REGISTER (OC1LR)**

Read/Write

Reset Value: 0000 0000 (00h)

This is an 8-bit register that contains the low part of the value to be compared to the CLR register.



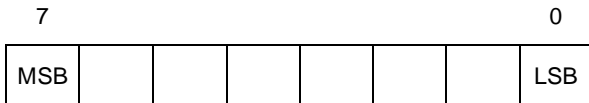
**16-BIT TIMER (Cont'd)**

**OUTPUT COMPARE 2 HIGH REGISTER (OC2HR)**

Read/Write

Reset Value: 1000 0000 (80h)

This is an 8-bit register that contains the high part of the value to be compared to the CHR register.



**OUTPUT COMPARE 2 LOW REGISTER (OC2LR)**

Read/Write

Reset Value: 0000 0000 (00h)

This is an 8-bit register that contains the low part of the value to be compared to the CLR register.

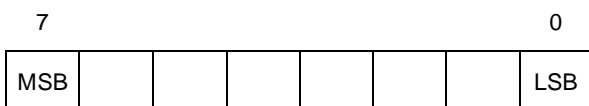


**COUNTER HIGH REGISTER (CHR)**

Read Only

Reset Value: 1111 1111 (FFh)

This is an 8-bit register that contains the high part of the counter value.

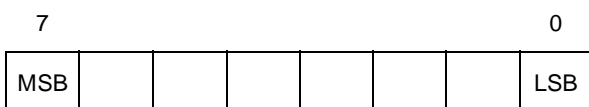


**COUNTER LOW REGISTER (CLR)**

Read Only

Reset Value: 1111 1100 (FCh)

This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after accessing the CSR register clears the TOF bit.



**ALTERNATE COUNTER HIGH REGISTER (ACHR)**

Read Only

Reset Value: 1111 1111 (FFh)

This is an 8-bit register that contains the high part of the counter value.



**ALTERNATE COUNTER LOW REGISTER (ACLRL)**

Read Only

Reset Value: 1111 1100 (FCh)

This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after an access to CSR register does not clear the TOF bit in the CSR register.



**INPUT CAPTURE 2 HIGH REGISTER (IC2HR)**

Read Only

Reset Value: Undefined

This is an 8-bit read only register that contains the high part of the counter value (transferred by the Input Capture 2 event).



**INPUT CAPTURE 2 LOW REGISTER (IC2LR)**

Read Only

Reset Value: Undefined

This is an 8-bit read only register that contains the low part of the counter value (transferred by the Input Capture 2 event).





## 16-BIT TIMER (Cont'd)

Table 15. 16-Bit Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Timer A: 32 Timer B: 42	<b>CR1</b> Reset Value	ICIE 0	OCIE 0	TOIE 0	FOLV2 0	FOLV1 0	OLVL2 0	IEDG1 0	OLVL1 0
Timer A: 31 Timer B: 41	<b>CR2</b> Reset Value	OC1E 0	OC2E 0	OPM 0	PWM 0	CC1 0	CC0 0	IEDG2 0	EXEDG 0
Timer A: 33 Timer B: 43	<b>CSR</b> Reset Value	ICF1 x	OCF1 x	TOF x	ICF2 x	OCF2 x	TIMD 0	- x	- x
Timer A: 34 Timer B: 44	<b>IC1HR</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 35 Timer B: 45	<b>IC1LR</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 36 Timer B: 46	<b>OC1HR</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 37 Timer B: 47	<b>OC1LR</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3E Timer B: 4E	<b>OC2HR</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3F Timer B: 4F	<b>OC2LR</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 38 Timer B: 48	<b>CHR</b> Reset Value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 39 Timer B: 49	<b>CLR</b> Reset Value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3A Timer B: 4A	<b>ACHR</b> Reset Value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 3B Timer B: 4B	<b>ACLRL</b> Reset Value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3C Timer B: 4C	<b>ICHR2</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3D Timer B: 4D	<b>ICLR2</b> Reset Value	MSB -	-	-	-	-	-	-	LSB -

### 11.4 SERIAL PERIPHERAL INTERFACE (SPI)

#### 11.4.1 Introduction

The Serial Peripheral Interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves or a system in which devices may be either masters or slaves.

#### 11.4.2 Main Features

- Full duplex synchronous transfers (on 3 lines)
- Simplex synchronous transfers (on 2 lines)
- Master or slave operation
- Six master mode frequencies ( $f_{CPU}/4$  max.)
- $f_{CPU}/2$  max. slave mode frequency
- $\overline{SS}$  Management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, Master Mode Fault and Overrun flags

#### 11.4.3 General Description

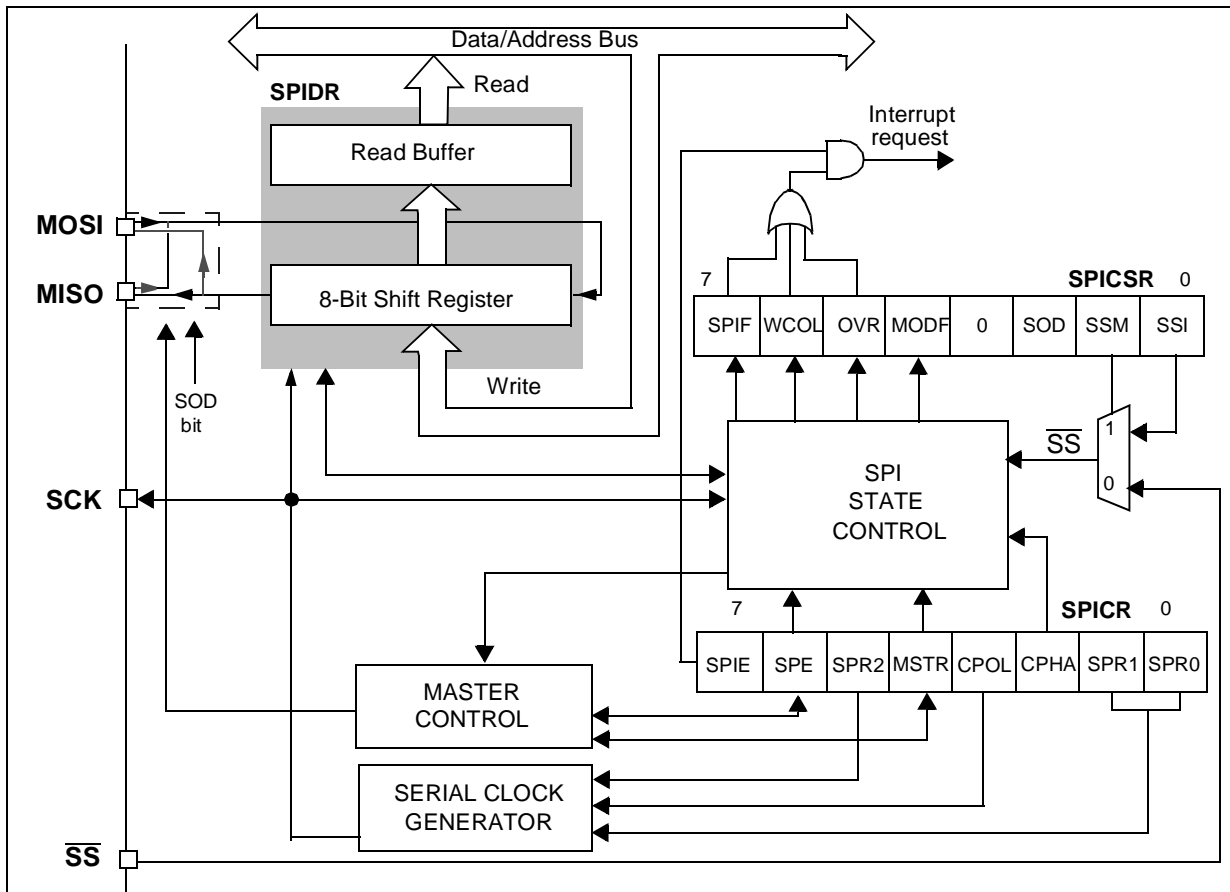
Figure 47 shows the serial peripheral interface (SPI) block diagram. There are 3 registers:

- SPI Control Register (SPICR)
- SPI Control/Status Register (SPICSR)
- SPI Data Register (SPIDR)

The SPI is connected to external devices through 3 pins:

- MISO: Master In / Slave Out data
- MOSI: Master Out / Slave In data
- SCK: Serial Clock out by SPI masters and input by SPI slaves
- $\overline{SS}$ : Slave select:  
This input signal acts as a 'chip select' to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave  $\overline{SS}$  inputs can be driven by standard I/O ports on the master Device.

Figure 47. Serial Peripheral Interface Block Diagram



**SERIAL PERIPHERAL INTERFACE (Cont'd)****11.4.3.1 Functional Description**

A basic example of interconnections between a single master and a single slave is illustrated in Figure 48.

The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

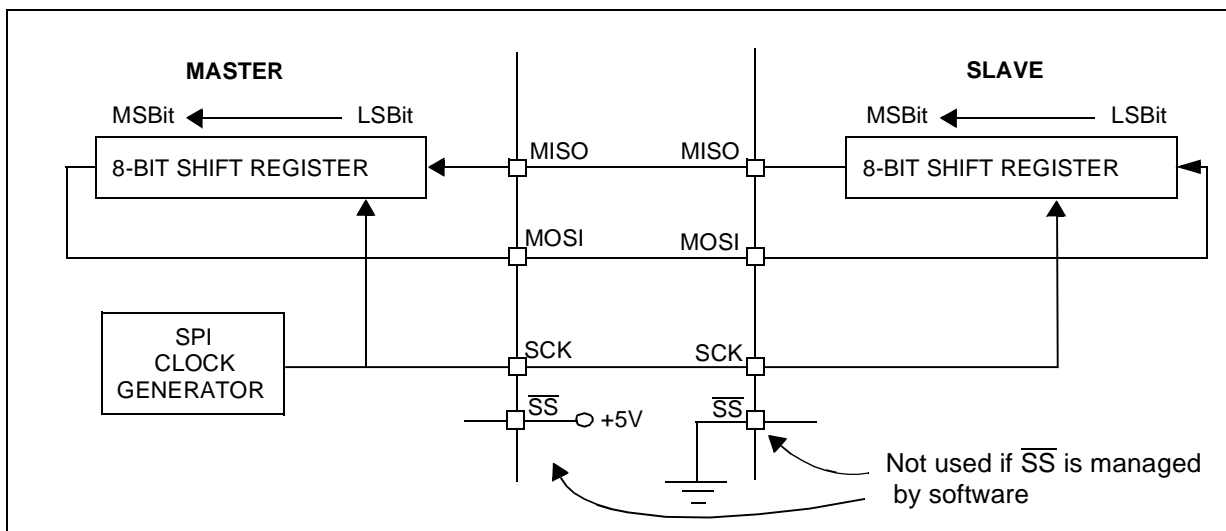
The communication is always initiated by the master. When the master device transmits data to a slave device via MOSI pin, the slave device re-

sponds by sending data to the master device via the MISO pin. This implies full duplex communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

To use a single data line, the MISO and MOSI pins must be connected at each node (in this case only simplex communication is possible).

Four possible data/clock timing relationships may be chosen (see Figure 51) but master and slave must be programmed with the same timing mode.

**Figure 48. Single Master/ Single Slave Application**



**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**11.4.3.2 Slave Select Management**

As an alternative to using the  $\overline{SS}$  pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 50)

In software management, the external  $\overline{SS}$  pin is free for other application uses and the internal  $\overline{SS}$  signal level is driven by writing to the SSI bit in the SPICSR register.

**In Master mode:**

- $\overline{SS}$  internal must be held high continuously

**In Slave Mode:**

There are two cases depending on the data/clock timing relationship (see Figure 49):

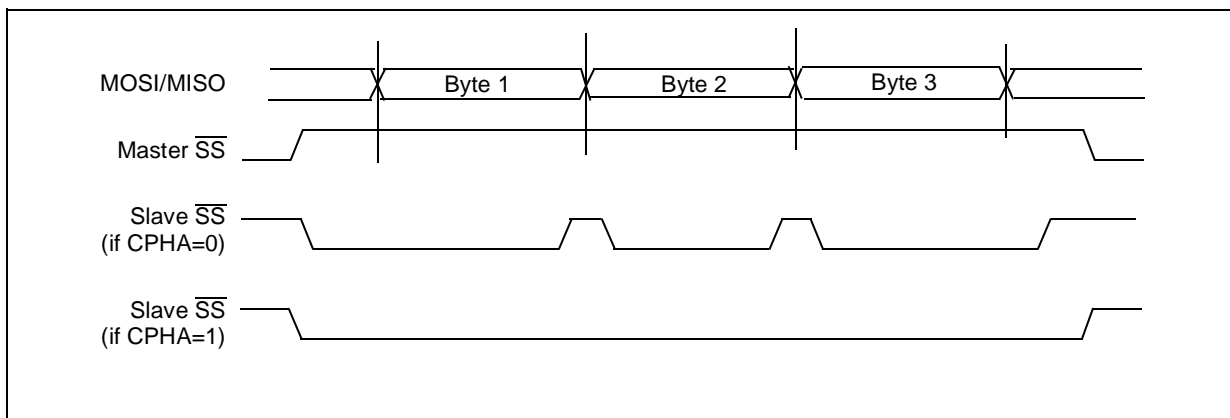
If CPHA=1 (data latched on 2nd clock edge):

- $\overline{SS}$  internal must be held low during the entire transmission. This implies that in single slave applications the  $\overline{SS}$  pin either can be tied to  $V_{SS}$ , or made free for standard I/O by managing the SS function by software (SSM= 1 and SSI=0 in the in the SPICSR register)

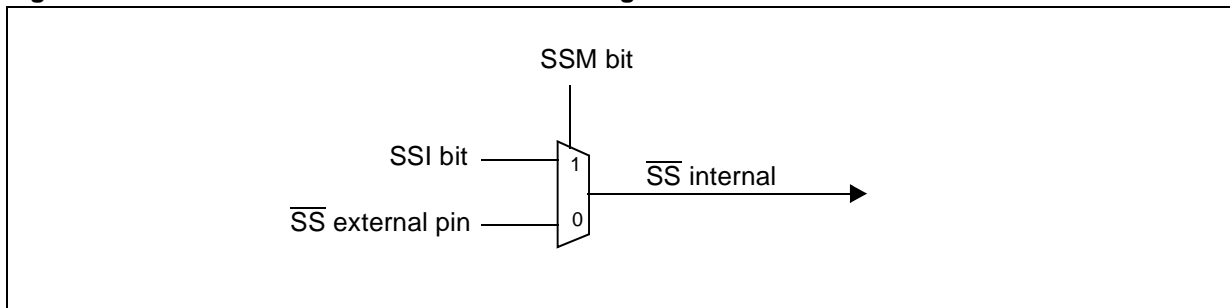
If CPHA=0 (data latched on 1st clock edge):

- $\overline{SS}$  internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If  $\overline{SS}$  is not pulled high, a Write Collision error will occur when the slave writes to the shift register (see Section 11.4.5.3).

**Figure 49. Generic  $\overline{SS}$  Timing Diagram**



**Figure 50. Hardware/Software Slave Select Management**



## SERIAL PERIPHERAL INTERFACE (Cont'd)

### 11.4.3.3 Master Mode Operation

In master mode, the serial clock is output on the SCK pin. The clock frequency, polarity and phase are configured by software (refer to the description of the SPICSR register).

**Note:** The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

To operate the SPI in master mode, perform the following two steps in order (if the SPICSR register is not written first, the SPICR register setting may be not taken into account):

- Write to the SPICSR register:
  - Select the clock frequency by configuring the SPR[2:0] bits.
  - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits. Figure 51 shows the four possible configurations.
 

**Note:** The slave must have the same CPOL and CPHA settings as the master.
  - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the SS pin high for the complete byte transmit sequence.

- Write to the SPICR register:

- Set the MSTR and SPE bits
 

**Note:** MSTR and SPE bits remain set only if SS is high).

The transmit sequence begins when software writes a byte in the SPIDR register.

### 11.4.3.4 Master Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware
- An interrupt request is generated if the SPIE bit is set and the interrupt mask in the CCR register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

- An access to the SPICSR register while the SPIF bit is set
- A read to the SPIDR register.

**Note:** While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

### 11.4.3.5 Slave Mode Operation

In slave mode, the serial clock is received on the SCK pin from the master device.

To operate the SPI in slave mode:

- Write to the SPICSR register to perform the following actions:
  - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see Figure 51).
 

**Note:** The slave must have the same CPOL and CPHA settings as the master.
  - Manage the  $\overline{SS}$  pin as described in Section 11.4.3.2 and Figure 49. If CPHA=1  $\overline{SS}$  must be held low continuously. If CPHA=0  $\overline{SS}$  must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
- Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

### 11.4.3.6 Slave Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

- The SPIF bit is set by hardware
- An interrupt request is generated if SPIE bit is set and interrupt mask in the CCR register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

- An access to the SPICSR register while the SPIF bit is set.
- A write or a read to the SPIDR register.

**Notes:** While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

The SPIF bit can be cleared during a second transmission; however, it must be cleared before the second SPIF bit in order to prevent an Overrun condition (see Section 11.4.5.2).

**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**11.4.4 Clock Phase and Clock Polarity**

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 51).

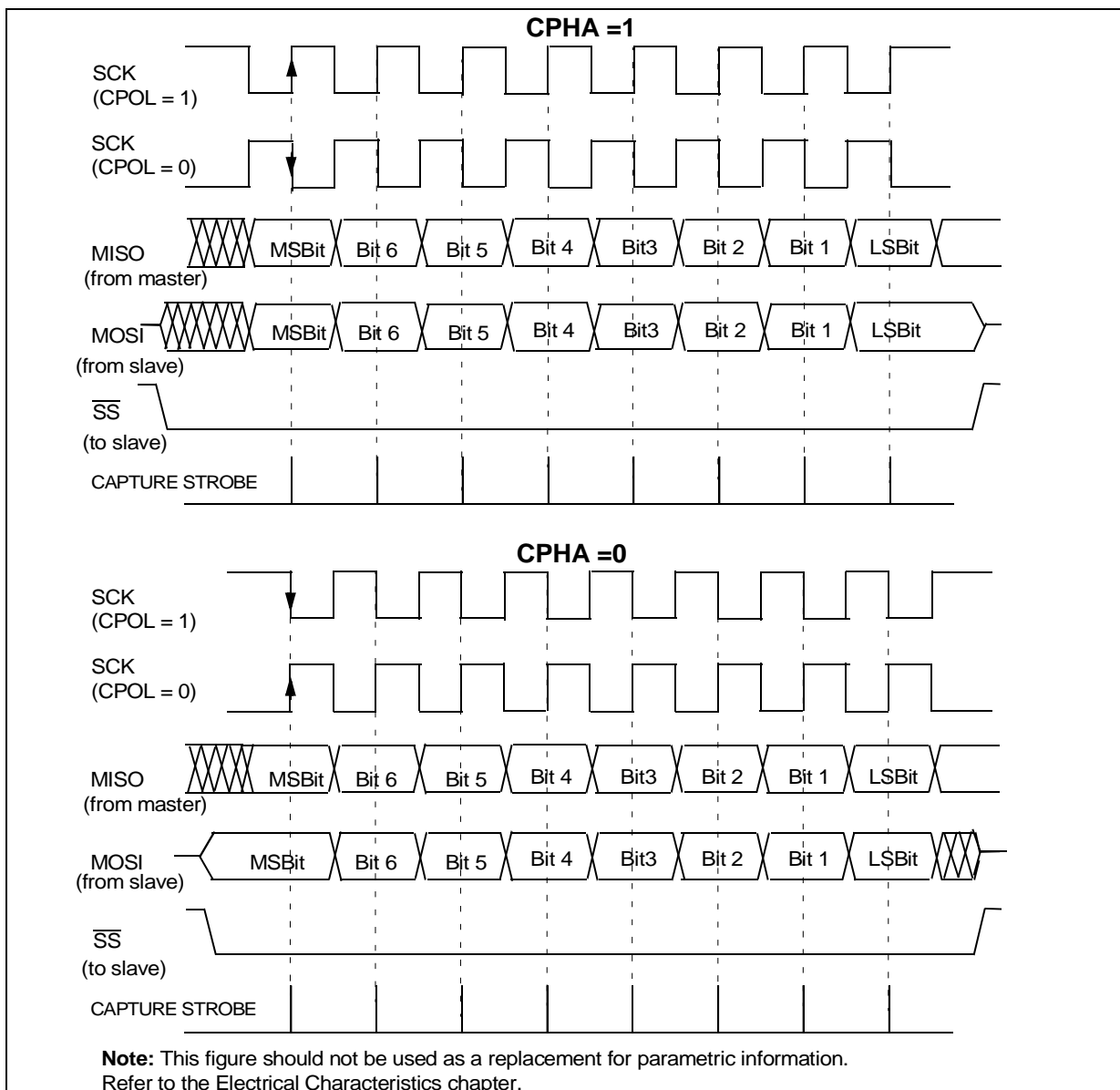
**Note:** The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

The combination of the CPOL clock polarity and CPHA (clock phase) bits selects the data capture clock edge

Figure 51, shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin, the MISO pin, the MOSI pin are directly connected between the master and the slave device.

**Note:** If CPOL is changed at the communication byte boundaries, the SPI must be disabled by re-setting the SPE bit.

**Figure 51. Data Clock Timing Diagram**



## SERIAL PERIPHERAL INTERFACE (Cont'd)

### 11.4.5 Error Flags

#### 11.4.5.1 Master Mode Fault (MODF)

Master mode fault occurs when the master device has its  $\overline{SS}$  pin pulled low.

When a Master mode fault occurs:

- The MODF bit is set and an SPI interrupt request is generated if the SPIE bit is set.
- The SPE bit is reset. This blocks all output from the Device and disables the SPI peripheral.
- The MSTR bit is reset, thus forcing the Device into slave mode.

Clearing the MODF bit is done through a software sequence:

1. A read access to the SPICSR register while the MODF bit is set.
2. A write to the SPICR register.

**Notes:** To avoid any conflicts in an application with multiple slaves, the  $\overline{SS}$  pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits may be restored to their original state during or after this clearing sequence.

Hardware does not allow the user to set the SPE and MSTR bits while the MODF bit is set except in the MODF bit clearing sequence.

In a slave device, the MODF bit can not be set, but in a multi master configuration the Device can be in slave mode with the MODF bit set.

The MODF bit indicates that there might have been a multi-master conflict and allows software to handle this using an interrupt routine and either perform to a reset or return to an application default state.

#### 11.4.5.2 Overrun Condition (OVR)

An overrun condition occurs, when the master device has sent a data byte and the slave device has not cleared the SPIF bit issued from the previously transmitted byte.

When an Overrun occurs:

- The OVR bit is set and an interrupt request is generated if the SPIE bit is set.

In this case, the receiver buffer contains the byte sent after the SPIF bit was last cleared. A read to the SPIDR register returns this byte. All other bytes are lost.

The OVR bit is cleared by reading the SPICSR register.

#### 11.4.5.3 Write Collision Error (WCOL)

A write collision occurs when the software tries to write to the SPIDR register while a data transfer is taking place with an external device. When this happens, the transfer continues uninterrupted; and the software write will be unsuccessful.

Write collisions can occur both in master and slave mode. See also Section 11.4.3.2 "Slave Select Management" on page 76.

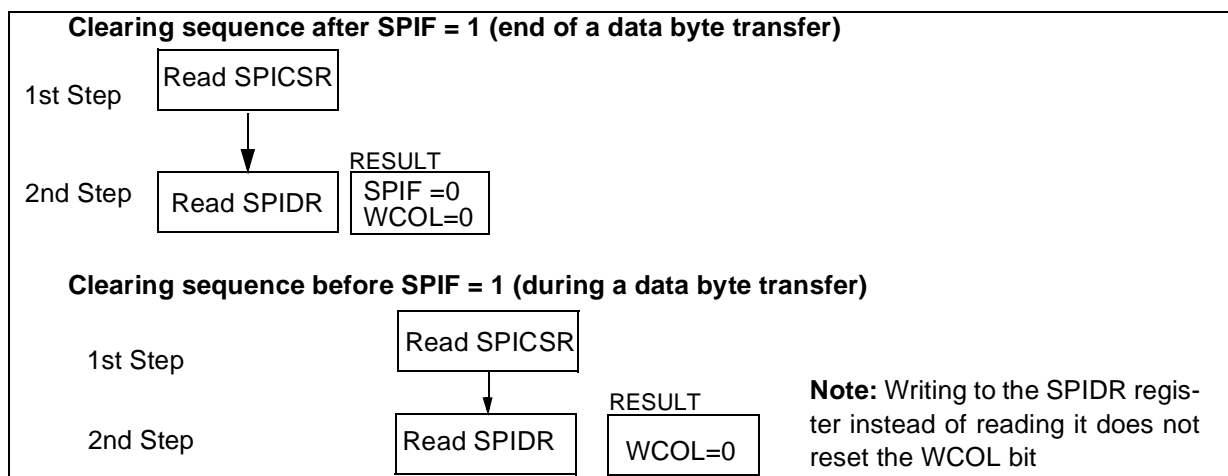
**Note:** a "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

The WCOL bit in the SPICSR register is set if a write collision occurs.

No SPI interrupt is generated when the WCOL bit is set (the WCOL bit is a status flag only).

Clearing the WCOL bit is done through a software sequence (see Figure 52).

**Figure 52. Clearing the WCOL bit (Write Collision Flag) Software Sequence**



**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**11.4.5.4 Single Master and Multimaster Configurations**

There are two types of SPI systems:

- Single Master System
- Multimaster System

**Single Master System**

A typical single master system may be configured, using a device as the master and four devices as slaves (see Figure 53).

The master device selects the individual slave devices by using four pins of a parallel port to control the four  $\overline{SS}$  pins of the slave devices.

The  $\overline{SS}$  pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices.

**Note:** To prevent a bus conflict on the MISO line the master allows only one active slave device during a transmission.

For more security, the slave device may respond to the master with the received data byte. Then the master will receive the previous byte back from the slave device if all MISO and MOSI pins are connected and the slave has not written to its SPIDR register.

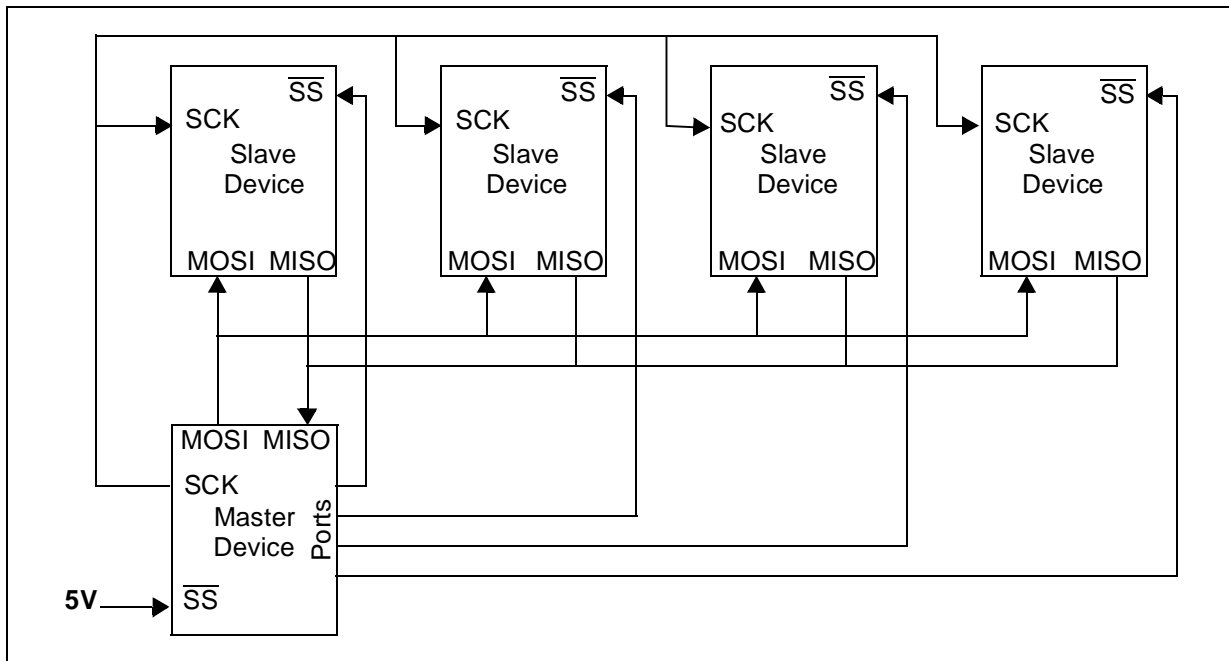
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

**Multi-Master System**

A multi-master system may also be configured by the user. Transfer of master control could be implemented using a handshake method through the I/O ports or by an exchange of code messages through the serial peripheral interface system.

The multi-master system is principally handled by the MSTR bit in the SPICR register and the MODF bit in the SPICSR register.

**Figure 53. Single Master / Multiple Slave Configuration**





## SERIAL PERIPHERAL INTERFACE (Cont'd)

## 11.4.6 Low Power Modes

Mode	Description
WAIT	No effect on SPI. SPI interrupt events cause the Device to exit from WAIT mode.
HALT	SPI registers are frozen. In HALT mode, the SPI is inactive. SPI operation resumes when the Device is woken up by an interrupt with "exit from HALT mode" capability. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetching). If several data are received before the wake-up event, then an overrun error is generated. This error can be detected after the fetch of the interrupt routine that woke up the Device.

## 11.4.6.1 Using the SPI to wake-up the Device from Halt mode

In slave configuration, the SPI is able to wake-up the Device from HALT mode through a SPIF interrupt. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetch). If multiple data transfers have been performed before software clears the SPIF bit, then the OVR bit is set by hardware.

**Note:** When waking up from Halt mode, if the SPI remains in Slave mode, it is recommended to perform an extra communications cycle to bring the SPI from Halt mode state to normal state. If the SPI exits from Slave mode, it returns to normal state immediately.

**Caution:** The SPI can wake-up the Device from Halt mode only if the Slave Select signal (external

$\overline{SS}$  pin or the SSI bit in the SPICSR register) is low when the Device enters Halt mode. So if Slave selection is configured as external (see Section 11.4.3.2), make sure the master drives a low level on the  $\overline{SS}$  pin when the slave enters Halt mode.

## 11.4.7 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
SPI End of Transfer Event	SPIF	SPIE	Yes	Yes
Master Mode Fault Event	MODF		Yes	No
Overrun Error	OVR		Yes	No

**Note:** The SPI interrupt events are connected to the same interrupt vector (see Interrupts chapter). They generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**11.4.8 Register Description**

**CONTROL REGISTER (SPICR)**

Read/Write

Reset Value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPIE** *Serial Peripheral Interrupt Enable*.

This bit is set and cleared by software.

0: Interrupt is inhibited

1: An SPI interrupt is generated whenever an End of Transfer event, Master Mode Fault or Over-run error occurs (SPIF=1, MODF=1 or OVR=1 in the SPICSR register)

Bit 6 = **SPE** *Serial Peripheral Output Enable*.

This bit is set and cleared by software. It is also cleared by hardware when, in master mode,  $\overline{SS}=0$  (see Section 11.4.5.1 "Master Mode Fault (MODF)" on page 79). The SPE bit is cleared by reset, so the SPI peripheral is not initially connected to the external pins.

0: I/O pins free for general purpose I/O

1: SPI I/O pin alternate functions enabled

Bit 5 = **SPR2** *Divider Enable*.

This bit is set and cleared by software and is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to Table 16 SPI Master mode SCK Frequency.

0: Divider by 2 enabled

1: Divider by 2 disabled

**Note:** This bit has no effect in slave mode.

Bit 4 = **MSTR** *Master Mode*.

This bit is set and cleared by software. It is also cleared by hardware when, in master mode,  $\overline{SS}=0$  (see Section 11.4.5.1 "Master Mode Fault (MODF)" on page 79).

0: Slave mode

1: Master mode. The function of the SCK pin changes from an input to an output and the functions of the MISO and MOSI pins are reversed.

Bit 3 = **CPOL** *Clock Polarity*.

This bit is set and cleared by software. This bit determines the idle state of the serial Clock. The CPOL bit affects both the master and slave modes.

0: SCK pin has a low level idle state

1: SCK pin has a high level idle state

**Note:** If CPOL is changed at the communication byte boundaries, the SPI must be disabled by re-setting the SPE bit.

Bit 2 = **CPHA** *Clock Phase*.

This bit is set and cleared by software.

0: The first clock transition is the first data capture edge.

1: The second clock transition is the first capture edge.

**Note:** The slave must have the same CPOL and CPHA settings as the master.

Bits 1:0 = **SPR[1:0]** *Serial Clock Frequency*.

These bits are set and cleared by software. Used with the SPR2 bit, they select the baud rate of the SPI serial clock SCK output by the SPI in master mode.

**Note:** These 2 bits have no effect in slave mode.

**Table 16. SPI Master mode SCK Frequency**

Serial Clock	SPR2	SPR1	SPR0
$f_{CPU}/4$	1	0	0
$f_{CPU}/8$	0	0	0
$f_{CPU}/16$	0	0	1
$f_{CPU}/32$	1	1	0
$f_{CPU}/64$	0	1	0
$f_{CPU}/128$	0	1	1

**SERIAL PERIPHERAL INTERFACE (Cont'd)****CONTROL/STATUS REGISTER (SPICSR)**

Read/Write (some bits Read Only)

Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	OVR	MODF	-	SOD	SSM	SSI

Bit 7 = **SPIF** *Serial Peripheral Data Transfer Flag (Read only).*

This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE=1 in the SPICR register. It is cleared by a software sequence (an access to the SPICSR register followed by a write or a read to the SPIDR register).

0: Data transfer is in progress or the flag has been cleared.

1: Data transfer between the Device and an external device has been completed.

**Note:** While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Bit 6 = **WCOL** *Write Collision status (Read only).*

This bit is set by hardware when a write to the SPIDR register is done during a transmit sequence. It is cleared by a software sequence (see Figure 52).

0: No write collision occurred

1: A write collision has been detected

Bit 5 = **OVR** *SPI Overrun error (Read only).*

This bit is set by hardware when the byte currently being received in the shift register is ready to be transferred into the SPIDR register while SPIF = 1 (See Section 11.4.5.2). An interrupt is generated if SPIE = 1 in SPICSR register. The OVR bit is cleared by software reading the SPICSR register.

0: No overrun error

1: Overrun error detected

Bit 4 = **MODF** *Mode Fault flag (Read only).*

This bit is set by hardware when the  $\overline{SS}$  pin is pulled low in master mode (see Section 11.4.5.1 "Master Mode Fault (MODF)" on page 79). An SPI interrupt can be generated if SPIE=1 in the SPICSR register. This bit is cleared by a software sequence (An access to the SPICSR register while MODF=1 followed by a write to the SPICR register).

0: No master mode fault detected

1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

Bit 2 = **SOD** *SPI Output Disable.*

This bit is set and cleared by software. When set, it disables the alternate function of the SPI output (MOSI in master mode / MISO in slave mode)

0: SPI output enabled (if SPE=1)

1: SPI output disabled

Bit 1 = **SSM**  $\overline{SS}$  *Management.*

This bit is set and cleared by software. When set, it disables the alternate function of the SPI  $\overline{SS}$  pin and uses the SSI bit value instead. See Section 11.4.3.2 "Slave Select Management" on page 76.

0: Hardware management ( $\overline{SS}$  managed by external pin)

1: Software management (internal  $\overline{SS}$  signal controlled by SSI bit. External  $\overline{SS}$  pin free for general-purpose I/O)

Bit 0 = **SSI**  $\overline{SS}$  *Internal Mode.*

This bit is set and cleared by software. It acts as a 'chip select' by controlling the level of the  $\overline{SS}$  slave select signal when the SSM bit is set.

0 : Slave selected

1 : Slave deselected

**DATA I/O REGISTER (SPIDR)**

Read/Write

Reset Value: Undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

The SPIDR register is used to transmit and receive data on the serial bus. In a master device, a write to this register will initiate transmission/reception of another byte.

**Notes:** During the last clock cycle the SPIF bit is set, a copy of the received data byte in the shift register is moved to a buffer. When the user reads the serial peripheral data I/O register, the buffer is actually being read.

While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

**Warning:** A write to the SPIDR register places data directly into the shift register for transmission.

A read to the SPIDR register returns the value located in the buffer and not the content of the shift register (see Figure 47).

## SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 17. SPI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0021h	<b>SPIDR</b> Reset Value	MSB x	x	x	x	x	x	x	LSB x
0022h	<b>SPICR</b> Reset Value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
0023h	<b>SPICSR</b> Reset Value	SPIF 0	WCOL 0	OR 0	MODF 0	0	SOD 0	SSM 0	SSI 0

## 11.5 SERIAL COMMUNICATIONS INTERFACE (SCI)

### 11.5.1 Introduction

The Serial Communications Interface (SCI) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The SCI offers a very wide range of baud rates using two baud rate generator systems.

### 11.5.2 Main Features

- Full duplex, asynchronous communications
- NRZ standard format (Mark/Space)
- Dual baud rate generator systems
- Independently programmable transmit and receive baud rates up to 500K baud.
- Programmable data word length (8 or 9 bits)
- Receive buffer full, Transmit buffer empty and End of Transmission flags
- Two receiver wake-up modes:
  - Address bit (MSB)
  - Idle line
- Muting function for multiprocessor configurations
- Separate enable bits for Transmitter and Receiver
- Four error detection flags:
  - Overrun error
  - Noise error
  - Frame error
  - Parity error
- Five interrupt sources with flags:
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error detected
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Reduced power consumption mode

### 11.5.3 General Description

The interface is externally connected to another device by two pins (see Figure 55):

- TDO: Transmit Data Output. When the transmitter and the receiver are disabled, the output pin returns to its I/O port configuration. When the transmitter and/or the receiver are enabled and nothing is to be transmitted, the TDO pin is at high level.
- RDI: Receive Data Input is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

Through these pins, serial data is transmitted and received as frames comprising:

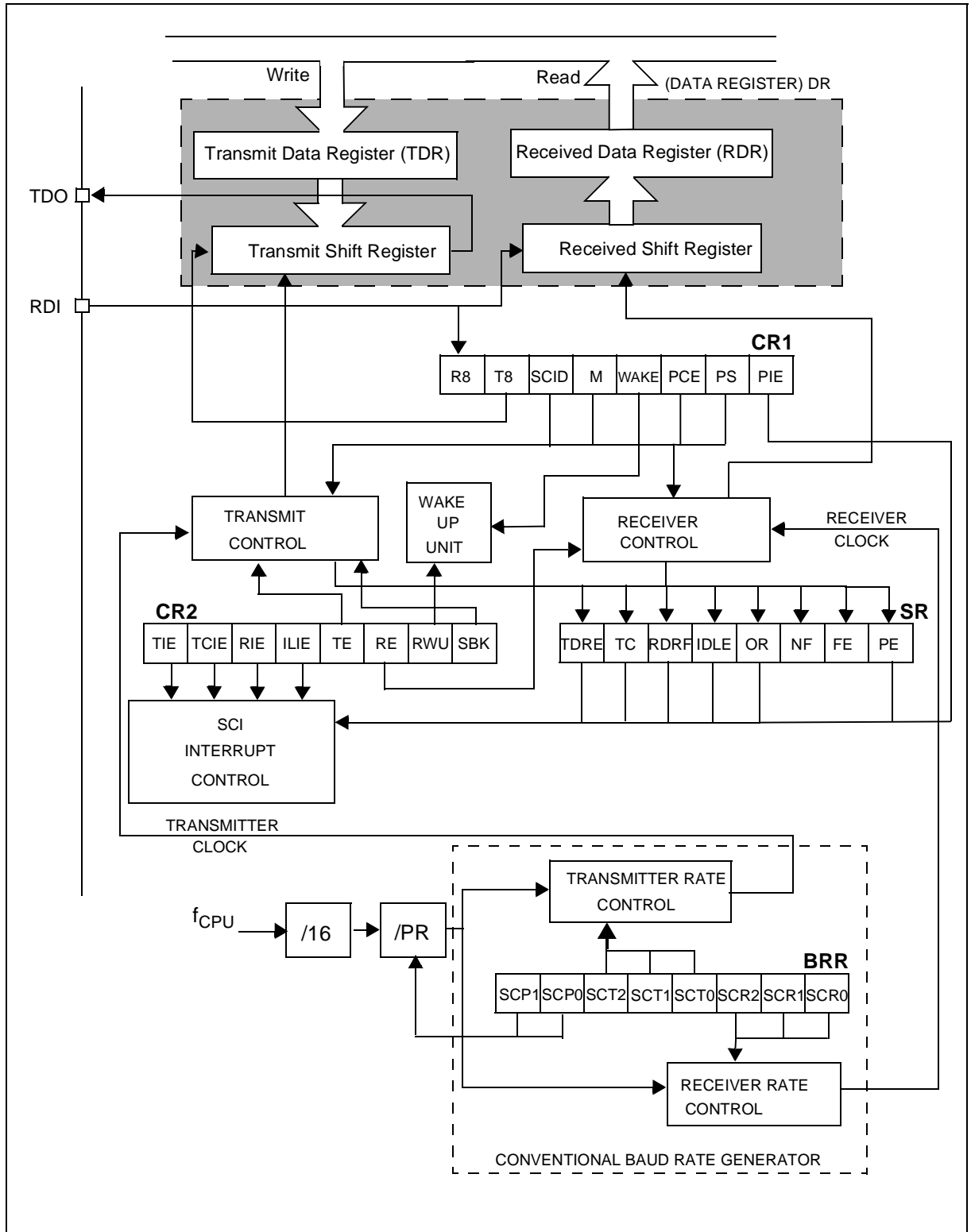
- An Idle Line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) least significant bit first
- A Stop bit indicating that the frame is complete.

This interface uses two types of baud rate generator:

- A conventional type for commonly-used baud rates,
- An extended type with a prescaler offering a very wide range of baud rates even with non-standard oscillator frequencies.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 54. SCI Block Diagram



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**11.5.4 Functional Description**

The block diagram of the Serial Control Interface, is shown in Figure 54. It contains 6 dedicated registers:

- Two control registers (SCICR1 & SCICR2)
- A status register (SCISR)
- A baud rate register (SCIBRR)
- An extended prescaler receiver register (SCIERR)
- An extended prescaler transmitter register (SCIETPR)

Refer to the register descriptions in Section 11.5.7 for the definitions of each bit.

**11.5.4.1 Serial Data Format**

Word length may be selected as being either 8 or 9 bits by programming the M bit in the SCICR1 register (see Figure 54).

The TDO pin is in low state during the start bit.

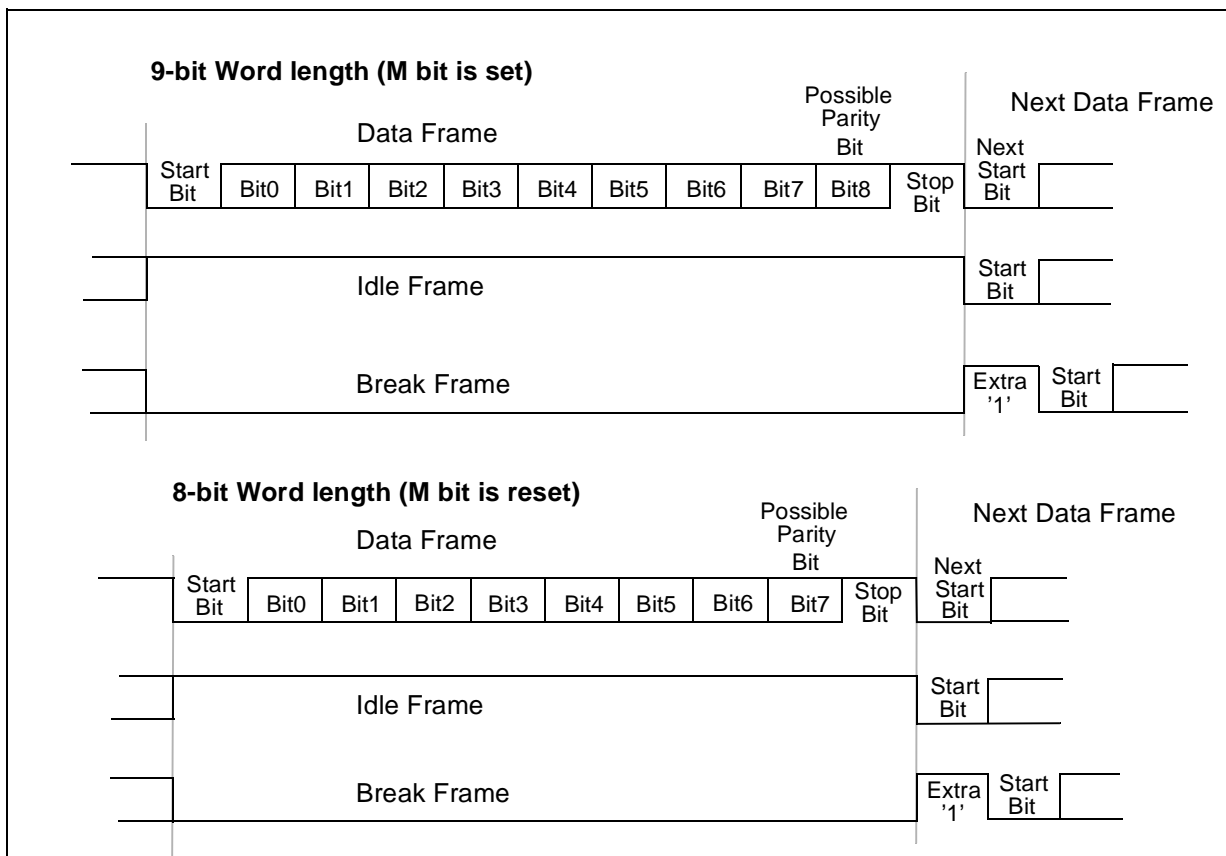
The TDO pin is in high state during the stop bit.

An Idle character is interpreted as an entire frame of "1"s followed by the start bit of the next frame which contains data.

A Break character is interpreted on receiving "0"s for some multiple of the frame period. At the end of the last break frame the transmitter inserts an extra "1" bit to acknowledge the start bit.

Transmission and reception are driven by their own baud rate generator.

**Figure 55. Word Length Programming**



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****11.5.4.2 Transmitter**

The transmitter can send data words of either 8 or 9 bits depending on the M bit status. When the M bit is set, word length is 9 bits and the 9th bit (the MSB) has to be stored in the T8 bit in the SCICR1 register.

**Character Transmission**

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the SCIDR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 54).

**Procedure**

- Select the M bit to define the word length.
- Select the desired baud rate using the SCIBRR and the SCIETPR registers.
- Set the TE bit to assign the TDO pin to the alternate function and to send a idle frame as first transmission.
- Access the SCISR register and write the data to send in the SCIDR register (this sequence clears the TDRE bit). Repeat this sequence for each data to be transmitted.

Clearing the TDRE bit is always performed by the following software sequence:

1. An access to the SCISR register
2. A write to the SCIDR register

The TDRE bit is set by hardware and it indicates:

- The TDR register is empty.
- The data transfer is beginning.
- The next data can be written in the SCIDR register without overwriting the previous data.

This flag generates an interrupt if the TIE bit is set and the I bit is cleared in the CCR register.

When a transmission is taking place, a write instruction to the SCIDR register stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the SCIDR register places the data directly in the shift register, the data transmission starts, and the TDRE bit is immediately set.

When a frame transmission is complete (after the stop bit or after the break frame) the TC bit is set and an interrupt is generated if the TCIE is set and the I bit is cleared in the CCR register.

Clearing the TC bit is performed by the following software sequence:

1. An access to the SCISR register
2. A write to the SCIDR register

**Note:** The TDRE and TC bits are cleared by the same software sequence.

**Break Characters**

Setting the SBK bit loads the shift register with a break character. The break frame length depends on the M bit (see Figure 55).

As long as the SBK bit is set, the SCI send break frames to the TDO pin. After clearing this bit by software the SCI insert a logic 1 bit at the end of the last break frame to guarantee the recognition of the start bit of the next frame.

**Idle Characters**

Setting the TE bit drives the SCI to send an idle frame before the first data frame.

Clearing and then setting the TE bit during a transmission sends an idle frame after the current word.

**Note:** Resetting and setting the TE bit causes the data in the TDR register to be lost. Therefore the best time to toggle the TE bit is when the TDRE bit is set i.e. before writing the next byte in the SCIDR.



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****11.5.4.3 Receiver**

The SCI can receive data words of either 8 or 9 bits. When the M bit is set, word length is 9 bits and the MSB is stored in the R8 bit in the SCICR1 register.

**Character reception**

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, the SCIDR register consists of a buffer (RDR) between the internal bus and the received shift register (see Figure 54).

**Procedure**

- Select the M bit to define the word length.
- Select the desired baud rate using the SCIBRR and the SCIERPR registers.
- Set the RE bit, this enables the receiver which begins searching for a start bit.

When a character is received:

- The RDRF bit is set. It indicates that the content of the shift register is transferred to the RDR.
- An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.
- The error flags can be set if a frame error, noise or an overrun error has been detected during reception.

Clearing the RDRF bit is performed by the following software sequence done by:

1. An access to the SCISR register
2. A read to the SCIDR register.

The RDRF bit must be cleared before the end of the reception of the next character to avoid an overrun error.

**Break Character**

When a break character is received, the SPI handles it as a framing error.

**Idle Character**

When an idle frame is detected, there is the same procedure as a data received character plus an interrupt if the ILIE bit is set and the I bit is cleared in the CCR register.

**Overrun Error**

An overrun error occurs when a character is received when RDRF has not been reset. Data can not be transferred from the shift register to the RDR register as long as the RDRF bit is not cleared.

When an overrun error occurs:

- The OR bit is set.
- The RDR content will not be lost.
- The shift register will be overwritten.
- An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.

The OR bit is reset by an access to the SCISR register followed by a SCIDR register read operation.

**Noise Error**

Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

When noise is detected in a frame:

- The NF is set at the rising edge of the RDRF bit.
- Data is transferred from the Shift register to the SCIDR register.
- No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The NF bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

**Framing Error**

A framing error is detected when:

- The stop bit is not recognized on reception at the expected time, following either a de-synchronization or excessive noise.
- A break is received.

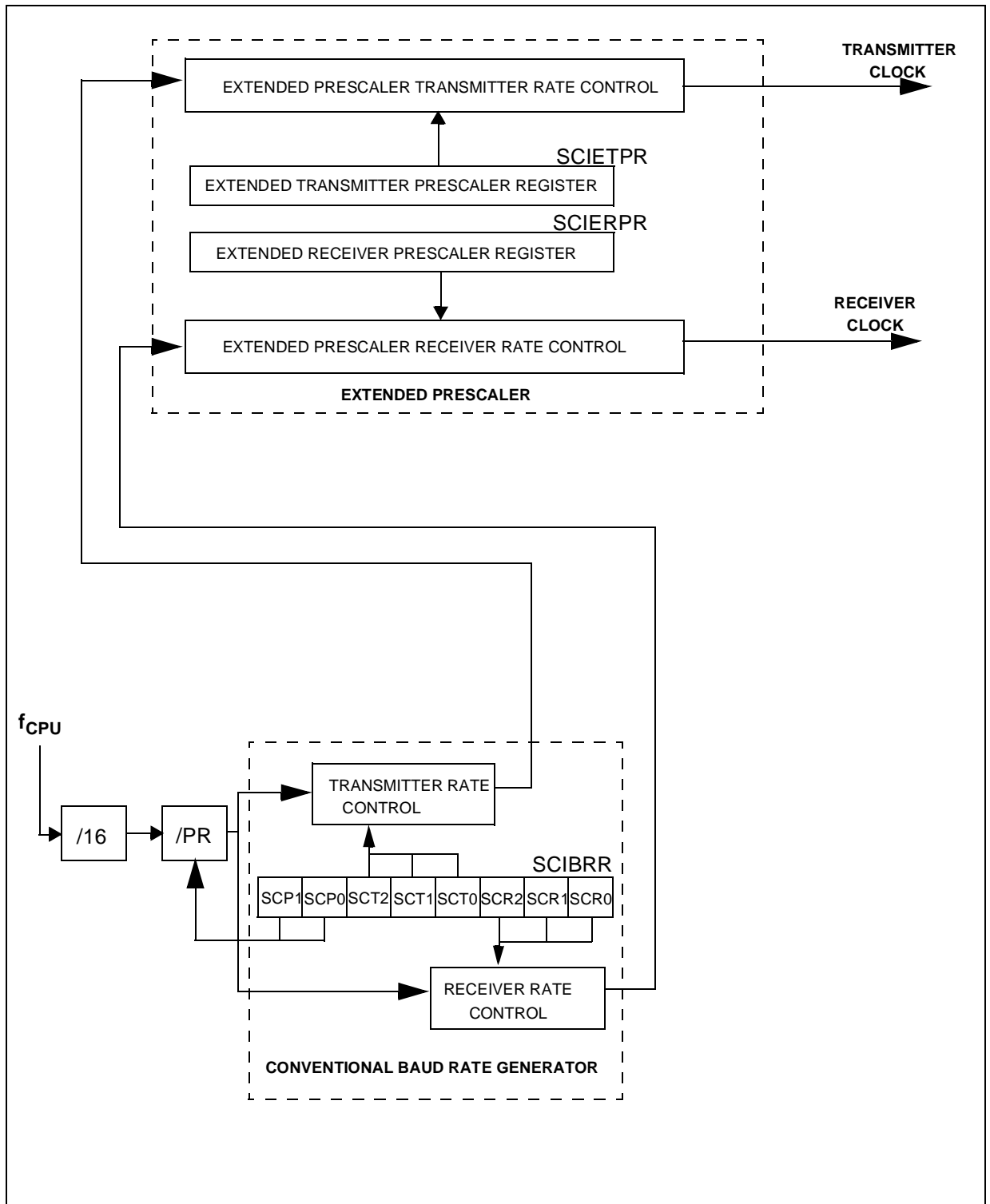
When the framing error is detected:

- the FE bit is set by hardware
- Data is transferred from the Shift register to the SCIDR register.
- No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The FE bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 56. SCI Baud Rate and Extended Prescaler Block Diagram



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****11.5.4.4 Conventional Baud Rate Generation**

The baud rate for the receiver and transmitter (Rx and Tx) are set independently and calculated as follows:

$$T_x = \frac{f_{\text{CPU}}}{(16 \cdot \text{PR}) \cdot \text{TR}} \quad R_x = \frac{f_{\text{CPU}}}{(16 \cdot \text{PR}) \cdot \text{RR}}$$

with:

PR = 1, 3, 4 or 13 (see SCP[1:0] bits)

TR = 1, 2, 4, 8, 16, 32, 64, 128

(see SCT[2:0] bits)

RR = 1, 2, 4, 8, 16, 32, 64, 128

(see SCR[2:0] bits)

All these bits are in the SCIBRR register.

**Example:** If  $f_{\text{CPU}}$  is 8 MHz (normal mode) and if PR=13 and TR=RR=1, the transmit and receive baud rates are 38400 baud.

**Note:** the baud rate registers MUST NOT be changed while the transmitter or the receiver is enabled.

**11.5.4.5 Extended Baud Rate Generation**

The extended prescaler option gives a very fine tuning on the baud rate, using a 255 value prescaler, whereas the conventional Baud Rate Generator retains industry standard software compatibility.

The extended baud rate generator block diagram is described in the Figure 56.

The output clock rate sent to the transmitter or to the receiver will be the output from the 16 divider divided by a factor ranging from 1 to 255 set in the SCIERPR or the SCIETPR register.

**Note:** the extended prescaler is activated by setting the SCIETPR or SCIERPR register to a value other than zero. The baud rates are calculated as follows:

$$T_x = \frac{f_{\text{CPU}}}{16 \cdot \text{ETPR} \cdot (\text{PR} \cdot \text{TR})} \quad R_x = \frac{f_{\text{CPU}}}{16 \cdot \text{ERPR} \cdot (\text{PR} \cdot \text{RR})}$$

with:

ETPR = 1,...,255 (see SCIETPR register)

ERPR = 1,.. 255 (see SCIERPR register)

**11.5.4.6 Receiver Muting and Wake-up Feature**

In multiprocessor configurations it is often desirable that only the intended message recipient should actively receive the full message contents, thus reducing redundant SCI service overhead for all non addressed receivers.

The non addressed devices may be placed in sleep mode by means of the muting function.

Setting the RWU bit by software puts the SCI in sleep mode:

All the reception status bits can not be set.

All the receive interrupts are inhibited.

A muted receiver may be awakened by one of the following two ways:

- by Idle Line detection if the WAKE bit is reset,
- by Address Mark detection if the WAKE bit is set.

Receiver wakes-up by Idle Line detection when the Receive line has recognised an Idle Frame. Then the RWU bit is reset by hardware but the IDLE bit is not set.

Receiver wakes-up by Address Mark detection when it received a "1" as the most significant bit of a word, thus indicating that the message is an address. The reception of this particular word wakes up the receiver, resets the RWU bit and sets the RDRF bit, which allows the receiver to receive this word normally and to use it as an address word.

**Caution:** In Mute mode, do not write to the SCICR2 register. If the SCI is in Mute mode during the read operation (RWU=1) and a address mark wake up event occurs (RWU is reset) before the write operation, the RWU bit will be set again by this write operation. Consequently the address byte is lost and the SCI is not woken up from Mute mode.

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**11.5.4.7 Parity Control**

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the SCICR1 register. Depending on the frame length defined by the M bit, the possible SCI frame formats are as listed in Table 18.

**Table 18. Frame Formats**

M bit	PCE bit	SCI frame
0	0	SB   8 bit data   STB
0	1	SB   7-bit data   PB   STB
1	0	SB   9-bit data   STB
1	1	SB   8-bit data PB   STB

**Legend:** SB = Start Bit, STB = Stop Bit, PB = Parity Bit

**Note:** In case of wake up by an address mark, the MSB bit of the data is taken into account and not the parity bit

**Even parity:** the parity bit is calculated to obtain an even number of “1s” inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Ex: data=00110101; 4 bits set => parity bit will be 0 if even parity is selected (PS bit = 0).

**Odd parity:** the parity bit is calculated to obtain an odd number of “1s” inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Ex: data=00110101; 4 bits set => parity bit will be 1 if odd parity is selected (PS bit = 1).

**Transmission mode:** If the PCE bit is set then the MSB bit of the data written in the data register is not transmitted but is changed by the parity bit.

**Reception mode:** If the PCE bit is set then the interface checks if the received data byte has an even number of “1s” if even parity is selected

(PS=0) or an odd number of “1s” if odd parity is selected (PS=1). If the parity check fails, the PE flag is set in the SCISR register and an interrupt is generated if PIE is set in the SCICR1 register.

**11.5.5 Low Power Modes**

Mode	Description
WAIT	No effect on SCI. SCI interrupts cause the device to exit from Wait mode.
HALT	SCI registers are frozen. In Halt mode, the SCI stops transmitting/receiving until Halt mode is exited.

**11.5.6 Interrupts**

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Transmit Data Register Empty	TDRE	TIE	Yes	No
Transmission Complete	TC	TCIE	Yes	No
Received Data Ready to be Read	RDRF	RIE	Yes	No
Overrun Error Detected	OR		Yes	No
Idle Line Detected	IDLE	ILIE	Yes	No
Parity Error	PE	PIE	Yes	No

The SCI interrupt events are connected to the same interrupt vector.

These events generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****11.5.7 Register Description****STATUS REGISTER (SCISR)**

Read Only

Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR	NF	FE	PE

**Bit 7 = TDRE** *Transmit data register empty.*

This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TIE bit=1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).

0: Data is not transferred to the shift register

1: Data is transferred to the shift register

**Note:** Data will not be transferred to the shift register unless the TDRE bit is cleared.**Bit 6 = TC** *Transmission complete.*

This bit is set by hardware when transmission of a frame containing Data, a Preamble or a Break is complete. An interrupt is generated if TCIE=1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).

0: Transmission is not complete

1: Transmission is complete

**Note:** TC is not set after the transmission of a Preamble or a Break.**Bit 5 = RDRF** *Received data ready flag.*

This bit is set by hardware when the content of the RDR register has been transferred to the SCIDR register. An interrupt is generated if RIE=1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: Data is not received

1: Received data is ready to be read

**Bit 4 = IDLE** *Idle line detect.*

This bit is set by hardware when an Idle Line is detected. An interrupt is generated if the ILIE=1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Idle Line is detected

1: Idle Line is detected

**Note:** The IDLE bit will not be set again until the RDRF bit has been set itself (i.e. a new idle line occurs).**Bit 3 = OR** *Overrun error.*

This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RDRF=1. An interrupt is generated if RIE=1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Overrun error

1: Overrun error is detected

**Note:** When this bit is set RDR register content will not be lost but the shift register will be overwritten.**Bit 2 = NF** *Noise flag.*

This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No noise is detected

1: Noise is detected

**Note:** This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt.**Bit 1 = FE** *Framing error.*

This bit is set by hardware when a de-synchronization, excessive noise or a break character is detected. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Framing error is detected

1: Framing error or break character is detected

**Note:** This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt. If the word currently being transferred causes both frame error and overrun error, it will be transferred and only the OR bit will be set.**Bit 0 = PE** *Parity error.*

This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE=1 in the SCICR1 register.

0: No parity error

1: Parity error

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**CONTROL REGISTER 1 (SCICR1)**

Read/Write

Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE	PS	PIE

**Bit 7 = R8 Receive data bit 8.**  
 This bit is used to store the 9th bit of the received word when M=1.

**Bit 6 = T8 Transmit data bit 8.**  
 This bit is used to store the 9th bit of the transmitted word when M=1.

**Bit 5 = SCID Disabled for low power consumption**  
 When this bit is set the SCI prescalers and outputs are stopped and the end of the current byte transfer in order to reduce power consumption. This bit is set and cleared by software.  
 0: SCI enabled  
 1: SCI prescaler and outputs disabled

**Bit 4 = M Word length.**  
 This bit determines the word length. It is set or cleared by software.  
 0: 1 Start bit, 8 Data bits, 1 Stop bit  
 1: 1 Start bit, 9 Data bits, 1 Stop bit

**Note:** The M bit must not be modified during a data transfer (both transmission and reception).

**Bit 3 = WAKE Wake-Up method.**  
 This bit determines the SCI Wake-Up method, it is set or cleared by software.  
 0: Idle Line  
 1: Address Mark

**Bit 2 = PCE Parity control enable.**  
 This bit selects the hardware parity control (generation and detection). When the parity control is enabled, the computed parity is inserted at the MSB position (9th bit if M=1; 8th bit if M=0) and parity is checked on the received data. This bit is set and cleared by software. Once it is set, PCE is active after the current byte (in reception and in transmission).  
 0: Parity control disabled  
 1: Parity control enabled

**Bit 1 = PS Parity selection.**  
 This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity will be selected after the current byte.  
 0: Even parity  
 1: Odd parity

**Bit 0 = PIE Parity interrupt enable.**  
 This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). It is set and cleared by software.  
 0: Parity error interrupt disabled  
 1: Parity error interrupt enabled.

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****CONTROL REGISTER 2 (SCICR2)**

Read/Write

Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

**Bit 7 = TIE** *Transmitter interrupt enable.*

This bit is set and cleared by software.

0: Interrupt is inhibited

1: An SCI interrupt is generated whenever TDRE=1 in the SCISR register

**Bit 6 = TCIE** *Transmission complete interrupt enable*

This bit is set and cleared by software.

0: Interrupt is inhibited

1: An SCI interrupt is generated whenever TC=1 in the SCISR register

**Bit 5 = RIE** *Receiver interrupt enable.*

This bit is set and cleared by software.

0: Interrupt is inhibited

1: An SCI interrupt is generated whenever OR=1 or RDRF=1 in the SCISR register

**Bit 4 = ILIE** *Idle line interrupt enable.*

This bit is set and cleared by software.

0: Interrupt is inhibited

1: An SCI interrupt is generated whenever IDLE=1 in the SCISR register.

**Bit 3 = TE** *Transmitter enable.*

This bit enables the transmitter. It is set and cleared by software.

0: Transmitter is disabled

1: Transmitter is enabled

**Notes:**

- During transmission, a “0” pulse on the TE bit (“0” followed by “1”) sends a preamble (idle line) after the current word.
- When TE is set there is a 1 bit-time delay before the transmission starts.

**Caution:** The TDO pin is free for general purpose I/O only when the TE and RE bits are both cleared (or if TE is never set).

**Bit 2 = RE** *Receiver enable.*

This bit enables the receiver. It is set and cleared by software.

0: Receiver is disabled

1: Receiver is enabled and begins searching for a start bit

**Bit 1 = RWU** *Receiver wake-up.*

This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wake-up sequence is recognized.

0: Receiver in Active mode

1: Receiver in Mute mode

**Note:** Before selecting Mute mode (setting the RWU bit), the SCI must receive some data first, otherwise it cannot function in Mute mode with wakeup by idle line detection.

**Bit 0 = SBK** *Send break.*

This bit set is used to send break characters. It is set and cleared by software.

0: No break character is transmitted

1: Break characters are transmitted

**Note:** If the SBK bit is set to “1” and then to “0”, the transmitter will send a BREAK word at the end of the current word.

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**DATA REGISTER (SCIDR)**

Read/Write

Reset Value: Undefined

Contains the Received or Transmitted data character, depending on whether it is read from or written to.

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).

The TDR register provides the parallel interface between the internal bus and the output shift register (see Figure 54).

The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 54).

**BAUD RATE REGISTER (SCIBRR)**

Read/Write

Reset Value: 0000 0000 (00h)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0

Bits 7:6= **SCP[1:0]** *First SCI Prescaler*

These 2 prescaling bits allow several standard clock division ranges:

PR Prescaling factor	SCP1	SCP0
1	0	0
3	0	1
4	1	0
13	1	1

Bits 5:3 = **SCT[2:0]** *SCI Transmitter rate divisor*  
 These 3 bits, in conjunction with the SCP1 & SCP0 bits define the total division applied to the bus clock to yield the transmit rate clock in conventional Baud Rate Generator mode.

TR dividing factor	SCT2	SCT1	SCT0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Bits 2:0 = **SCR[2:0]** *SCI Receiver rate divisor.*  
 These 3 bits, in conjunction with the SCP[1:0] bits define the total division applied to the bus clock to yield the receive rate clock in conventional Baud Rate Generator mode.

RR Dividing factor	SCR2	SCR1	SCR0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**EXTENDED RECEIVE PRESCALER DIVISION REGISTER (SCIERP)**

Read/Write

Reset Value: 0000 0000 (00h)

Allows setting of the Extended Prescaler rate division factor for the receive circuit.

7							0
ERPR 7	ERPR 6	ERPR 5	ERPR 4	ERPR 3	ERPR 2	ERPR 1	ERPR 0

Bits 7:0 = **ERPR[7:0]** 8-bit Extended Receive Prescaler Register.

The extended Baud Rate Generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see Figure 56) is divided by the binary factor set in the SCIERP register (in the range 1 to 255).

The extended baud rate generator is not used after a reset.

**EXTENDED TRANSMIT PRESCALER DIVISION REGISTER (SCIETPR)**

Read/Write

Reset Value:0000 0000 (00h)

Allows setting of the External Prescaler rate division factor for the transmit circuit.

7							0
ETPR 7	ETPR 6	ETPR 5	ETPR 4	ETPR 3	ETPR 2	ETPR 1	ETPR 0

Bits 7:0 = **ETPR[7:0]** 8-bit Extended Transmit Prescaler Register.

The extended Baud Rate Generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see Figure 56) is divided by the binary factor set in the SCIETPR register (in the range 1 to 255).

The extended baud rate generator is not used after a reset.

**Table 19. Baudrate Selection**

Symbol	Parameter	Conditions			Standard	Baud Rate	Unit
		f <sub>CPU</sub>	Accuracy vs. Standard	Prescaler			
f <sub>Tx</sub> f <sub>Rx</sub>	Communication frequency	8MHz	~0.16%	Conventional Mode TR (or RR)=128, PR=13 TR (or RR)= 32, PR=13 TR (or RR)= 16, PR=13 TR (or RR)= 8, PR=13 TR (or RR)= 4, PR=13 TR (or RR)= 16, PR= 3 TR (or RR)= 2, PR=13 TR (or RR)= 1, PR=13	300 1200 2400 4800 9600 10400 19200 38400	~300.48 ~1201.92 ~2403.84 ~4807.69 ~9615.38 ~10416.67 ~19230.77 ~38461.54	Hz
				~0.79%	Extended Mode ETPR (or ERPR) = 35, TR (or RR)= 1, PR=1	14400	

## SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Table 20. SCI Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
50	<b>SCISR</b> Reset Value	TDRE 1	TC 1	RDRF 0	IDLE 0	OR 0	NF 0	FE 0	PE 0
51	<b>SCIDR</b> Reset Value	DR7 x	DR6 x	DR5 x	DR4 x	DR3 x	DR2 x	DR1 x	DR0 x
52	<b>SCIBRR</b> Reset Value	SCP1 0	SCP0 0	SCT2 0	SCT1 0	SCT0 0	SCR2 0	SCR1 0	SCR0 0
53	<b>SCICR1</b> Reset Value	R8 x	T8 0	SCID 0	M 0	WAKE 0	PCE 0	PS 0	PIE 0
54	<b>SCICR2</b> Reset Value	TIE 0	TCIE 0	RIE 0	ILIE 0	TE 0	RE 0	RWU 0	SBK 0
55	<b>SCIERPR</b> Reset Value	ERPR7 0	ERPR6 0	ERPR5 0	ERPR4 0	ERPR3 0	ERPR2 0	ERPR1 0	ERPR0 0
56	<b>SCIETPR</b> Reset Value	ETPR7 0	ETPR6 0	ETPR5 0	ETPR4 0	ETPR3 0	ETPR2 0	ETPR1 0	ETPR0 0

## 11.6 I<sup>2</sup>C BUS INTERFACE (I2C)

### 11.6.1 Introduction

The I<sup>2</sup>C Bus Interface serves as an interface between the microcontroller and the serial I<sup>2</sup>C bus. It provides both multimaster and slave functions, and controls all I<sup>2</sup>C bus-specific sequencing, protocol, arbitration and timing. It supports fast I<sup>2</sup>C mode (400kHz).

### 11.6.2 Main Features

- Parallel-bus/I<sup>2</sup>C protocol converter
- Multi-master capability
- 7-bit/10-bit Addressing
- Transmitter/Receiver flag
- End-of-byte transmission flag
- Transfer problem detection

#### I<sup>2</sup>C Master Features:

- Clock generation
- I<sup>2</sup>C bus busy flag
- Arbitration Lost Flag
- End of byte transmission flag
- Transmitter/Receiver Flag
- Start bit detection flag
- Start and Stop generation

#### I<sup>2</sup>C Slave Features:

- Stop bit detection
- I<sup>2</sup>C bus busy flag
- Detection of misplaced start or stop condition
- Programmable I<sup>2</sup>C Address detection
- Transfer problem detection
- End-of-byte transmission flag
- Transmitter/Receiver flag

### 11.6.3 General Description

In addition to receiving and transmitting data, this interface converts it from serial to parallel format and vice versa, using either an interrupt or polled

handshake. The interrupts are enabled or disabled by software. The interface is connected to the I<sup>2</sup>C bus by a data pin (SDA) and by a clock pin (SCL). It can be connected both with a standard I<sup>2</sup>C bus and a Fast I<sup>2</sup>C bus. This selection is made by software.

#### Mode Selection

The interface can operate in the four following modes:

- Slave transmitter/receiver
- Master transmitter/receiver

By default, it operates in slave mode.

The interface automatically switches from slave to master after it generates a START condition and from master to slave in case of arbitration loss or a STOP generation, allowing then Multi-Master capability.

#### Communication Flow

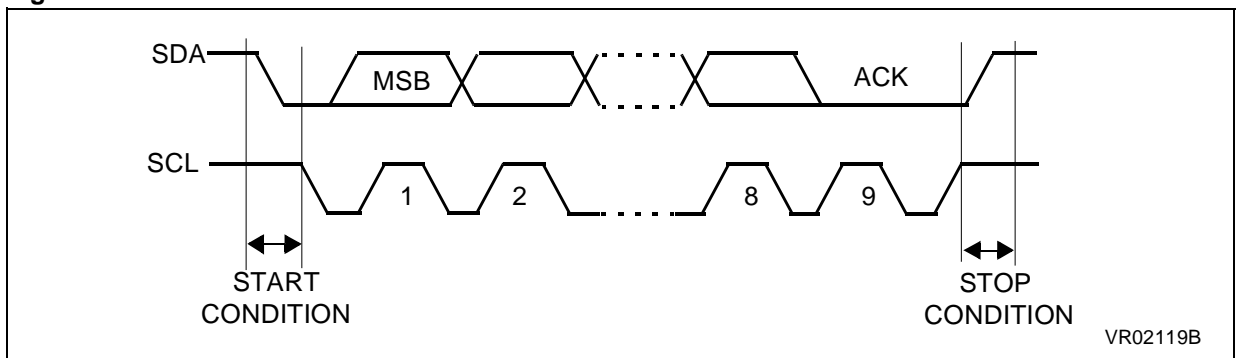
In Master mode, it initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition. Both start and stop conditions are generated in master mode by software.

In Slave mode, the interface is capable of recognising its own address (7 or 10-bit), and the General Call address. The General Call address detection may be enabled or disabled by software.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the start condition contain the address (one in 7-bit mode, two in 10-bit mode). The address is always transmitted in Master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledge bit to the transmitter. Refer to Figure 57.

Figure 57. I<sup>2</sup>C BUS Protocol



**I<sup>2</sup>C BUS INTERFACE** (Cont'd)

Acknowledge may be enabled and disabled by software.

The I<sup>2</sup>C interface address and/or general call address can be selected by software.

The speed of the I<sup>2</sup>C interface may be selected between Standard (0-100KHz) and Fast I<sup>2</sup>C (100-400KHz).

**SDA/SCL Line Control**

Transmitter mode: the interface holds the clock line low before transmission to wait for the microcontroller to write the byte in the Data Register.

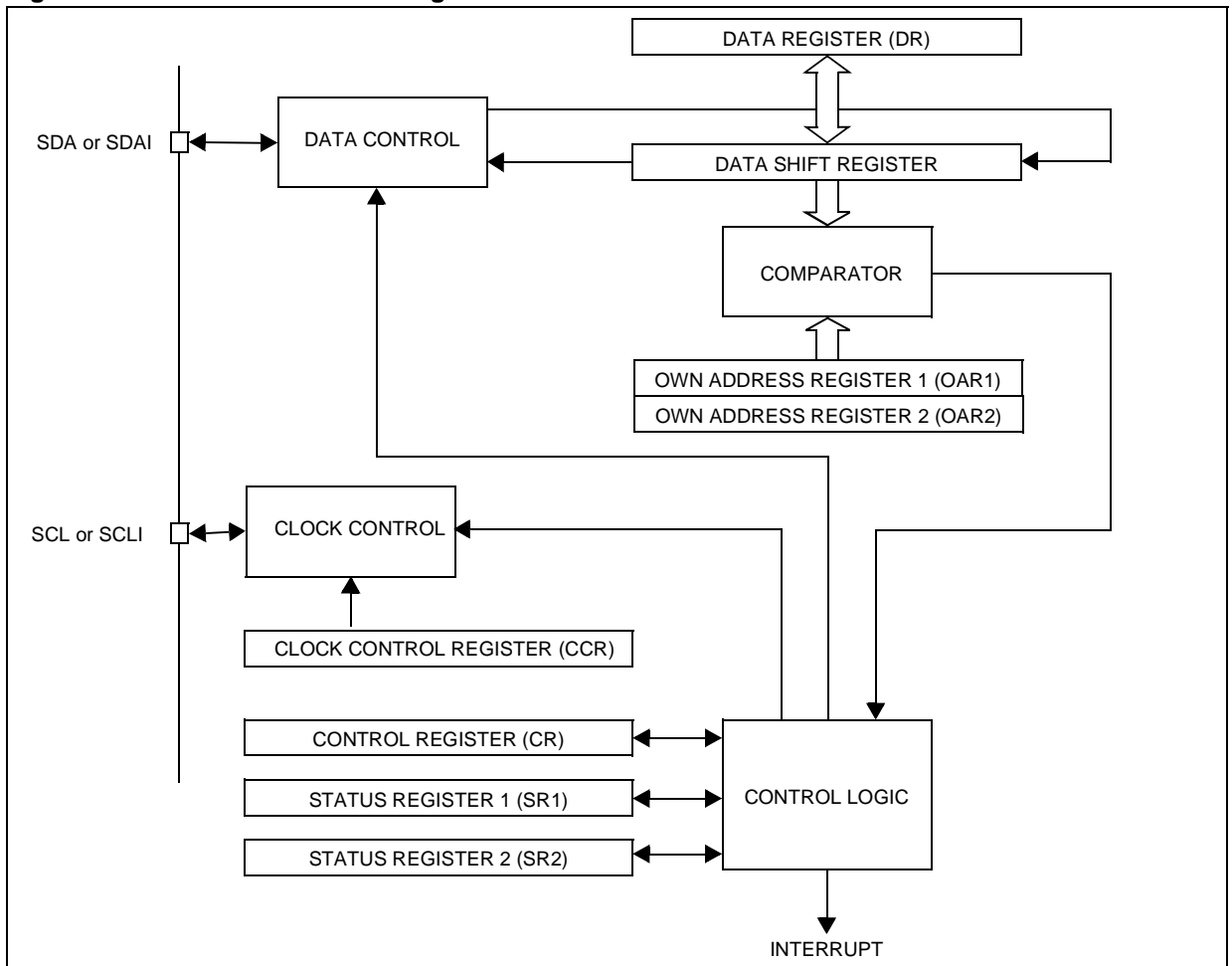
Receiver mode: the interface holds the clock line low after reception to wait for the microcontroller to read the byte in the Data Register.

The SCL frequency ( $F_{SCL}$ ) is controlled by a programmable clock divider which depends on the I<sup>2</sup>C bus mode.

When the I<sup>2</sup>C cell is enabled, the SDA and SCL ports must be configured as floating inputs. In this case, the value of the external pull-up resistor used depends on the application.

When the I<sup>2</sup>C cell is disabled, the SDA and SCL ports revert to being standard I/O port pins.

**Figure 58. I<sup>2</sup>C Interface Block Diagram**



## I<sup>2</sup>C BUS INTERFACE (Cont'd)

### 11.6.4 Functional Description

Refer to the CR, SR1 and SR2 registers in Section 11.6.7. for the bit definitions.

By default the I<sup>2</sup>C interface operates in Slave mode (M/SL bit is cleared) except when it initiates a transmit or receive sequence.

First the interface frequency must be configured using the FRi bits in the OAR2 register.

#### 11.6.4.1 Slave Mode

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register; then it is compared with the address of the interface or the General Call address (if selected by software).

**Note:** In 10-bit addressing mode, the comparison includes the header sequence (11110xx0) and the two most significant bits of the address.

**Header matched** (10-bit mode only): the interface generates an acknowledge pulse if the ACK bit is set.

**Address not matched:** the interface ignores it and waits for another Start condition.

**Address matched:** the interface generates in sequence:

- Acknowledge pulse if the ACK bit is set.
- EVF and ADSL bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV1).

Next, in 7-bit mode read the DR register to determine from the least significant bit (Data Direction Bit) if the slave must enter Receiver or Transmitter mode.

In 10-bit mode, after receiving the address sequence the slave is always in receive mode. It will enter transmit mode on receiving a repeated Start condition followed by the header sequence with matching address bits and the least significant bit set (11110xx1) .

#### Slave Receiver

Following the address reception and after SR1 register has been read, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- Acknowledge pulse if the ACK bit is set

- EVF and BTF bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV2).

#### Slave Transmitter

Following the address reception and after SR1 register has been read, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV3).

When the acknowledge pulse is received:

- The EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

#### Closing slave communication

After the last data byte is transferred a Stop Condition is generated by the master. The interface detects this condition and sets:

- EVF and STOPF bits with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR2 register (see Figure 59 Transfer sequencing EV4).

#### Error Cases

- **BERR:** Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and the BERR bits are set with an interrupt if the ITE bit is set.

If it is a Stop then the interface discards the data, released the lines and waits for another Start condition.

If it is a Start then the interface discards the data and waits for the next slave address on the bus.

- **AF:** Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set with an interrupt if the ITE bit is set.

**Note:** In both cases, SCL line is not held low; however, SDA line can remain low due to possible «0» bits transmitted last. It is then necessary to release both lines by software.

## I<sup>2</sup>C BUS INTERFACE (Cont'd)

### How to release the SDA / SCL lines

Set and subsequently clear the STOP bit while BTF is set. The SDA/SCL lines are released after the transfer of the current byte.

#### 11.6.4.2 Master Mode

To switch from default Slave mode to Master mode a Start condition generation is needed.

#### Start condition

Setting the START bit while the BUSY bit is cleared causes the interface to switch to Master mode (M/SL bit set) and generates a Start condition.

Once the Start condition is sent:

- The EVF and SB bits are set by hardware with an interrupt if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the DR register with the Slave address, **holding the SCL line low** (see Figure 59 Transfer sequencing EV5).

#### Slave address transmission

Then the slave address is sent to the SDA line via the internal shift register.

In 7-bit addressing mode, one address byte is sent.

In 10-bit addressing mode, sending the first byte including the header sequence causes the following event:

- The EVF bit is set by hardware with interrupt generation if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV9).

Then the second address byte is sent by the interface.

After completion of this transfer (and acknowledge from the slave if the ACK bit is set):

- The EVF bit is set by hardware with interrupt generation if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the CR register (for example set PE bit), **holding the SCL line low** (see Figure 59 Transfer sequencing EV6).

Next the master must enter Receiver or Transmitter mode.

**Note:** In 10-bit addressing mode, to switch the master to Receiver mode, software must generate a repeated Start condition and resend the header sequence with the least significant bit set (11110xx1).

#### Master Receiver

Following the address transmission and after SR1 and CR registers have been accessed, the master receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- Acknowledge pulse if the ACK bit is set
- EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV7).

To close the communication: before reading the last byte from the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

**Note:** In order to generate the non-acknowledge pulse after the last received data byte, the ACK bit must be cleared just before reading the second last data byte.

## I<sup>2</sup>C BUS INTERFACE (Cont'd)

### Master Transmitter

Following the address transmission and after SR1 register has been read, the master sends bytes from the DR register to the SDA line via the internal shift register.

The master waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV8).

When the acknowledge bit is received, the interface sets:

- EVF and BTF bits with an interrupt if the ITE bit is set.

To close the communication: after writing the last byte to the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

### Error Cases

- **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and

BERR bits are set by hardware with an interrupt if ITE is set.

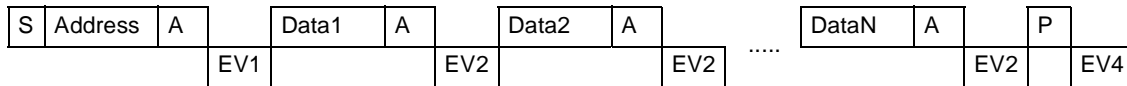
- **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set by hardware with an interrupt if the ITE bit is set. To resume, set the START or STOP bit.
- **ARLO**: Detection of an arbitration lost condition. In this case the ARLO bit is set by hardware (with an interrupt if the ITE bit is set and the interface goes automatically back to slave mode (the M/SL bit is cleared)).

**Note:** In all these cases, the SCL line is not held low; however, the SDA line can remain low due to possible «0» bits transmitted last. It is then necessary to release both lines by software.

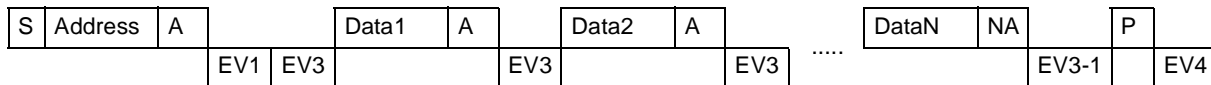
I<sup>2</sup>C BUS INTERFACE (Cont'd)

Figure 59. Transfer Sequencing

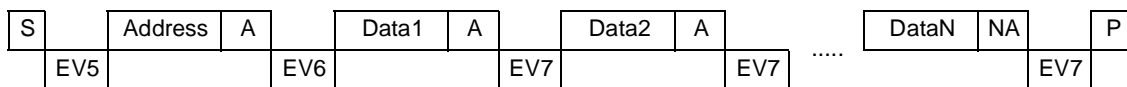
7-bit Slave receiver:



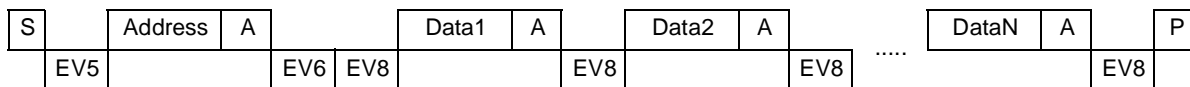
7-bit Slave transmitter:



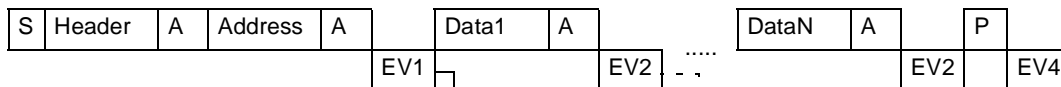
7-bit Master receiver:



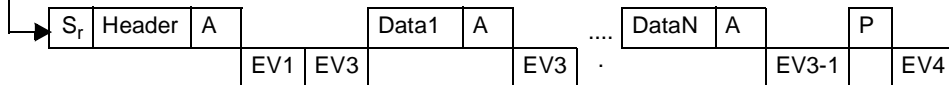
7-bit Master transmitter:



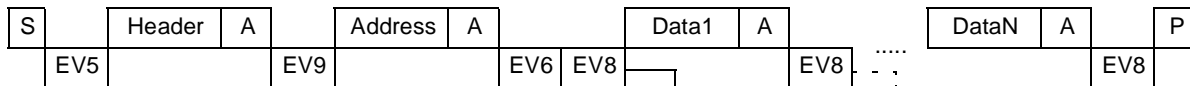
10-bit Slave receiver:



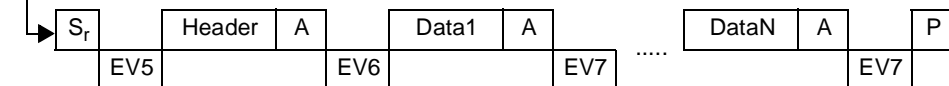
10-bit Slave transmitter:



10-bit Master transmitter



10-bit Master receiver:



**Legend:** S=Start, S<sub>r</sub> = Repeated Start, P=Stop, A=Acknowledge, NA=Non-acknowledge, EVx=Event (with interrupt if ITE=1)

**EV1:** EVF=1, ADSL=1, cleared by reading SR1 register.

**EV2:** EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

**EV3:** EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

**EV3-1:** EVF=1, AF=1, BTF=1; AF is cleared by reading SR1 register. BTF is cleared by releasing the lines (STOP=1, STOP=0) or by writing DR register (DR=FFh). **Note:** If lines are released by STOP=1, STOP=0, the subsequent EV4 is not seen.

**EV4:** EVF=1, STOPF=1, cleared by reading SR2 register.

**EV5:** EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.

**EV6:** EVF=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).

**EV7:** EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

**EV8:** EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

**EV9:** EVF=1, ADD10=1, cleared by reading SR1 register followed by writing DR register.



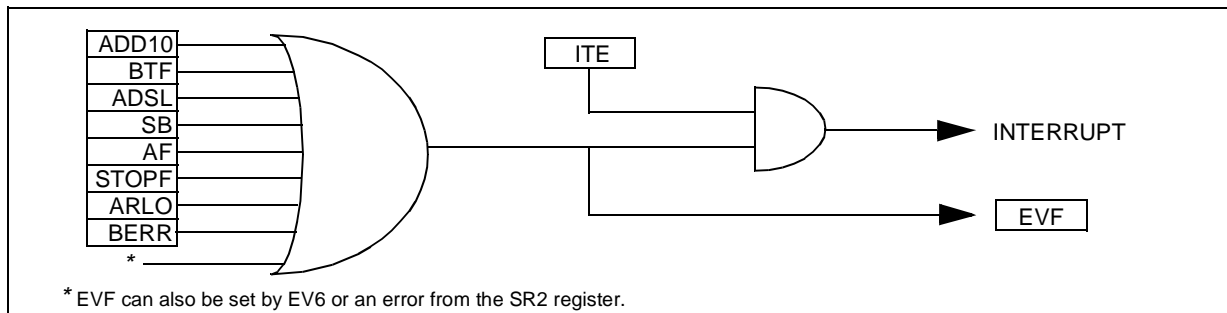
I<sup>2</sup>C BUS INTERFACE (Cont'd)

## 11.6.5 Low Power Modes

Mode	Description
WAIT	No effect on I <sup>2</sup> C interface. I <sup>2</sup> C interrupts cause the device to exit from WAIT mode.
HALT	I <sup>2</sup> C registers are frozen. In HALT mode, the I <sup>2</sup> C interface is inactive and does not acknowledge data on the bus. The I <sup>2</sup> C interface resumes operation when the MCU is woken up by an interrupt with "exit from HALT mode" capability.

## 11.6.6 Interrupts

Figure 60. Event Flags and Interrupt Generation



Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
10-bit Address Sent Event (Master mode)	ADD10	ITE	Yes	No
End of Byte Transfer Event	BTF		Yes	No
Address Matched Event (Slave mode)	ADSEL		Yes	No
Start Bit Generation Event (Master mode)	SB		Yes	No
Acknowledge Failure Event	AF		Yes	No
Stop Detection Event (Slave mode)	STOPF		Yes	No
Arbitration Lost Event (Multimaster configuration)	ARLO		Yes	No
Bus Error Event	BERR		Yes	No

**Note:** The I<sup>2</sup>C interrupt events are connected to the same interrupt vector (see Interrupts chapter). They generate an interrupt if the corresponding Enable Control Bit is set and the I-bit in the CC register is reset (RIM instruction).

**I<sup>2</sup>C BUS INTERFACE** (Cont'd)

**11.6.7 Register Description**

**I<sup>2</sup>C CONTROL REGISTER (CR)**

Read / Write

Reset Value: 0000 0000 (00h)

7							0
0	0	PE	ENGC	START	ACK	STOP	ITE

Bit 7:6 = Reserved. Forced to 0 by hardware.

Bit 5 = **PE** *Peripheral enable*.

This bit is set and cleared by software.

0: Peripheral disabled

1: Master/Slave capability

Notes:

- When PE=0, all the bits of the CR register and the SR register except the Stop bit are reset. All outputs are released while PE=0
- When PE=1, the corresponding I/O pins are selected by hardware as alternate functions.
- To enable the I<sup>2</sup>C interface, write the CR register **TWICE** with PE=1 as the first write only activates the interface (only PE is set).

Bit 4 = **ENGC** *Enable General Call*.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0). The 00h General Call address is acknowledged (01h ignored).

0: General Call disabled

1: General Call enabled

**Note:** In accordance with the I2C standard, when GCAL addressing is enabled, an I2C slave can only receive data. It will not transmit data to the master.

Bit 3 = **START** *Generation of a Start condition*.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0) or when the Start condition is sent (with interrupt generation if ITE=1).

- In master mode:

0: No start generation

1: Repeated start generation

- In slave mode:

0: No start generation

1: Start generation when the bus is free

Bit 2 = **ACK** *Acknowledge enable*.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0).

0: No acknowledge returned

1: Acknowledge returned after an address byte or a data byte is received

Bit 1 = **STOP** *Generation of a Stop condition*.

This bit is set and cleared by software. It is also cleared by hardware in master mode. Note: This bit is not cleared when the interface is disabled (PE=0).

- In master mode:

0: No stop generation

1: Stop generation after the current byte transfer or after the current Start condition is sent. The STOP bit is cleared by hardware when the Stop condition is sent.

- In slave mode:

0: No stop generation

1: Release the SCL and SDA lines after the current byte transfer (BTF=1). In this mode the STOP bit has to be cleared by software.

Bit 0 = **ITE** *Interrupt enable*.

This bit is set and cleared by software and cleared by hardware when the interface is disabled (PE=0).

0: Interrupts disabled

1: Interrupts enabled

Refer to Figure 60 for the relationship between the events and the interrupt.

SCL is held low when the ADD10, SB, BTF or ADSL flags or an EV6 event (See Figure 59) is detected.

**I<sup>2</sup>C BUS INTERFACE** (Cont'd)**I<sup>2</sup>C STATUS REGISTER 1 (SR1)**

Read Only

Reset Value: 0000 0000 (00h)

7							0
EVF	ADD10	TRA	BUSY	BTF	ADSL	M/SL	SB

**Bit 7 = EVF Event flag.**

This bit is set by hardware as soon as an event occurs. It is cleared by software reading SR2 register in case of error event or as described in Figure 59. It is also cleared by hardware when the interface is disabled (PE=0).

0: No event

1: One of the following events has occurred:

- BTF=1 (Byte received or transmitted)
- ADSL=1 (Address matched in Slave mode while ACK=1)
- SB=1 (Start condition generated in Master mode)
- AF=1 (No acknowledge received after byte transmission)
- STOPF=1 (Stop condition detected in Slave mode)
- ARLO=1 (Arbitration lost in Master mode)
- BERR=1 (Bus error, misplaced Start or Stop condition detected)
- ADD10=1 (Master has sent header byte)
- Address byte successfully transmitted in Master mode.

**Bit 6 = ADD10 10-bit addressing in Master mode.**

This bit is set by hardware when the master has sent the first byte in 10-bit address mode. It is cleared by software reading SR2 register followed by a write in the DR register of the second address byte. It is also cleared by hardware when the peripheral is disabled (PE=0).

0: No ADD10 event occurred.

1: Master has sent first address byte (header)

**Bit 5 = TRA Transmitter/Receiver.**

When BTF is set, TRA=1 if a data byte has been transmitted. It is cleared automatically when BTF is cleared. It is also cleared by hardware after detection of Stop condition (STOPF=1), loss of bus

arbitration (ARLO=1) or when the interface is disabled (PE=0).

0: Data byte received (if BTF=1)

1: Data byte transmitted

**Bit 4 = BUSY Bus busy.**

This bit is set by hardware on detection of a Start condition and cleared by hardware on detection of a Stop condition. It indicates a communication in progress on the bus. This information is still updated when the interface is disabled (PE=0).

0: No communication on the bus

1: Communication ongoing on the bus

**Bit 3 = BTF Byte transfer finished.**

This bit is set by hardware as soon as a byte is correctly received or transmitted with interrupt generation if ITE=1. It is cleared by software reading SR1 register followed by a read or write of DR register. It is also cleared by hardware when the interface is disabled (PE=0).

– Following a byte transmission, this bit is set after reception of the acknowledge clock pulse. In case an address byte is sent, this bit is set only after the EV6 event (See Figure 59). BTF is cleared by reading SR1 register followed by writing the next byte in DR register.

– Following a byte reception, this bit is set after transmission of the acknowledge clock pulse if ACK=1. BTF is cleared by reading SR1 register followed by reading the byte from DR register.

The SCL line is held low while BTF=1.

0: Byte transfer not done

1: Byte transfer succeeded

**Bit 2 = ADSL Address matched (Slave mode).**

This bit is set by hardware as soon as the received slave address matched with the OAR register content or a general call is recognized. An interrupt is generated if ITE=1. It is cleared by software reading SR1 register or by hardware when the interface is disabled (PE=0).

The SCL line is held low while ADSL=1.

0: Address mismatched or not received

1: Received address matched

**I<sup>2</sup>C BUS INTERFACE** (Cont'd)

Bit 1 = **M/SL** *Master/Slave*.

This bit is set by hardware as soon as the interface is in Master mode (writing START=1). It is cleared by hardware after detecting a Stop condition on the bus or a loss of arbitration (ARLO=1). It is also cleared when the interface is disabled (PE=0).

- 0: Slave mode
- 1: Master mode

Bit 0 = **SB** *Start bit (Master mode)*.

This bit is set by hardware as soon as the Start condition is generated (following a write START=1). An interrupt is generated if ITE=1. It is cleared by software reading SR1 register followed by writing the address byte in DR register. It is also cleared by hardware when the interface is disabled (PE=0).

- 0: No Start condition
- 1: Start condition generated

**I<sup>2</sup>C STATUS REGISTER 2 (SR2)**

Read Only

Reset Value: 0000 0000 (00h)

7								0
0	0	0	AF	STOPF	ARLO	BERR	GCAL	

Bit 7:5 = Reserved. Forced to 0 by hardware.

Bit 4 = **AF** *Acknowledge failure*.

This bit is set by hardware when no acknowledge is returned. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while AF=1.

- 0: No acknowledge failure
- 1: Acknowledge failure

Bit 3 = **STOPF** *Stop detection (Slave mode)*.

This bit is set by hardware when a Stop condition is detected on the bus after an acknowledge (if ACK=1). An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while STOPF=1.

- 0: No Stop condition detected
- 1: Stop condition detected

Bit 2 = **ARLO** *Arbitration lost*.

This bit is set by hardware when the interface loses the arbitration of the bus to another master. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

After an ARLO event the interface switches back automatically to Slave mode (M/SL=0).

The SCL line is not held low while ARLO=1.

- 0: No arbitration lost detected
- 1: Arbitration lost detected

Bit 1 = **BERR** *Bus error*.

This bit is set by hardware when the interface detects a misplaced Start or Stop condition. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while BERR=1.

- 0: No misplaced Start or Stop condition
- 1: Misplaced Start or Stop condition

Bit 0 = **GCAL** *General Call (Slave mode)*.

This bit is set by hardware when a general call address is detected on the bus while ENGC=1. It is cleared by hardware detecting a Stop condition (STOPF=1) or when the interface is disabled (PE=0).

- 0: No general call address detected on bus
- 1: general call address detected on bus

**I<sup>2</sup>C BUS INTERFACE** (Cont'd)**I<sup>2</sup>C CLOCK CONTROL REGISTER (CCR)**

Read / Write

Reset Value: 0000 0000 (00h)

7							0
FM/SM	CC6	CC5	CC4	CC3	CC2	CC1	CC0

Bit 7 = **FM/SM** *Fast/Standard I<sup>2</sup>C mode*.

This bit is set and cleared by software. It is not cleared when the interface is disabled (PE=0).

0: Standard I<sup>2</sup>C mode1: Fast I<sup>2</sup>C modeBit 6:0 = **CC[6:0]** *7-bit clock divider*.These bits select the speed of the bus (F<sub>SCL</sub>) depending on the I<sup>2</sup>C mode. They are not cleared when the interface is disabled (PE=0).– Standard mode (FM/SM=0): F<sub>SCL</sub> ≤ 100kHz

$$F_{SCL} = F_{CPU} / (2 \times ([CC6..CC0] + 2))$$

– Fast mode (FM/SM=1): F<sub>SCL</sub> > 100kHz

$$F_{SCL} = F_{CPU} / (3 \times ([CC6..CC0] + 2))$$

Note: The programmed F<sub>SCL</sub> assumes no load on SCL and SDA lines.**I<sup>2</sup>C DATA REGISTER (DR)**

Read / Write

Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7:0 = **D[7:0]** *8-bit Data Register*.

These bits contain the byte to be received or transmitted on the bus.

– Transmitter mode: Byte transmission start automatically when the software writes in the DR register.

– Receiver mode: the first data byte is received automatically in the DR register using the least significant bit of the address.

Then, the following data bytes are received one by one after reading the DR register.

**I<sup>2</sup>C BUS INTERFACE (Cont'd)**

**I<sup>2</sup>C OWN ADDRESS REGISTER (OAR1)**

Read / Write

Reset Value: 0000 0000 (00h)

7								0
ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0	

**7-bit Addressing Mode**

Bit 7:1 = **ADD[7:1]** *Interface address.*

These bits define the I<sup>2</sup>C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

Bit 0 = **ADD0** *Address direction bit.*

This bit is don't care, the interface acknowledges either 0 or 1. It is not cleared when the interface is disabled (PE=0).

Note: Address 01h is always ignored.

**10-bit Addressing Mode**

Bit 7:0 = **ADD[7:0]** *Interface address.*

These are the least significant bits of the I<sup>2</sup>C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

**I<sup>2</sup>C OWN ADDRESS REGISTER (OAR2)**

Read / Write

Reset Value: 0100 0000 (40h)

7								0
FR1	FR0	0	0	0	ADD9	ADD8	0	

Bit 7:6 = **FR[1:0]** *Frequency bits.*

These bits are set by software only when the interface is disabled (PE=0). To configure the interface to I<sup>2</sup>C specified delays select the value corresponding to the microcontroller frequency F<sub>CPU</sub>.

f <sub>CPU</sub>	FR1	FR0
< 6 MHz	0	0
6 to 8 MHz	0	1

Bit 5:3 = Reserved

Bit 2:1 = **ADD[9:8]** *Interface address.*

These are the most significant bits of the I<sup>2</sup>C bus address of the interface (10-bit mode only). They are not cleared when the interface is disabled (PE=0).

Bit 0 = Reserved.

I<sup>2</sup>C BUS INTERFACE (Cont'd)Table 21. I<sup>2</sup>C Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0028h	<b>I2CCR</b> Reset Value	0	0	PE 0	ENGC 0	START 0	ACK 0	STOP 0	ITE 0
0029h	<b>I2CSR1</b> Reset Value	EVF 0	ADD10 0	TRA 0	BUSY 0	BTF 0	ADSL 0	M/SL 0	SB 0
002Ah	<b>I2CSR2</b> Reset Value	0	0	0	AF 0	STOPF 0	ARLO 0	BERR 0	GCAL 0
02Bh	<b>I2CCCR</b> Reset Value	FM/SM 0	CC6 0	CC5 0	CC4 0	CC3 0	CC2 0	CC1 0	CC0 0
02Ch	<b>I2COAR1</b> Reset Value	ADD7 0	ADD6 0	ADD5 0	ADD4 0	ADD3 0	ADD2 0	ADD1 0	ADD0 0
002Dh	<b>I2COAR2</b> Reset Value	FR1 0	FR0 1	0	0	0	ADD9 0	ADD8 0	0
002Eh	<b>I2CDR</b> Reset Value	MSB 0	0	0	0	0	0	0	LSB 0

### 11.7 10-BIT A/D CONVERTER (ADC)

#### 11.7.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 10-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has 6 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from 6 different sources.

The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

#### 11.7.2 Main Features

- 10-bit conversion
- 6 channels with multiplexed input
- Linear successive approximation

- Data register (DR) which contains the results
- Conversion complete status flag
- On/off bit (to reduce consumption)

The block diagram is shown in Figure 61.

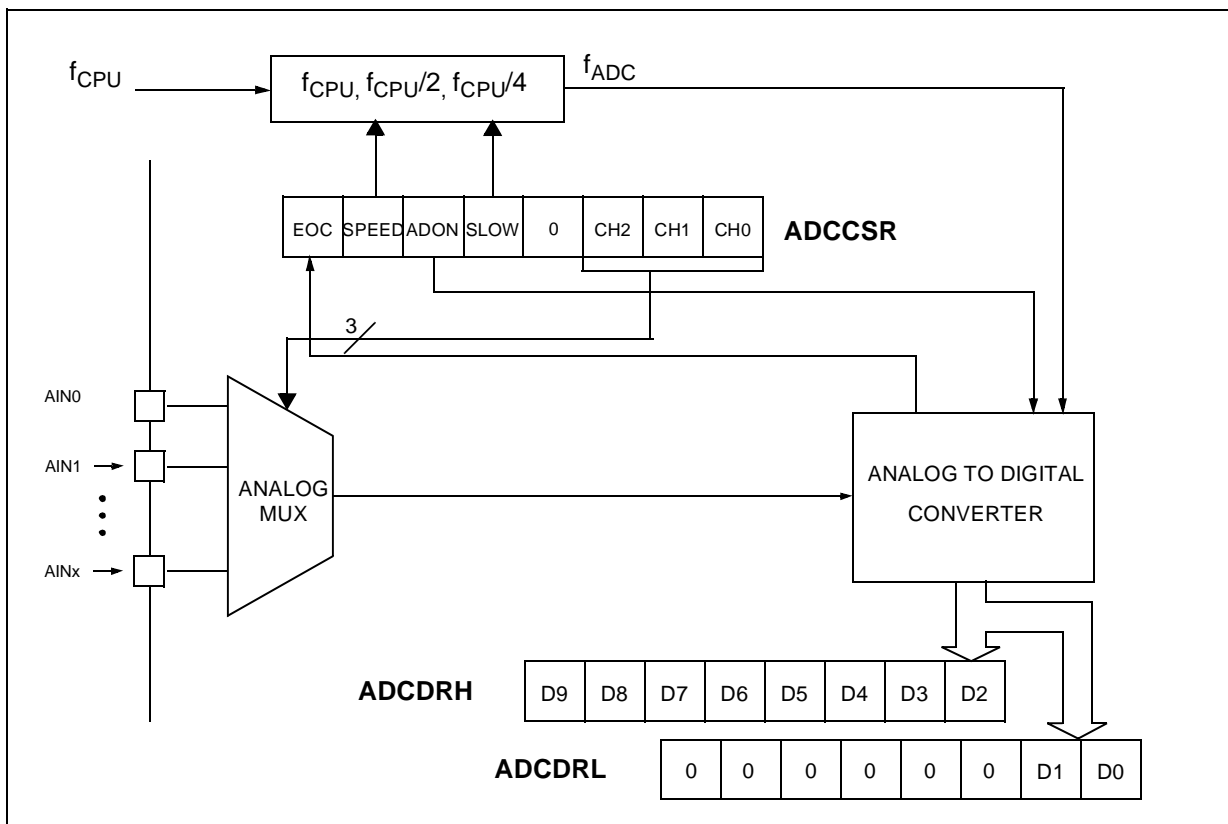
#### 11.7.3 Functional Description

##### 11.7.3.1 Analog Power Supply

$V_{DDA}$  and  $V_{SSA}$  are the high and low level reference voltage pins. In some devices (refer to device pin out description) they are internally connected to the  $V_{DD}$  and  $V_{SS}$  pins.

Conversion accuracy may therefore be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines.

Figure 61. ADC Block Diagram





## 10-BIT A/D CONVERTER (ADC) (Cont'd)

### 11.7.3.2 Digital A/D Conversion Result

The conversion is monotonic, meaning that the result never decreases if the analog input does not and never increases if the analog input does not.

If the input voltage ( $V_{AIN}$ ) is greater than  $V_{DDA}$  (high-level voltage reference) then the conversion result is FFh in the ADCDRH register and 03h in the ADCDRL register (without overflow indication).

If the input voltage ( $V_{AIN}$ ) is lower than  $V_{SSA}$  (low-level voltage reference) then the conversion result in the ADCDRH and ADCDRL registers is 00 00h.

The A/D converter is linear and the digital result of the conversion is stored in the ADCDRH and ADCDRL registers. The accuracy of the conversion is described in the Electrical Characteristics Section.

$R_{AIN}$  is the maximum recommended impedance for an analog input signal. If the impedance is too high, this will result in a loss of accuracy due to leakage and sampling not being completed in the allotted time.

### 11.7.3.3 A/D Conversion

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to the «I/O ports» chapter. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

In the ADCCSR register:

- Select the CH[2:0] bits to assign the analog channel to convert.

### ADC Conversion mode

In the ADCCSR register:

- Set the SPEED or the SLOW bits
- Set the ADON bit to enable the A/D converter and to start the conversion. From this time on, the ADC performs a continuous conversion of the selected channel.

When a conversion is complete:

- The EOC bit is set by hardware.
- The result is in the ADCDR registers.

A read to the ADCDRH or a write to any bit of the ADCCSR resets the EOC bit.

To read the 10 bits, perform the following steps:

1. Poll EOC bit
2. Read ADCDRL. This locks the ADCDRH until it is read.
3. Read ADCDRH. This clears EOC automatically.

To read only 8 bits, perform the following steps:

1. Poll EOC bit
2. Read ADCDRH. This clears EOC automatically.

### 11.7.4 Low Power Modes

**Note:** The A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed and between single shot conversions.

Mode	Description
WAIT	No effect on A/D Converter
HALT	A/D Converter disabled. After wakeup from Halt mode, the A/D Converter requires a stabilisation time $t_{STAB}$ (see Electrical Characteristics) before accurate conversions can be performed.

### 11.7.5 Interrupts

None.

10-BIT A/D CONVERTER (ADC) (Cont'd)

11.7.6 Register Description

CONTROL/STATUS REGISTER (ADCCSR)

Read/Write (Except bit 7 read only)

Reset Value: 0000 0000 (00h)

							7	0
EOC	SPEED	ADON	SLOW	0	CH2	CH1	CH0	

Bit 7 = **EOC** *End of Conversion*

This bit is set by hardware. It is cleared by software reading the ADCDRH register or writing to any bit of the ADCCSR register.

0: Conversion is not complete

1: Conversion complete

Bit 6 = **SPEED** *A/D clock selection*

This bit is set and cleared by software.

Table 22. A/D Clock Selection (See Note 1)

f <sub>ADC</sub> Frequency	SLOW	SPEED
f <sub>CPU</sub> (See Note 2)	0	1
f <sub>CPU</sub> /2	1	1
	0	0
f <sub>CPU</sub> /4	1	0

1)The SPEED and SLOW bits must be updated before setting the ADON bit.

2)Use this setting only if f<sub>CPU</sub> ≤ 4 MHz

Bit 5 = **ADON** *A/D Converter on*

This bit is set and cleared by software.

0: Disable ADC and stop conversion

1: Enable ADC and start conversion

Bit 4 = **SLOW** *A/D Clock Selection*

This bit is set and cleared by software. It works together with the SPEED bit. Refer to Table 22.

Bit 2:0 = **CH[2:0]** *Channel Selection*

These bits are set and cleared by software. They select the analog input to convert.

Channel Pin	CH2	CH1	CH0
AIN0	0	0	0
AIN1	0	0	1
AIN2	0	1	0
AIN3	0	1	1
AIN4	1	0	0
AIN5	1	0	1

DATA REGISTER (ADCDRH)

Read Only

Reset Value: 0000 0000 (00h)

							7	0
D9	D8	D7	D6	D5	D4	D3	D2	

Bit 7:0 = **D[9:2]** *MSB of Analog Converted Value*

DATA REGISTER (ADCDRL)

Read Only

Reset Value: 0000 0000 (00h)

							7	0
0	0	0	0	0	0	D1	D0	

Bit 7:2 = Reserved. Forced by hardware to 0.

Bit 1:0 = **D[1:0]** *LSB of Analog Converted Value*

## 10-BIT A/D CONVERTER (ADC) (Cont'd)

Table 23. ADC Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
006Fh	<b>ADCDRL</b> Reset Value	0	0	0	0	0	0	D1 0	D0 0
0070h	<b>ADCDRH</b> Reset Value	D9 0	D8 0	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0
0071h	<b>ADCCSR</b> Reset Value	EOC 0	SPEED 0	ADON 0	SLOW 0	0	CH2 0	CH1 0	CH0 0

## 12 INSTRUCTION SET

### 12.1 CPU ADDRESSING MODES

The CPU features 17 different addressing modes which can be classified in 7 main groups:

Addressing Mode	Example
Inherent	nop
Immediate	ld A,#\$55
Direct	ld A,\$55
Indexed	ld A,(\$55,X)
Indirect	ld A,[\$55],X)
Relative	jrne loop
Bit operation	bset byte,#5

The CPU Instruction set is designed to minimize the number of bytes required per instruction: To do

so, most of the addressing modes may be subdivided in two sub-modes called long and short:

- Long addressing mode is more powerful because it can use the full 64 Kbyte address space, however it uses more bytes and more CPU cycles.
- Short addressing mode is less powerful because it can generally only access page zero (0000h - 00FFh range), but the instruction size is more compact, and faster. All memory to memory instructions use short addressing modes only (CLR, CPL, NEG, BSET, BRES, BTJT, BTJF, INC, DEC, RLC, RRC, SLL, SRL, SRA, SWAP)

The ST7 Assembler optimizes the use of long and short addressing modes.

**Table 24. CPU Addressing Mode Overview**

Mode		Syntax	Destination	Pointer Address (Hex.)	Pointer Size (Hex.)	Length (Bytes)
Inherent		nop				+ 0
Immediate		ld A,#\$55				+ 1
Short	Direct	ld A,\$10	00..FF			+ 1
Long	Direct	ld A,\$1000	0000..FFFF			+ 2
No Offset	Direct	Indexed ld A,(X)	00..FF			+ 0
Short	Direct	Indexed ld A,(\$10,X)	00..1FE			+ 1
Long	Direct	Indexed ld A,(\$1000,X)	0000..FFFF			+ 2
Short	Indirect	ld A,[\$10]	00..FF	00..FF	byte	+ 2
Long	Indirect	ld A,[\$10.w]	0000..FFFF	00..FF	word	+ 2
Short	Indirect	Indexed ld A,[\$10],X)	00..1FE	00..FF	byte	+ 2
Long	Indirect	Indexed ld A,[\$10.w],X)	0000..FFFF	00..FF	word	+ 2
Relative	Direct	jrne loop	PC+/-127			+ 1
Relative	Indirect	jrne [\$10]	PC+/-127	00..FF	byte	+ 2
Bit	Direct	bset \$10,#7	00..FF			+ 1
Bit	Indirect	bset [\$10],#7	00..FF	00..FF	byte	+ 2
Bit	Direct	Relative btjt \$10,#7,skip	00..FF			+ 2
Bit	Indirect	Relative btjt [\$10],#7,skip	00..FF	00..FF	byte	+ 3

**INSTRUCTION SET OVERVIEW (Cont'd)****12.1.1 Inherent**

All Inherent instructions consist of a single byte. The opcode fully specifies all the required information for the CPU to process the operation.

Inherent Instruction	Function
NOP	No operation
TRAP	S/W Interrupt
WFI	Wait For Interrupt (Low Power Mode)
HALT	Halt Oscillator (Lowest Power Mode)
RET	Sub-routine Return
IRET	Interrupt Sub-routine Return
SIM	Set Interrupt Mask (level 3)
RIM	Reset Interrupt Mask (level 0)
SCF	Set Carry Flag
RCF	Reset Carry Flag
RSP	Reset Stack Pointer
LD	Load
CLR	Clear
PUSH/POP	Push/Pop to/from the stack
INC/DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
MUL	Byte Multiplication
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles

**12.1.2 Immediate**

Immediate instructions have two bytes, the first byte contains the opcode, the second byte contains the operand value.

Immediate Instruction	Function
LD	Load
CP	Compare
BCP	Bit Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Operations

**12.1.3 Direct**

In Direct instructions, the operands are referenced by their memory address.

The direct addressing mode consists of two sub-modes:

**Direct (short)**

The address is a byte, thus requires only one byte after the opcode, but only allows 00 - FF addressing space.

**Direct (long)**

The address is a word, thus allowing 64 Kbyte addressing space, but requires 2 bytes after the opcode.

**12.1.4 Indexed (No Offset, Short, Long)**

In this mode, the operand is referenced by its memory address, which is defined by the unsigned addition of an index register (X or Y) with an offset.

The indirect addressing mode consists of three sub-modes:

**Indexed (No Offset)**

There is no offset, (no extra byte after the opcode), and allows 00 - FF addressing space.

**Indexed (Short)**

The offset is a byte, thus requires only one byte after the opcode and allows 00 - 1FE addressing space.

**Indexed (long)**

The offset is a word, thus allowing 64 Kbyte addressing space and requires 2 bytes after the opcode.

**12.1.5 Indirect (Short, Long)**

The required data byte to do the operation is found by its memory address, located in memory (pointer).

The pointer address follows the opcode. The indirect addressing mode consists of two sub-modes:

**Indirect (short)**

The pointer address is a byte, the pointer size is a byte, thus allowing 00 - FF addressing space, and requires 1 byte after the opcode.

**Indirect (long)**

The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

**INSTRUCTION SET OVERVIEW (Cont'd)**

**12.1.6 Indirect Indexed (Short, Long)**

This is a combination of indirect and short indexed addressing modes. The operand is referenced by its memory address, which is defined by the unsigned addition of an index register value (X or Y) with a pointer value located in memory. The pointer address follows the opcode.

The indirect indexed addressing mode consists of two sub-modes:

**Indirect Indexed (Short)**

The pointer address is a byte, the pointer size is a byte, thus allowing 00 - 1FE addressing space, and requires 1 byte after the opcode.

**Indirect Indexed (Long)**

The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

**Table 25. Instructions Supporting Direct, Indexed, Indirect and Indirect Indexed Addressing Modes**

Long and Short Instructions	Function
LD	Load
CP	Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Additions/Subtractions operations
BCP	Bit Compare

Short Instructions Only	Function
CLR	Clear
INC, DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
BSET, BRES	Bit Operations
BTJT, BTJF	Bit Test and Jump Operations
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles
CALL, JP	Call or Jump subroutine

**12.1.7 Relative mode (Direct, Indirect)**

This addressing mode is used to modify the PC register value, by adding an 8-bit signed offset to it.

Available Relative Direct/Indirect Instructions	Function
JRxx	Conditional Jump
CALLR	Call Relative

The relative addressing mode consists of two sub-modes:

**Relative (Direct)**

The offset is following the opcode.

**Relative (Indirect)**

The offset is defined in memory, which address follows the opcode.

## INSTRUCTION SET OVERVIEW (Cont'd)

### 12.2 INSTRUCTION GROUPS

The ST7 family devices use an Instruction Set consisting of 63 instructions. The instructions may

be subdivided into 13 main groups as illustrated in the following table:

Load and Transfer	LD	CLR						
Stack operation	PUSH	POP	RSP					
Increment/Decrement	INC	DEC						
Compare and Tests	CP	TNZ	BCP					
Logical operations	AND	OR	XOR	CPL	NEG			
Bit Operation	BSET	BRES						
Conditional Bit Test and Branch	BTJT	BTJF						
Arithmetic operations	ADC	ADD	SUB	SBC	MUL			
Shift and Rotates	SLL	SRL	SRA	RLC	RRC	SWAP	SLA	
Unconditional Jump or Call	JRA	JRT	JRF	JP	CALL	CALLR	NOP	RET
Conditional Branch	JRxx							
Interruption management	TRAP	WFI	HALT	IRET				
Condition Code Flag modification	SIM	RIM	SCF	RCF				

#### Using a pre-byte

The instructions are described with one to four opcodes.

In order to extend the number of available opcodes for an 8-bit CPU (256 opcodes), three different prebyte opcodes are defined. These prebytes modify the meaning of the instruction they precede.

The whole instruction becomes:

PC-2            End of previous instruction  
 PC-1            Prebyte  
 PC               opcode  
 PC+1            Additional word (0 to 2) according to the number of bytes required to compute the effective address

These prebytes enable instruction in Y as well as indirect addressing modes to be implemented. They precede the opcode of the instruction in X or the instruction using direct addressing mode. The prebytes are:

PDY 90        Replace an X based instruction using immediate, direct, indexed, or inherent addressing mode by a Y one.

PIX 92        Replace an instruction using direct, direct bit, or direct relative addressing mode to an instruction using the corresponding indirect addressing mode.

It also changes an instruction using X indexed addressing mode to an instruction using indirect X indexed addressing mode.

PIY 91        Replace an instruction using X indirect indexed addressing mode by a Y one.

INSTRUCTION SET OVERVIEW (Cont'd)

Mnemo	Description	Function/Example	Dst	Src
ADC	Add with Carry	A = A + M + C	A	M
ADD	Addition	A = A + M	A	M
AND	Logical And	A = A . M	A	M
BCP	Bit compare A, Memory	tst (A . M)	A	M
BRES	Bit Reset	bres Byte, #3	M	
BSET	Bit Set	bset Byte, #3	M	
BTJF	Jump if bit is false (0)	btjf Byte, #3, Jmp1	M	
BTJT	Jump if bit is true (1)	btjt Byte, #3, Jmp1	M	
CALL	Call subroutine			
CALLR	Call subroutine relative			
CLR	Clear		reg, M	
CP	Arithmetic Compare	tst(Reg - M)	reg	M
CPL	One Complement	A = FFH-A	reg, M	
DEC	Decrement	dec Y	reg, M	
HALT	Halt			
IRET	Interrupt routine return	Pop CC, A, X, PC		
INC	Increment	inc X	reg, M	
JP	Absolute Jump	jp [TBL.w]		
JRA	Jump relative always			
JRT	Jump relative			
JRF	Never jump	jrf *		
JRIH	Jump if Port B INT pin = 1	(no Port B Interrupts)		
JRIL	Jump if Port B INT pin = 0	(Port B interrupt)		
JRH	Jump if H = 1	H = 1 ?		
JRNH	Jump if H = 0	H = 0 ?		
JRM	Jump if I1:0 = 11	I1:0 = 11 ?		
JRNM	Jump if I1:0 <> 11	I1:0 <> 11 ?		
JRMI	Jump if N = 1 (minus)	N = 1 ?		
JRPL	Jump if N = 0 (plus)	N = 0 ?		
JREQ	Jump if Z = 1 (equal)	Z = 1 ?		
JRNE	Jump if Z = 0 (not equal)	Z = 0 ?		
JRC	Jump if C = 1	C = 1 ?		
JRNC	Jump if C = 0	C = 0 ?		
JRULT	Jump if C = 1	Unsigned <		
JRUGE	Jump if C = 0	Jmp if unsigned >=		
JRUGT	Jump if (C + Z = 0)	Unsigned >		

I1	H	I0	N	Z	C
	H		N	Z	C
	H		N	Z	C
			N	Z	
			N	Z	
					C
					C
			0	1	
			N	Z	C
			N	Z	1
			N	Z	
1		0			
I1	H	I0	N	Z	C
			N	Z	





## INSTRUCTION SET OVERVIEW (Cont'd)

Mnemo	Description	Function/Example	Dst	Src	I1	H	I0	N	Z	C
JRULE	Jump if (C + Z = 1)	Unsigned <=								
LD	Load	dst <= src	reg, M	M, reg				N	Z	
MUL	Multiply	X,A = X * A	A, X, Y	X, Y, A		0				0
NEG	Negate (2's compl)	neg \$10	reg, M					N	Z	C
NOP	No Operation									
OR	OR operation	A = A + M	A	M				N	Z	
POP	Pop from the Stack	pop reg	reg	M						
		pop CC	CC	M	I1	H	I0	N	Z	C
PUSH	Push onto the Stack	push Y	M	reg, CC						
RCF	Reset carry flag	C = 0								0
RET	Subroutine Return									
RIM	Enable Interrupts	I1:0 = 10 (level 0)			1		0			
RLC	Rotate left true C	C <= A <= C	reg, M					N	Z	C
RRC	Rotate right true C	C => A => C	reg, M					N	Z	C
RSP	Reset Stack Pointer	S = Max allowed								
SBC	Subtract with Carry	A = A - M - C	A	M				N	Z	C
SCF	Set carry flag	C = 1								1
SIM	Disable Interrupts	I1:0 = 11 (level 3)			1		1			
SLA	Shift left Arithmetic	C <= A <= 0	reg, M					N	Z	C
SLL	Shift left Logic	C <= A <= 0	reg, M					N	Z	C
SRL	Shift right Logic	0 => A => C	reg, M					0	Z	C
SRA	Shift right Arithmetic	A7 => A => C	reg, M					N	Z	C
SUB	Substraction	A = A - M	A	M				N	Z	C
SWAP	SWAP nibbles	A7-A4 <=> A3-A0	reg, M					N	Z	
TNZ	Test for Neg & Zero	tnz  b1						N	Z	
TRAP	S/W trap	S/W interrupt			1		1			
WFI	Wait for Interrupt				1		0			
XOR	Exclusive OR	A = A XOR M	A	M				N	Z	

## 13 ELECTRICAL CHARACTERISTICS

### 13.1 PARAMETER CONDITIONS

Unless otherwise specified, all voltages are referred to  $V_{SS}$ .

#### 13.1.1 Minimum and Maximum values

Unless otherwise specified the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage and frequencies by tests in production on 100% of the devices with an ambient temperature at  $T_A=25^\circ\text{C}$  and  $T_A=T_{A\text{max}}$  (given by the selected temperature range).

Data based on characterization results, design simulation and/or technology characteristics are indicated in the table footnotes and are not tested in production. Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value plus or minus three times the standard deviation ( $\text{mean} \pm 3\sigma$ ).

#### 13.1.2 Typical values

Unless otherwise specified, typical data are based on  $T_A=25^\circ\text{C}$ ,  $V_{DD}=5\text{V}$  (for the  $3\text{V} \leq V_{DD} \leq 5.5\text{V}$  voltage range) and  $V_{DD}=2.7\text{V}$  (for the  $2.4\text{V} \leq V_{DD} \leq 3\text{V}$  voltage range). They are given only as design guidelines and are not tested.

Typical ADC accuracy values are determined by characterization of a batch of samples from a standard diffusion lot over the full temperature range, where 95% of the devices have an error less than or equal to the value indicated ( $\text{mean} \pm 2\sigma$ ).

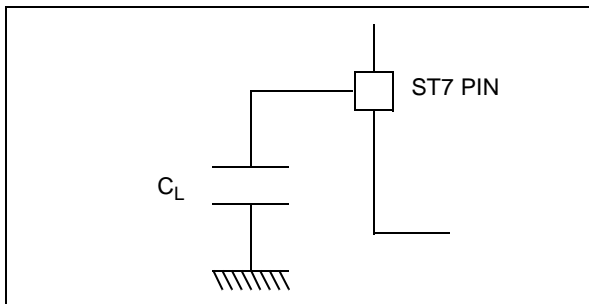
#### 13.1.3 Typical curves

Unless otherwise specified, all typical curves are given only as design guidelines and are not tested.

#### 13.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in Figure 62.

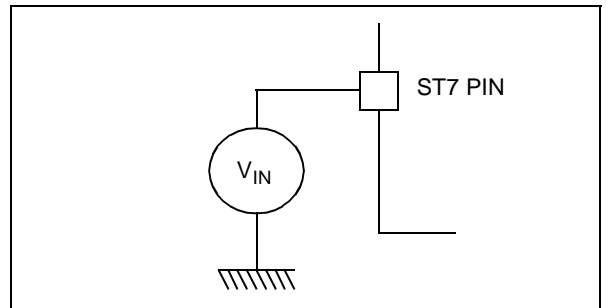
**Figure 62. Pin loading conditions**



#### 13.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in Figure 63.

**Figure 63. Pin input voltage**



## 13.2 ABSOLUTE MAXIMUM RATINGS

Stresses above those listed as “absolute maximum ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device under these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

tions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

### 13.2.1 Voltage Characteristics

Symbol	Ratings	Maximum value	Unit
$V_{DD} - V_{SS}$	Supply voltage	6.5	V
$V_{IN}$	Input voltage on any pin <sup>1) &amp; 2)</sup>	$V_{SS}-0.3$ to $V_{DD}+0.3$	
$V_{ESD(HBM)}$	Electrostatic discharge voltage (Human Body Model)	see Section 13.7.3 on page 137	
$V_{ESD(MM)}$	Electrostatic discharge voltage (Machine Model)		

### 13.2.2 Current Characteristics

Symbol	Ratings	Maximum value	Unit
$I_{VDD}$	Total current into $V_{DD}$ power lines (source) <sup>3)</sup>	100	mA
$I_{VSS}$	Total current out of $V_{SS}$ ground lines (sink) <sup>3)</sup>	150	
$I_{IO}$	Output current sunk by any standard I/O and control pin	25	
	Output current sunk by any high sink I/O pin	50	
	Output current source by any I/Os and control pin	- 25	
$I_{INJ(PIN)}$ <sup>2) &amp; 4)</sup>	Injected current on ICCSEL pin	± 5	
	Injected current on $\overline{RESET}$ pin	± 5	
	Injected current on OSC1 and OSC2 pins	± 5	
	Injected current on any other pin <sup>5) &amp; 6)</sup>	± 5	
$\Sigma I_{INJ(PIN)}$ <sup>2)</sup>	Total injected current (sum of all I/O and control pins) <sup>5)</sup>	± 20	

### 13.2.3 Thermal Characteristics

Symbol	Ratings	Value	Unit
$T_{STG}$	Storage temperature range	-65 to +150	°C
$T_J$	Maximum junction temperature (see Section Figure 111. "Low Profile Fine Pitch Ball Grid Array Package" on page 155)		

#### Notes:

- Directly connecting the  $\overline{RESET}$  and I/O pins to  $V_{DD}$  or  $V_{SS}$  could damage the device if an unintentional internal reset is generated or an unexpected change of the I/O configuration occurs (for example, due to a corrupted program counter). To guarantee safe operation, this connection has to be done through a pull-up or pull-down resistor (typical: 4.7kΩ for  $\overline{RESET}$ , 10kΩ for I/Os). Unused I/O pins must be tied in the same way to  $V_{DD}$  or  $V_{SS}$  according to their reset configuration.
- When the current limitation is not possible, the  $V_{IN}$  absolute maximum rating must be respected, otherwise refer to  $I_{INJ(PIN)}$  specification. A positive injection is induced by  $V_{IN} > V_{DD}$  while a negative injection is induced by  $V_{IN} < V_{SS}$ .
- All power ( $V_{DD}$ ) and ground ( $V_{SS}$ ) lines must always be connected to the external supply.
- Negative injection disturbs the analog performance of the device. See note in “10-BIT ADC CHARACTERISTICS” on page 152. For best reliability, it is recommended to avoid negative injection of more than 1.6mA.
- When several inputs are submitted to a current injection, the maximum  $\Sigma I_{INJ(PIN)}$  is the absolute sum of the positive and negative injected currents (instantaneous values). These results are based on characterisation with  $\Sigma I_{INJ(PIN)}$  maximum current injection on four I/O port pins of the device.
- True open drain I/O port pins do not accept positive injection.

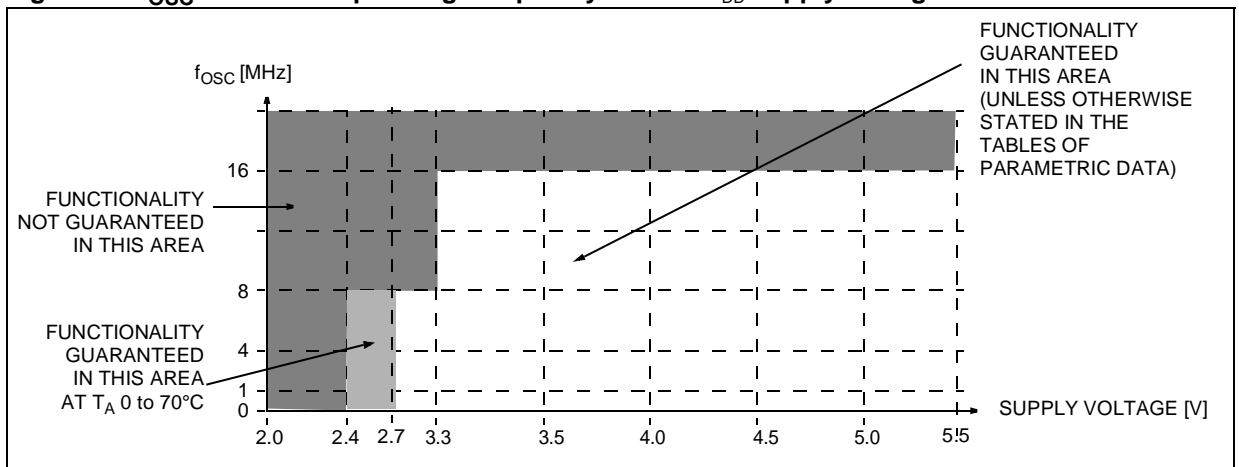
13.3 OPERATING CONDITIONS

13.3.1 General Operating Conditions

T<sub>A</sub> = -40 to +85°C unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max	Unit
V <sub>DD</sub>	Supply voltage	f <sub>OSC</sub> = 8 MHz. max., T <sub>A</sub> = 0 to 70°C	2.4	5.5	V
		f <sub>OSC</sub> = 8 MHz. max.	2.7	5.5	
		f <sub>OSC</sub> = 16 MHz. max.	3.3	5.5	
f <sub>OSC</sub>	External clock frequency on OSC1 pin	V <sub>DD</sub> ≥ 3.3V	0	16	MHz
		V <sub>DD</sub> ≥ 2.4V, T <sub>A</sub> = 0 to +70°C	0	8	
		V <sub>DD</sub> ≥ 2.7V, T <sub>A</sub> = -40 to +85°C			

Figure 64. f<sub>OSC</sub> Maximum Operating Frequency Versus V<sub>DD</sub> Supply Voltage



## OPERATING CONDITIONS (Cont'd)

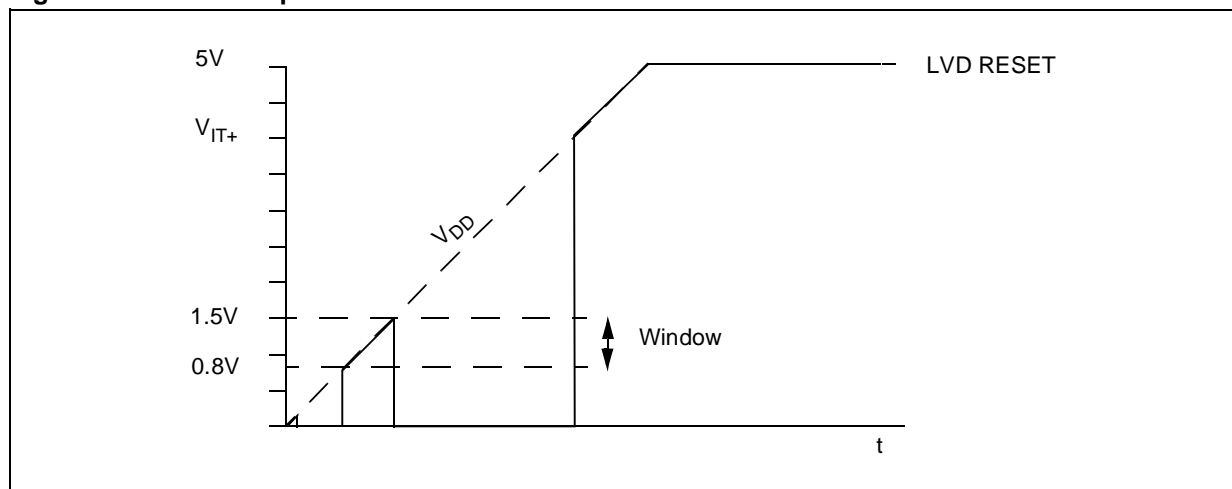
## 13.3.2 Low Voltage Detector (LVD) Thresholds

$T_A = -40$  to  $+85^\circ\text{C}$  unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(LVD)}$	Reset release threshold ( $V_{DD}$ rise)	High Threshold Med. Threshold Low Threshold	4.0 <sup>1)</sup> 3.55 <sup>1)</sup> 2.95 <sup>1)</sup>	4.2 3.75 3.15	4.5 4.0 3.35	V
$V_{IT-(LVD)}$	Reset generation threshold ( $V_{DD}$ fall)	High Threshold Med. Threshold Low Threshold	3.75 3.3 2.75	4.0 3.55 3.0	4.25 <sup>1)</sup> 3.75 <sup>1)</sup> 3.15 <sup>1)</sup>	V
$V_{hys(LVD)}$	LVD voltage threshold hysteresis	$V_{IT+(LVD)} - V_{IT-(LVD)}$	150	200	250	mV
$V_{tPOR}$	$V_{DD}$ rise time rate <sup>1)2)</sup>		20			$\mu\text{s/V}$
					100	ms/V
$t_g(V_{DD})$	Filtered glitch delay on $V_{DD}$ <sup>1)</sup>	Not detected by the LVD			40	ns

**Notes:**

1. Data based on characterization results, not tested in production.
2. When  $V_{tPOR}$  is faster than  $100 \mu\text{s/V}$ , the Reset signal is released after a delay of max.  $42\mu\text{s}$  after  $V_{DD}$  crosses the  $V_{IT+(LVD)}$  threshold.

**Figure 65. LVD Startup Behaviour**

**Note:** When the LVD is enabled, the MCU reaches its authorized operating voltage from a reset state. However, in some devices, the reset state is released when  $V_{DD}$  is approximately between 0.8V and 1.5V. As a consequence, depending on the ramp-up speed, the I/Os may toggle when  $V_{DD}$  is within this window.

This may be an issue especially for applications where the MCU drives power components.

Because Flash write access is impossible within this window, the Flash memory contents will not be corrupted.

## OPERATING CONDITIONS (Cont'd)

## 13.3.3 Auxiliary Voltage Detector (AVD) Thresholds

$T_A = -40$  to  $+85^\circ\text{C}$  unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(AVD)}$	1 $\Rightarrow$ 0 AVDF flag toggle threshold ( $V_{DD}$ rise)	VD level = Low in option byte VD level = Med. in option byte VD level = High in option byte	4.4 <sup>1)</sup> 3.9 <sup>1)</sup> 3.4 <sup>1)</sup>	4.6 4.2 3.6	4.9 4.4 3.8	V
$V_{IT-(AVD)}$	0 $\Rightarrow$ 1 AVDF flag toggle threshold ( $V_{DD}$ fall)	VD level = Low in option byte VD level = Med. in option byte VD level = High in option byte	4.15 3.75 3.1	4.4 3.95 3.4	4.65 <sup>1)</sup> 4.2 <sup>1)</sup> 3.6 <sup>1)</sup>	
$V_{hys(AVD)}$	AVD voltage threshold hysteresis	$V_{IT+(AVD)} - V_{IT-(AVD)}$		250		mV
$\Delta V_{IT-}$	Voltage drop between AVD flag set and LVD reset activated	$V_{IT-(AVD)} - V_{IT-(LVD)}$		450		

1. Data based on characterization results, not tested in production.

### 13.4 SUPPLY CURRENT CHARACTERISTICS

The following current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock source current consumption. To get the total device consumption, the two current values must be added (except for HALT mode for which the clock is stopped).

Symbol	Parameter	Conditions	Max	Unit
$\Delta I_{DD}(\Delta T_A)$	Supply current variation vs. temperature	Constant $V_{DD}$ and $f_{CPU}$	10	%

#### 13.4.1 RUN, SLOW, WAIT and SLOW WAIT Modes

$T_A = -40$  to  $+85^\circ\text{C}$  unless otherwise specified

Symbol	Parameter	Conditions	Typ	Max	Unit
$I_{DD}$	Supply current in RUN mode <sup>2)</sup> (see Figure 66)	$V_{DD}=5.5\text{V}, f_{OSC}=16\text{MHz}, f_{CPU}=8\text{MHz}$ $V_{DD}=2.7\text{V}, f_{OSC}=8\text{MHz}, f_{CPU}=4\text{MHz}$	7.2 3.5	11 <sup>1)</sup> 5.25 <sup>4)</sup>	mA
	Supply current in SLOW mode <sup>3)</sup> (see Figure 67)	$V_{DD}=5.5\text{V}, f_{OSC}=16\text{MHz}, f_{CPU}=500\text{kHz}$ $V_{DD}=2.7\text{V}, f_{OSC}=8\text{MHz}, f_{CPU}=250\text{MHz}$	0.7 0.38	1.2 <sup>1)</sup> 0.6 <sup>4)</sup>	
	Supply current in WAIT mode <sup>2)</sup> (see Figure 68)	$V_{DD}=5.5\text{V}, f_{OSC}=16\text{MHz}, f_{CPU}=8\text{MHz}$ $V_{DD}=2.7\text{V}, f_{OSC}=8\text{MHz}, f_{CPU}=4\text{MHz}$	3.6 1.8	5.55 <sup>1)</sup> 3 <sup>4)</sup>	
	Supply current in SLOW WAIT mode <sup>3)</sup> (see Figure 69)	$V_{DD}=5.5\text{V}, f_{OSC}=16\text{MHz}, f_{CPU}=500\text{kHz}$ $V_{DD}=2.7\text{V}, f_{OSC}=8\text{MHz}, f_{CPU}=250\text{MHz}$	0.45 0.25	1 <sup>1)</sup> 0.5 <sup>4)</sup>	

Figure 66. Typical  $I_{DD}$  in RUN at  $T_A=25^\circ\text{C}$

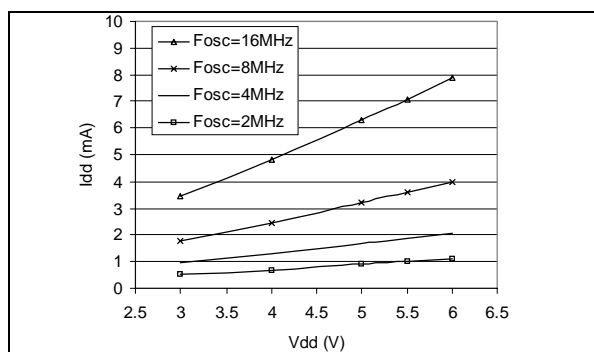


Figure 68. Typical  $I_{DD}$  in WAIT at  $T_A=25^\circ\text{C}$

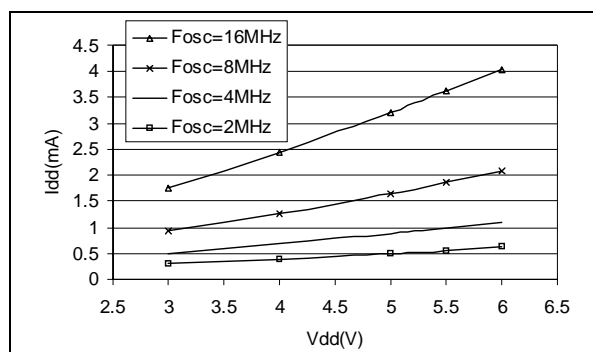


Figure 67. Typical  $I_{DD}$  in SLOW at  $T_A=25^\circ\text{C}$

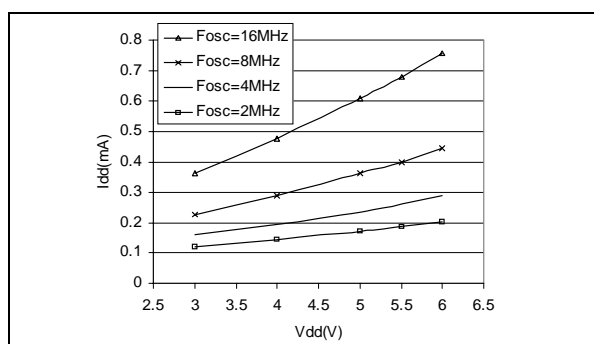
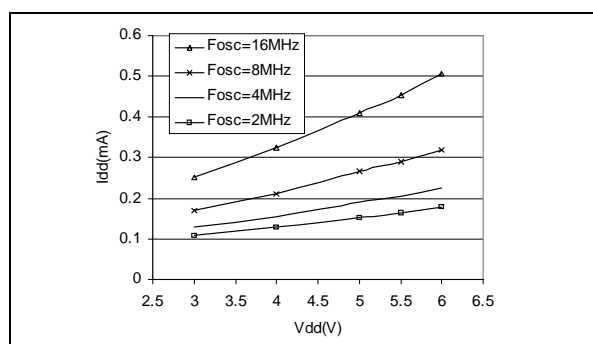


Figure 69. Typ.  $I_{DD}$  in SLOW-WAIT at  $T_A=25^\circ\text{C}$



#### Notes:

1. Data based on characterization results, tested in production at  $V_{DD}$  max. and  $f_{CPU}$  max.
2. Program executed from RAM, CPU running with memory access, all I/O pins in input mode with a static value at  $V_{DD}$  or  $V_{SS}$  (no load), all peripherals in reset state; clock input (OSC1) driven by external square wave, CSS and LVD disabled.
3. All I/O pins in input mode with a static value at  $V_{DD}$  or  $V_{SS}$  (no load), all peripherals in reset state; clock input (OSC1) driven by external square wave, CSS and LVD disabled.
4. Data based on characterization results, not tested in production.

**SUPPLY CURRENT CHARACTERISTICS** (Cont'd)**13.4.2 HALT and ACTIVE-HALT Modes**

Symbol	Parameter	Conditions		Typ	Max	Unit
I <sub>DD</sub>	Supply current in HALT mode <sup>1)</sup>	V <sub>DD</sub> =5.5V	-40°C ≤ T <sub>A</sub> ≤ +85°C		10	μA
		V <sub>DD</sub> =2.7V	-40°C ≤ T <sub>A</sub> ≤ +85°C		6	
	Supply current in ACTIVE-HALT mode <sup>2)</sup>			500	No max. guaranteed	

**13.4.3 Supply and Clock Managers**

The previous current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock

source current consumption. To get the total device consumption, the two current values must be added (except for HALT mode).

Symbol	Parameter	Conditions		Typ	Max	Unit
I <sub>DD(RCINT)</sub>	Supply current of internal RC oscillator			900		μA
I <sub>DD(RES)</sub>	Supply current of resonator oscillator <sup>3) &amp; 4)</sup>			see Section 13.5.3 on page 131		
I <sub>DD(PLL)</sub>	PLL supply current	V <sub>DD</sub> =5V		100		
I <sub>DD(CSS)</sub>	Clock security system supply current	V <sub>DD</sub> =5V		220		
I <sub>DD(LVD)</sub>	LVD supply current	HALT mode		100		

**Notes:**

1. All I/O pins in output mode with a static value at V<sub>SS</sub> (no load), CSS and LVD disabled. Data based on characterization results, tested in production at V<sub>DD</sub> max. and f<sub>CPU</sub> max.
2. Data based on characterisation results, not tested in production. All I/O pins in output mode with a static value at V<sub>SS</sub> (no load); clock input (OSC1) driven by external square wave, LVD disabled. To obtain the total current consumption of the device, add the clock source consumption (Section 13.5.3 and Section 13.5.4).
3. Data based on characterization results done with the external components specified in Section 13.5.3 and Section 13.5.4, not tested in production.
4. As the oscillator is based on a current source, the consumption does not depend on the voltage.



## SUPPLY CURRENT CHARACTERISTICS (Cont'd)

## 13.4.4 On-chip peripherals

Symbol	Parameter	Conditions		Typ	Unit
I <sub>DD</sub> (TIM)	16-bit Timer supply current <sup>1)</sup>	f <sub>CPU</sub> =4MHz	V <sub>DD</sub> =3.0V	TBD	μA
		f <sub>CPU</sub> =8MHz	V <sub>DD</sub> =5.0V	60	
I <sub>DD</sub> (SPI)	SPI supply current <sup>2)</sup>	f <sub>CPU</sub> =4MHz	V <sub>DD</sub> =3.0V	TBD	
		f <sub>CPU</sub> =8MHz	V <sub>DD</sub> =5.0V	200	
I <sub>DD</sub> (SCI)	SCI supply current <sup>3)</sup>	f <sub>CPU</sub> =4MHz	V <sub>DD</sub> =3.0V	TBD	
		f <sub>CPU</sub> =8MHz	V <sub>DD</sub> =5.0V	400	
I <sub>DD</sub> (I2C)	I2C supply current <sup>4)</sup>	f <sub>CPU</sub> =4MHz	V <sub>DD</sub> =3.0V	TBD	
		f <sub>CPU</sub> =8MHz	V <sub>DD</sub> =5.0V	500	
I <sub>DD</sub> (ADC)	ADC supply current when converting <sup>5)</sup>	f <sub>ADC</sub> =4MHz	V <sub>DD</sub> =3.0V	TBD	
			V <sub>DD</sub> =5.0V	500	

**Notes:**

1. Data based on a differential I<sub>DD</sub> measurement between reset configuration (timer counter running at f<sub>CPU</sub>/2) and timer counter stopped (only TIMD bit set). Data valid for one timer.
2. Data based on a differential I<sub>DD</sub> measurement between reset configuration (SPI disabled) and a permanent SPI master communication at maximum speed (data sent equal to FFh). This measurement includes the pad toggling consumption.
3. Data based on a differential I<sub>DD</sub> measurement between SCI running at maximum speed configuration (500 kbaud, continuous transmission of AA +RE enabled and SCI off). This measurement includes the pad toggling consumption.
4. Data based on a differential I<sub>DD</sub> measurement between reset configuration (I2C disabled) and a permanent I2C master communication at 300kHz (data sent equal to AAh). This measurement includes the pad toggling consumption (4.7kOhm external pull-up on clock and data lines).
5. Data based on a differential I<sub>DD</sub> measurement between reset configuration (ADC off) and continuous A/D conversion (f<sub>ADC</sub>=4MHz).

### 13.5 CLOCK AND TIMING CHARACTERISTICS

Subject to general operating conditions for  $V_{DD}$ ,  $f_{OSC}$ , and  $T_A$ .

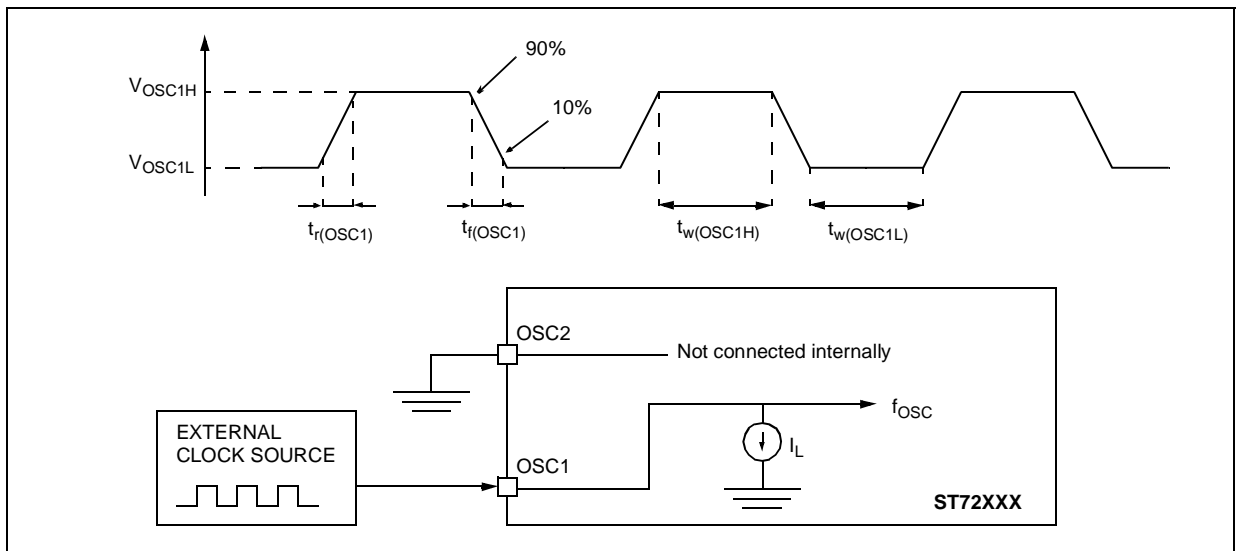
#### 13.5.1 General Timings

Symbol	Parameter	Conditions	Min	Typ <sup>1)</sup>	Max	Unit
$t_{c(INST)}$	Instruction cycle time		2	3	12	$t_{CPU}$
		$f_{CPU}=8MHz$	250	375	1500	ns
$t_{v(IT)}$	Interrupt reaction time <sup>2)</sup> $t_{v(IT)} = \Delta t_{c(INST)} + 10$		10		22	$t_{CPU}$
		$f_{CPU}=8MHz$	1.25		2.75	$\mu s$

#### 13.5.2 External Clock Source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{OSC1H}$	OSC1 input pin high level voltage	see Figure 70	$0.7 \times V_{DD}$		$V_{DD}$	V
$V_{OSC1L}$	OSC1 input pin low level voltage		$V_{SS}$		$0.3 \times V_{DD}$	
$t_{w(OSC1H)}$ $t_{w(OSC1L)}$	OSC1 high or low time <sup>3)</sup>		15			ns
$t_{r(OSC1)}$ $t_{f(OSC1)}$	OSC1 rise or fall time <sup>3)</sup>				15	
$I_L$	OSCx Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$			$\pm 1$	$\mu A$

Figure 70. Typical Application with an External Clock Source



**Notes:**

1. Data based on typical application software.
2. Time measured between interrupt event and interrupt vector fetch.  $\Delta t_{c(INST)}$  is the number of  $t_{CPU}$  cycles needed to finish the current instruction execution.
3. Data based on design simulation and/or technology characteristics, not tested in production.

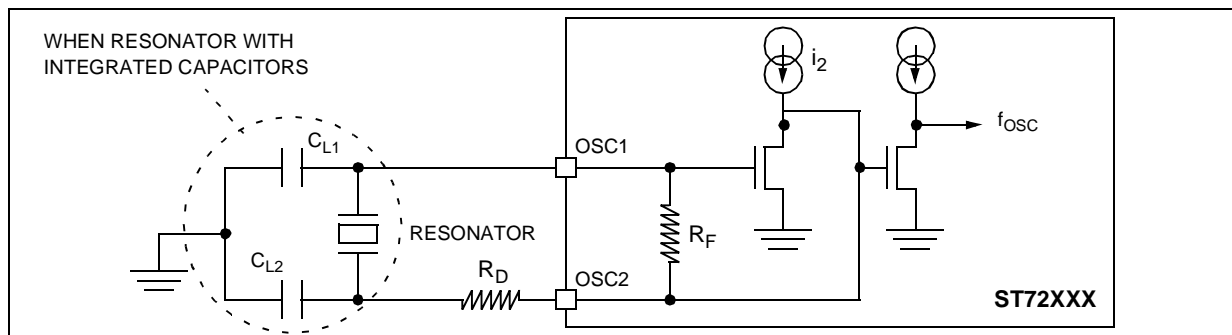
**CLOCK AND TIMING CHARACTERISTICS (Cont'd)****13.5.3 Crystal and Ceramic Resonator Oscillators**

The ST7 internal clock can be supplied with four different Crystal/Ceramic resonator oscillators. All the information given in this paragraph are based on characterization results with specified typical external components. In the application, the resonator and the load capacitors have to be placed as

close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. Refer to the crystal/ceramic resonator manufacturer for more details (frequency, package, accuracy...).

Symbol	Parameter	Conditions	Min	Max	Unit
$f_{OSC}$	Oscillator Frequency <sup>1)</sup>	VLP : Very Low power oscillator LP: Low power oscillator MP: Medium power oscillator MS: Medium speed oscillator HS: High speed oscillator	0.032 1 >2 >4 >8	0.1 2 4 8 16	MHz
$R_F$	Feedback resistor		20	40	k $\Omega$
$C_{L1}$ $C_{L2}$	Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator ( $R_S$ )	$R_S=200\Omega$ VLP oscillator $R_S=200\Omega$ LP oscillator $R_S=200\Omega$ MP oscillator $R_S=200\Omega$ MS oscillator $R_S=100\Omega$ HS oscillator	60 38 32 10 10	100 100 47 47 30	pF

Symbol	Parameter	Conditions	Typ	Max	Unit
$i_2$	OSC2 driving current	$V_{DD}=5V$ $V_{IN}=V_{SS}$ VLP oscillator LP oscillator MP oscillator MS oscillator HS oscillator	2.5 80 160 310 610	5 150 250 460 900	$\mu A$

**Figure 71. Typical Application with a Crystal or Ceramic Resonator****Notes:**

1. The oscillator selection can be optimized in terms of supply current using an high quality resonator with small  $R_S$  value. Refer to crystal/ceramic resonator manufacturer for more details.

## CLOCK AND TIMING CHARACTERISTICS (Cont'd)

Oscil.	Typical Crystal or Ceramic Resonators				C <sub>L1</sub> [pF]	C <sub>L2</sub> [pF]	R <sub>D</sub> [ohm]
	Reference <sup>2)</sup>	Type <sup>3)</sup>	Freq. (MHz)	Characteristic <sup>1)</sup>			
MURATA Ceramic	CSBFB1M00J58-R0	SMD	1	$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, 0.01\%_{\text{correl}}]$	100	100	3.3k
	CSBLA1M00J58-B0	LEAD		$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, 0.01\%_{\text{correl}}]$	100	100	3.3k
	CSTCC2M00G56-R0	SMD	2	$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, -0.19\%_{\text{correl}}]$	(47)	(47)	0
	CSTCR4M00G55-R0	SMD	4	$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.2\%_{\Delta T_a}, \pm 0.1\%_{\text{aging}}, -0.24\%_{\text{correl}}]$	(39)	(39)	0
	CSTLS4M00G56-B0	LEAD		$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.2\%_{\Delta T_a}, \pm 0.2\%_{\text{aging}}, 0.1\%_{\text{correl}}]$	(47)	(47)	0
	CSTCE8M00G52-R0	SMD	8	$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.2\%_{\Delta T_a}, \pm 0.1\%_{\text{aging}}, -0.16\%_{\text{correl}}]$	(10)	(10)	0
	CSTLS8M00G53-B0	LEAD		$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.2\%_{\Delta T_a}, \pm 0.2\%_{\text{aging}}, 0.13\%_{\text{correl}}]$	(15)	(15)	0
	CSTCE12M0G52-R0	SMD	12	$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.2\%_{\Delta T_a}, \pm 0.1\%_{\text{aging}}, -0.18\%_{\text{correl}}]$	(10)	(10)	0
	CSTLA12M0T55-B0	LEAD		$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, 0.29\%_{\text{correl}}]$	(30)	(30)	0
	CSTCE16M0V53-R0	SMD	16	$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, -0.08\%_{\text{correl}}]$	(15)	(15)	0
	CSALS16M0X55-B0	LEAD		$\Delta f_{OSC}=[\pm 0.5\%_{\text{tolerance}}, \pm 0.2\%_{\Delta T_a}, \pm 0.2\%_{\text{aging}}, 0.11\%_{\text{correl}}]$	10	10	470

**Notes:**

1. Resonator characteristics given by the crystal/ceramic resonator manufacturer.
2. CSTxx types have built-in loading capacitors (values shown in parentheses)
3. SMD = [-R0: Plastic tape package ( $\varnothing = 180\text{mm}$ ), -B0: Bulk]  
LEAD = [-A0: Flat pack package (Radial taping Ho= 18mm), -B0: Bulk]

For more information, please consult the Murata report (02gc\_5959\_ST72F264G2.zip) on [www.murata.com](http://www.murata.com) or <http://mcu.st.com>

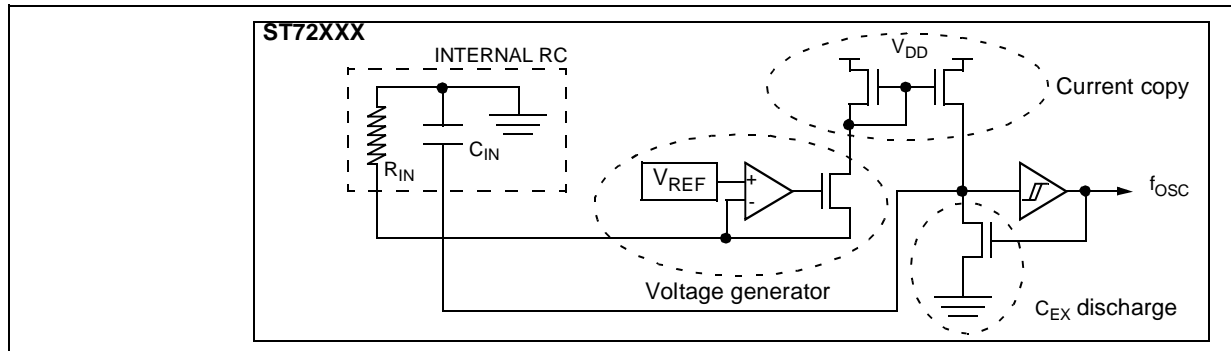
**CLOCK CHARACTERISTICS (Cont'd)**

**13.5.4 RC Oscillators**

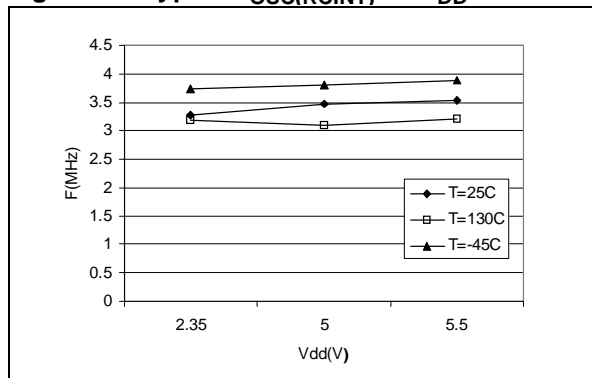
The ST7 internal clock can be supplied with an internal RC oscillator.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f <sub>OSC (RCINT)</sub>	Internal RC oscillator frequency See Figure 73	T <sub>A</sub> =25°C, V <sub>DD</sub> =5V	2	3.5	6	MHz

**Figure 72. Typical Application with RC oscillator**



**Figure 73. Typical f<sub>OSC(RCINT)</sub> vs V<sub>DD</sub>**



**CLOCK CHARACTERISTICS** (Cont'd)

**13.5.5 Clock Security System (CSS)**

T<sub>A</sub> = -40 to +85°C unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f <sub>SFOSC</sub>	Safe Oscillator Frequency <sup>1)</sup>			3		MHz

**Note:**

1. Data based on characterisation results.

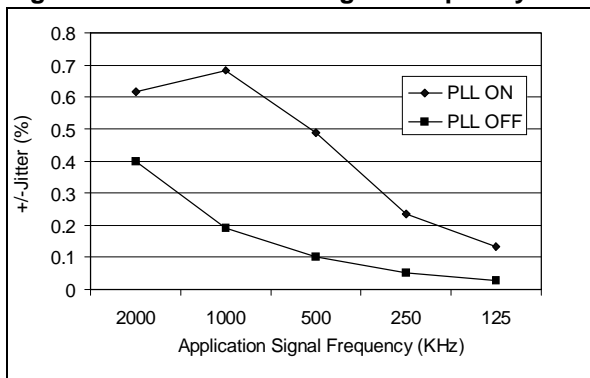
**13.5.6 PLL Characteristics**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V <sub>DD(PLL)</sub>	PLL Operating Range	T <sub>A</sub> 0 to 70°C	3.5		5.5	V
		T <sub>A</sub> -40 to +85°C	4.5		5.5	
f <sub>OSC</sub>	PLL input frequency range		2		4	MHz
Δ f <sub>CPU</sub> /f <sub>CPU</sub>	Instantaneous PLL jitter <sup>1)</sup>	f <sub>OSC</sub> = 4 MHz.		1.0	2.5	%
		f <sub>OSC</sub> = 2 MHz.		2.5	4.0	%

**Note:**

1. Data characterized but not tested.

**Figure 74. PLL Jitter vs. Signal frequency<sup>1</sup>**



The user must take the PLL jitter into account in the application (for example in serial communication or sampling of high frequency signals). The PLL jitter is a periodic effect, which is integrated over several CPU cycles. Therefore the longer the period of the application signal, the less it will be impacted by the PLL jitter.

Figure 74 shows the PLL jitter integrated on application signals in the range 125kHz to 2MHz. At frequencies of less than 125KHz, the jitter is negligible.

**Note 1:** Measurement conditions: f<sub>CPU</sub> = 4MHz, T<sub>A</sub> = 25°C

## 13.6 MEMORY CHARACTERISTICS

### 13.6.1 RAM and Hardware Registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{RM}$	Data retention mode <sup>1)</sup>	HALT mode (or RESET)	1.6			V

### 13.6.2 XFlash Program Memory

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{DD}$	Operating voltage for Flash write/erase		2.4		5.5	V
$t_{prog}$	Programming time for 1~32 bytes <sup>2)</sup>	$T_A = -40$ to $+85^\circ\text{C}$		5	10	ms
	Programming time for 1.5kBytes	$T_A = +25^\circ\text{C}$		0.24	0.48	
$t_{RET}$	Data retention <sup>4)</sup>	$T_A = +55^\circ\text{C}$ <sup>3)</sup>	20			years
$N_{RW}$	Write erase cycles	$T_A = +25^\circ\text{C}$	10			kcycles
$I_{DD}$	Supply current	Read / Write / Erase modes $f_{CPU} = 8\text{MHz}$ , $V_{DD} = 5.5\text{V}$			2.6 <sup>3)</sup>	mA
		No Read/No Write Mode			100	$\mu\text{A}$
		Power down mode / HALT		0	0.1 <sup>5)</sup>	$\mu\text{A}$

#### Notes:

1. Minimum  $V_{DD}$  supply voltage without losing data stored in RAM (in HALT mode or under RESET) or in hardware registers (only in HALT mode). Guaranteed by construction, not tested in production.
2. Up to 32 bytes can be programmed at a time.
3. The data retention time increases when the  $T_A$  decreases.
4. Data based on reliability test results and monitored in production.
5. Data based on characterization results, not tested in production.

**13.7 EMC CHARACTERISTICS**

Susceptibility tests are performed on a sample basis during product characterization.

**13.7.1 Functional EMS**

(Electro Magnetic Susceptibility)

Based on a simple running application on the product (toggling 2 LEDs through I/O ports), the product is stressed by two electro magnetic events until a failure occurs (indicated by the LEDs).

The level classification is described in application note AN1637.

- **ESD:** Electro-Static Discharge (positive and negative) is applied on all pins of the device until a functional disturbance occurs. This test conforms with the IEC 1000-4-2 standard.
- **FTB:** A Burst of Fast Transient voltage (positive and negative) is applied to  $V_{DD}$  and  $V_{SS}$  through a 100pF capacitor, until a functional disturbance occurs. This test conforms with the IEC 1000-4-4 standard.

A device reset allows normal operations to be resumed.

Symbol	Parameter	Conditions	Neg <sup>1)</sup>	Pos <sup>1)</sup>	Unit
$V_{FESD}$	Voltage limits to be applied on any I/O pin to induce a functional disturbance	$V_{DD}=5V, T_A=+25^{\circ}C, f_{OSC}=8MHz$ conforms to IEC 1000-4-2	TBD	TBD	kV
$V_{FFTB}$	Fast transient voltage burst limits to be applied through 100pF on $V_{DD}$ and $V_{DD}$ pins to induce a functional disturbance	$V_{DD}=5V, T_A=+25^{\circ}C, f_{OSC}=8MHz$ conforms to IEC 1000-4-4	TBD	TBD	

**13.7.2 Electro Magnetic Interference (EMI)**

Based on a simple application running on the product (toggling 2 LEDs through the I/O ports), the product is monitored in terms of emission. This emission test is in line with the norm SAE J 1752/3 which specifies the board and the loading of each pin.

Symbol	Parameter	Conditions	Monitored Frequency Band	Max vs. [ $f_{OSC}/f_{CPU}$ ]		Unit
				8/4MHz	16/8MHz	
$S_{EMI}$	Peak level	$V_{DD}=5V, T_A=+25^{\circ}C,$ conforming to SAE J 1752/3	0.1MHz to 30MHz	10	13	dB $\mu$ V
			30MHz to 130MHz	13	24	
			130MHz to 1GHz	16	31	
			SAE EMI Level	2.5	4	-

**Notes:**

1. Data based on characterization results, not tested in production.



**EMC CHARACTERISTICS (Cont'd)****13.7.3 Absolute Electrical Sensitivity**

Based on three different tests (ESD, LU and DLU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity. For more details, refer to the AN1181 ST7 application note.

**13.7.3.1 Electro-Static Discharge (ESD)**

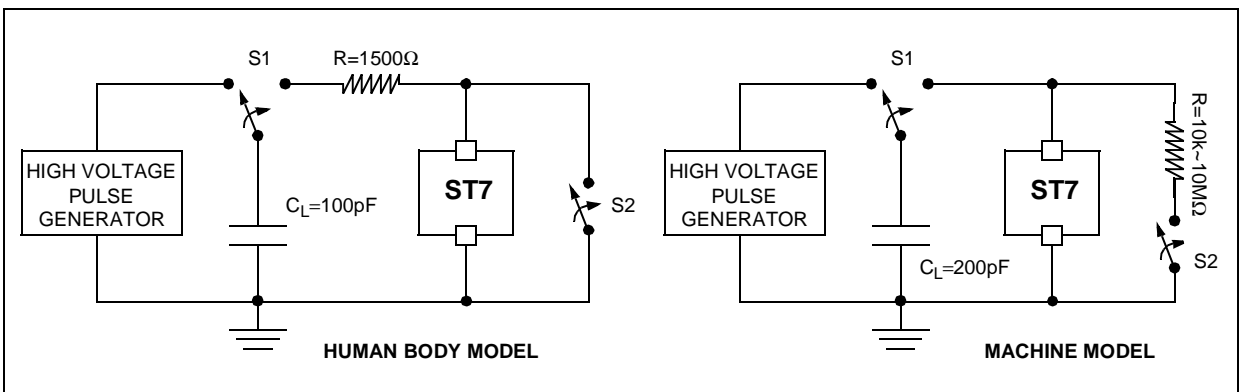
Electro-Static Discharges (a positive then a negative pulse separated by 1 second) are applied to the pins of each sample according to each pin combination. The sample size depends of the number of supply pins of the device (3 parts\*(n+1) supply pin). Two models are usually simulated: Human Body Model and Machine Model. This test conforms to the JESD22-A114A/A115A standard. See Figure 75 and the following test sequences.

**Human Body Model Test Sequence**

- $C_L$  is loaded through S1 by the HV pulse generator.
- S1 switches position from generator to R.
- A discharge from  $C_L$  through R (body resistance) to the ST7 occurs.
- S2 must be closed 10 to 100ms after the pulse delivery period to ensure the ST7 is not left in charge state. S2 must be opened at least 10ms prior to the delivery of the next pulse.

**Absolute Maximum Ratings**

Symbol	Ratings	Conditions	Maximum value <sup>1)</sup>	Unit
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human Body Model)	$T_A=+25^\circ\text{C}$	2000	V
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine Model)	$T_A=+25^\circ\text{C}$	200	

**Figure 75. Typical Equivalent ESD Circuits****Notes:**

1. Data based on characterization results, not tested in production.

**Machine Model Test Sequence**

- $C_L$  is loaded through S1 by the HV pulse generator.
- S1 switches position from generator to ST7.
- A discharge from  $C_L$  to the ST7 occurs.
- S2 must be closed 10 to 100ms after the pulse delivery period to ensure the ST7 is not left in charge state. S2 must be opened at least 10ms prior to the delivery of the next pulse.
- R (machine resistance), in series with S2, ensures a slow discharge of the ST7.

**EMC CHARACTERISTICS (Cont'd)**

**13.7.3.2 Static and Dynamic Latch-Up**

- **LU:** 3 complementary static tests are required on 10 parts to assess the latch-up performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latch-up standard. For more details, refer to the AN1181 ST7 application note.
- **DLU:** Electro-Static Discharges (one positive then one negative test) are applied to each pin of 3 samples when the micro is running to assess the latch-up performance in dynamic mode. Power supplies are set to the typical values, the oscillator is connected as near as possible to the pins of the micro and the component is put in reset mode. This test conforms to the IEC1000-4-2 and SAEJ1752/3 standards and is described in Figure 76. For more details, refer to the AN1181 ST7 application note.

**13.7.3.3 Designing hardened software to avoid noise problems**

EMC characterization and optimization are performed at component level with a typical application environment and simplified MCU software. It

should be noted that good EMC performance is highly dependent on the user application and the software in particular.

Therefore it is recommended that the user applies EMC software optimization and prequalification tests in relation with the EMC level requested for his application.

**Software recommendations:**

The software flowchart must include the management of runaway conditions such as:

- Corrupted program counter
- Unexpected reset
- Critical Data corruption (control registers...)

**Prequalification trials:**

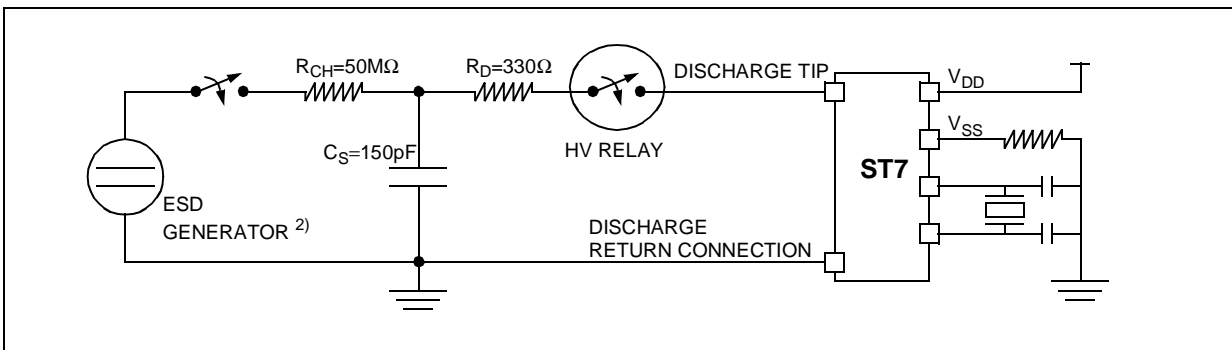
Most of the common failures (unexpected reset and program counter corruption) can be reproduced by manually forcing a low state on the RESET pin or the Oscillator pins for 1 second.

To complete these trials, ESD stress can be applied directly on the device, over the range of specification values. When unexpected behaviour is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

**Electrical Sensitivities**

Symbol	Parameter	Conditions	Class <sup>1)</sup>
LU	Static latch-up class	T <sub>A</sub> =+25°C	A
		T <sub>A</sub> =+85°C	A
DLU	Dynamic latch-up class	V <sub>DD</sub> =5.5V, f <sub>OSC</sub> =4MHz, T <sub>A</sub> =+25°C	A

**Figure 76. Simplified Diagram of the ESD Generator for DLU**



**Notes:**

1. Class description: A Class is an STMicroelectronics internal specification. All its limits are higher than the JEDEC specifications, that means when a device belongs to Class A it exceeds the JEDEC standard. B Class strictly covers all the JEDEC criteria (international standard).

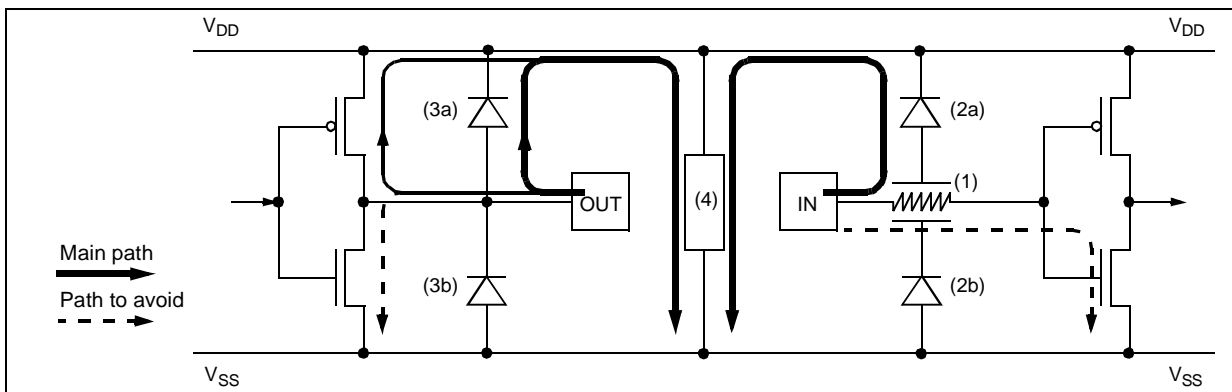
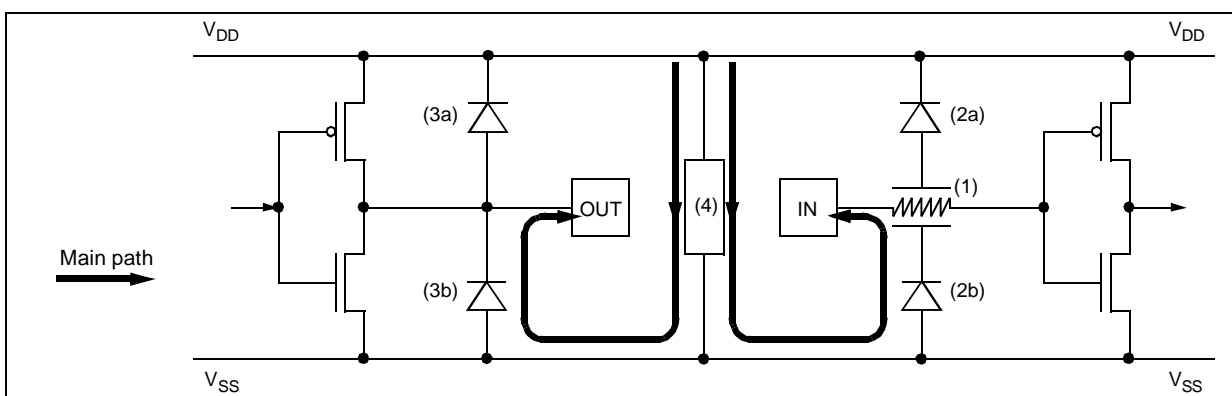
2. Schaffner NSG435 with a pointed test finger.

## EMC CHARACTERISTICS (Cont'd)

## 13.7.4 ESD Pin Protection Strategy

To protect an integrated circuit against Electro-Static Discharge the stress must be controlled to prevent degradation or destruction of the circuit elements. The stress generally affects the circuit elements which are connected to the pads but can also affect the internal devices when the supply pads receive the stress. The elements to be protected must not receive excessive current, voltage or heating within their structure.

An ESD network combines the different input and output ESD protections. This network works, by allowing safe discharge paths for the pins subjected to ESD stress. Two critical ESD stress cases are presented in Figure 77 and Figure 78 for standard pins and in Figure 79 and Figure 80 for true open drain pins.

Figure 77. Positive Stress on a Standard Pad vs.  $V_{SS}$ Figure 78. Negative Stress on a Standard Pad vs.  $V_{DD}$ 

## Standard Pin Protection

To protect the output structure the following elements are added:

- A diode to  $V_{DD}$  (3a) and a diode from  $V_{SS}$  (3b)
- A protection device between  $V_{DD}$  and  $V_{SS}$  (4)

To protect the input structure the following elements are added:

- A resistor in series with the pad (1)
- A diode to  $V_{DD}$  (2a) and a diode from  $V_{SS}$  (2b)
- A protection device between  $V_{DD}$  and  $V_{SS}$  (4)

**EMC CHARACTERISTICS (Cont'd)**

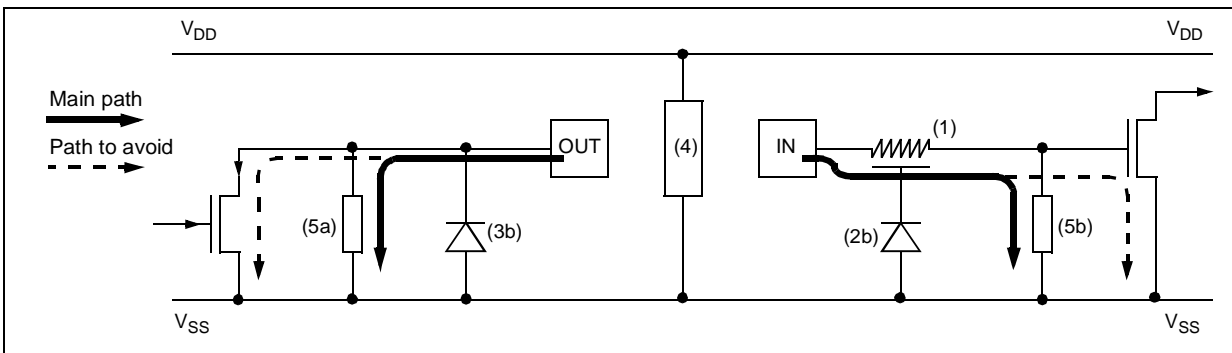
**True Open Drain Pin Protection**

The centralized protection (4) is not involved in the discharge of the ESD stresses applied to true open drain pads due to the fact that a P-Buffer and diode to  $V_{DD}$  are not implemented. An additional local protection between the pad and  $V_{SS}$  (5a & 5b) is implemented to completely absorb the positive ESD discharge.

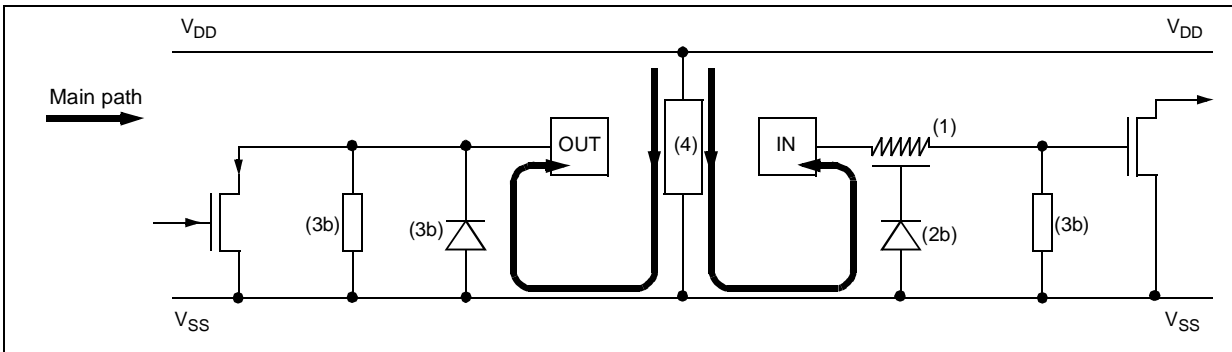
**Multisupply Configuration**

When several types of ground ( $V_{SS}$ ,  $V_{SSA}$ , ...) and power supply ( $V_{DD}$ ,  $V_{AREF}$ , ...) are available for any reason (better noise immunity...), the structure shown in Figure 81 is implemented to protect the device against ESD.

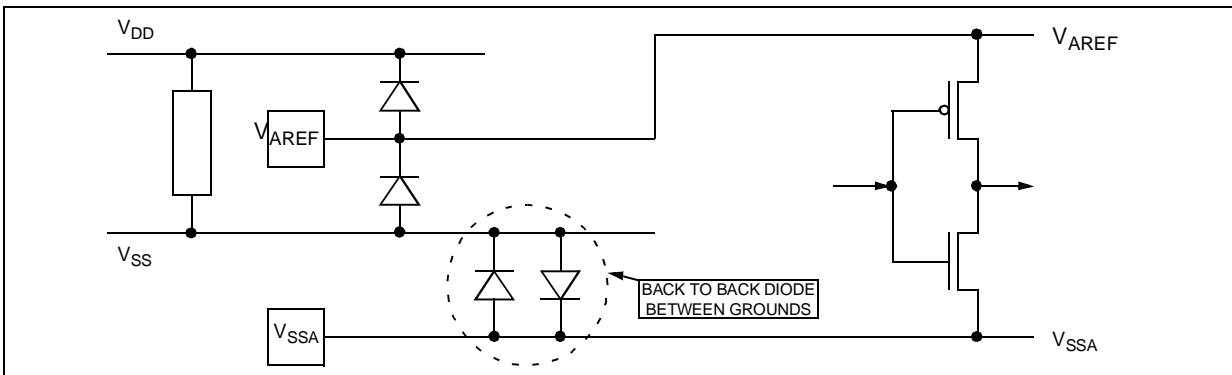
**Figure 79. Positive Stress on a True Open Drain Pad vs.  $V_{SS}$**



**Figure 80. Negative Stress on a True Open Drain Pad vs.  $V_{DD}$**



**Figure 81. Multisupply Configuration**



## 13.8 I/O PORT PIN CHARACTERISTICS

### 13.8.1 General Characteristics

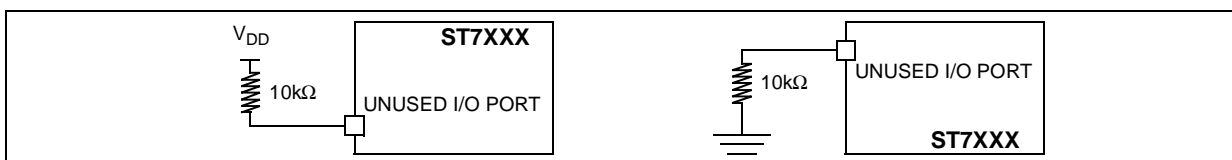
$T_A = -40$  to  $+85^\circ\text{C}$  unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IL}$	Input low level voltage <sup>1)</sup>				$0.3 \times V_{DD}$	V
$V_{IH}$	Input high level voltage <sup>1)</sup>		$0.7 \times V_{DD}$			
$V_{hys}$	Schmitt trigger voltage hysteresis <sup>1)</sup>			400		mV
$I_{INJ(PIN)}^{2)}$	Injected Current on an I/O pin	$V_{DD}=5V$			$\pm 4$	mA
$\Sigma I_{INJ(PIN)}^{2)}$	Total injected current (sum of all I/O and control pins)				$\pm 25$	
$I_L$	Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$			$\pm 1$	$\mu\text{A}$
$I_S$	Static current consumption	Floating input mode <sup>3)</sup>			200	
$R_{PU}$	Weak pull-up equivalent resistor <sup>4)</sup>	$V_{IN}=V_{SS}$	$V_{DD}=5V$ 50 $V_{DD}=3V$ 170 <sup>1)</sup>	85 190	250 230 <sup>1)</sup>	$\text{k}\Omega$
$C_{IO}$	I/O pin capacitance			5		
$t_{f(I/O)out}$	Output high to low level fall time <sup>1)</sup>	$C_L=50\text{pF}$ Between 10% and 90%		25		ns
$t_{r(I/O)out}$	Output low to high level rise time <sup>1)</sup>			25		
$t_{w(IT)in}$	External interrupt pulse time <sup>5)</sup>		1			$t_{CPU}$

#### Notes:

1. Data based on characterization results, not tested in production.
2. When the current limitation is not possible, the  $V_{IN}$  maximum must be respected, otherwise refer to  $I_{INJ(PIN)}$  specification. A positive injection is induced by  $V_{IN} > V_{DD}$  while a negative injection is induced by  $V_{IN} < V_{SS}$ . Refer to Section 13.2.2 on page 123 for more details.
3. Configuration not recommended, all unused pins must be kept at a fixed voltage: using the output mode of the I/O for example or an external pull-up or pull-down resistor (see Figure 82). Data based on design simulation and/or technology characteristics, not tested in production.
4. The  $R_{PU}$  pull-up equivalent resistor is based on a resistive transistor (corresponding  $I_{PU}$  current characteristics described in Figure 83).
5. To generate an external interrupt, a minimum pulse width has to be applied on an I/O port pin configured as an external interrupt source.

**Figure 82. Two typical Applications with unused I/O Pin**



I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 83. Typical  $I_{PU}$  vs.  $V_{DD}$  with  $V_{IN}=V_{SS}$

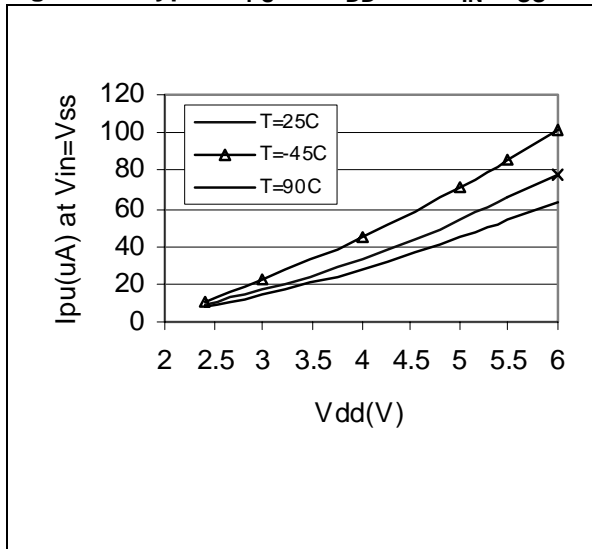


Figure 84. Typical  $V_{IL}$

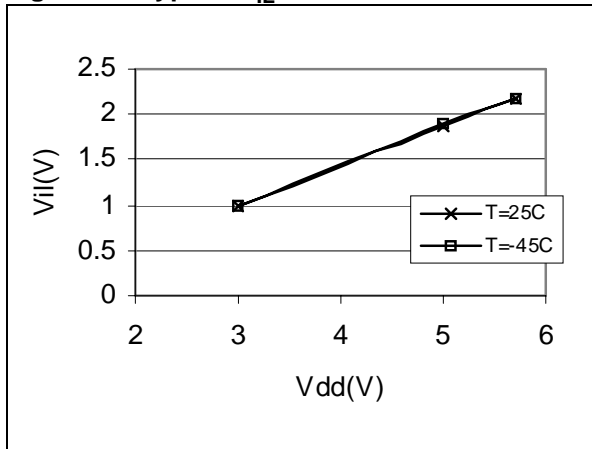
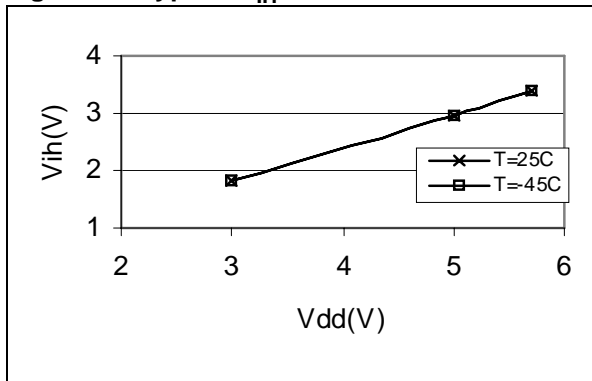


Figure 85. Typical  $V_{IH}$



**I/O PORT PIN CHARACTERISTICS (Cont'd)****13.8.2 Output Driving Current**

$T_A = -40$  to  $+85^\circ\text{C}$  unless otherwise specified

Symbol	Parameter	Conditions	Min	Max	Unit
$V_{OL}^{1)}$	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time	$I_{IO}=+5\text{mA}$		1.2	V
		$I_{IO}=+2\text{mA}$		0.5	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time	$I_{IO}=+20\text{mA}$ ,		1.3	
		$I_{IO}=+8\text{mA}$		0.75	
$V_{OH}^{2)}$	Output high level voltage for an I/O pin when 4 pins are sourced at same time	$I_{IO}=-5\text{mA}$ ,	$V_{DD}-1.6$		
		$I_{IO}=-2\text{mA}$	$V_{DD}-0.8$		
$V_{OL}^{1)3)}$	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time	$V_{DD}=5\text{V}$	$I_{IO}=+2\text{mA}$	0.6	
				$I_{IO}=+8\text{mA}$	
Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time	$V_{DD}=3.3\text{V}$		$I_{IO}=+2\text{mA}$	0.6	
				$I_{IO}=+8\text{mA}$	
$V_{OH}^{2)3)}$	Output high level voltage for an I/O pin when 4 pins are sourced at same time		$I_{IO}=-2\text{mA}$	$T_A \leq 85^\circ\text{C}$	$V_{DD}-0.8$
					$V_{DD}-0.8$
$V_{OL}^{1)3)}$	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time	$V_{DD}=2.7\text{V}$	$I_{IO}=+2\text{mA}$	0.7	
				$I_{IO}=+8\text{mA}$	0.7
Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time	$I_{IO}=+2\text{mA}$		0.7		
			$I_{IO}=+8\text{mA}$	0.7	
$V_{OH}^{2)3)}$	Output high level voltage for an I/O pin when 4 pins are sourced at same time		$I_{IO}=-2\text{mA}$	$V_{DD}-0.9$	
				$V_{DD}-0.9$	

**Notes:**

1. The  $I_{IO}$  current sunk must always respect the absolute maximum rating specified in Section 13.2.2 and the sum of  $I_{IO}$  (I/O ports and control pins) must not exceed  $I_{VSS}$ .
2. The  $I_{IO}$  current sourced must always respect the absolute maximum rating specified in Section 13.2.2 and the sum of  $I_{IO}$  (I/O ports and control pins) must not exceed  $I_{VDD}$ . True open drain I/O pins does not have  $V_{OH}$ .
3. Not tested in production, based on characterization results.

I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 86. Typ.  $V_{OL}$  at  $V_{DD}=2.4V$  (standard)

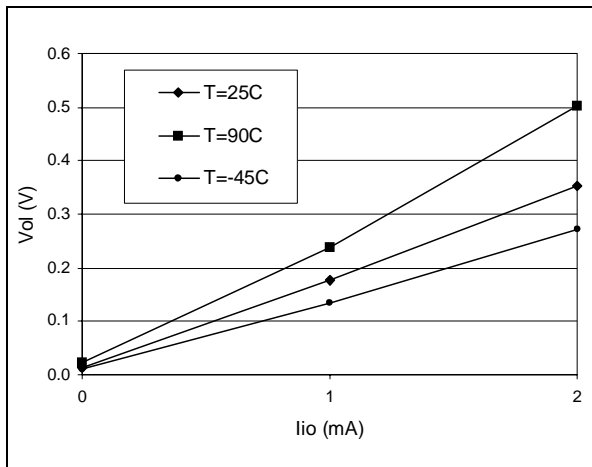


Figure 89. Typ.  $V_{OL}$  at  $V_{DD}=2.7V$  (standard)

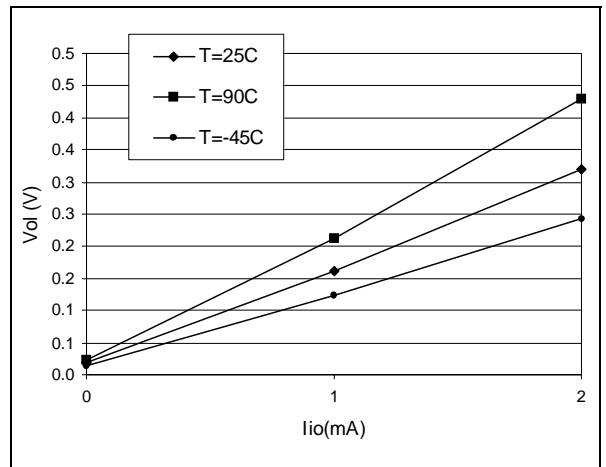


Figure 87. Typ.  $V_{OL}$  at  $V_{DD}=5V$  (standard)

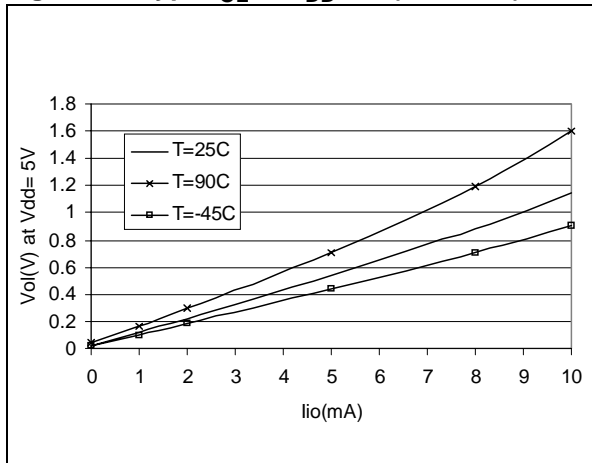


Figure 90. Typ.  $V_{OL}$  at  $V_{DD}=2.4V$  (high-sink)

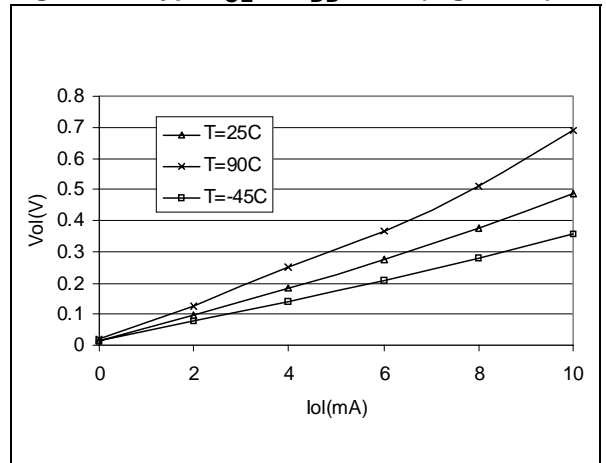


Figure 88. Typ.  $V_{OL}$  at  $V_{DD}=3V$  (high-sink)

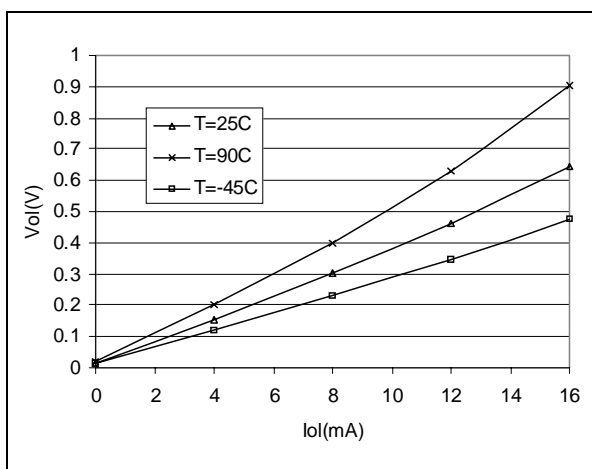
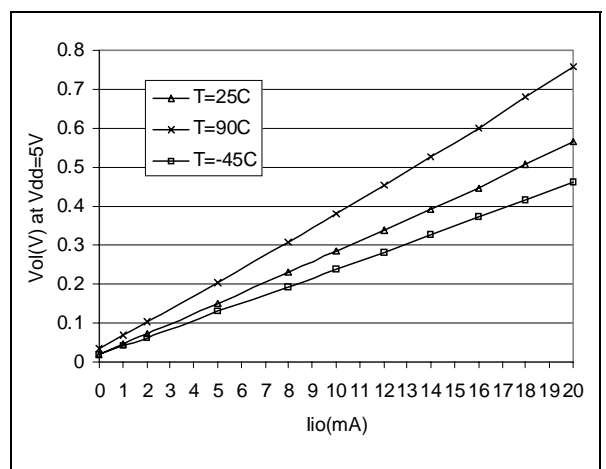


Figure 91. Typ.  $V_{OL}$  at  $V_{DD}=5V$  (high-sink)





I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 92. Typ.  $V_{OH}$  at  $V_{DD}=2.7V$

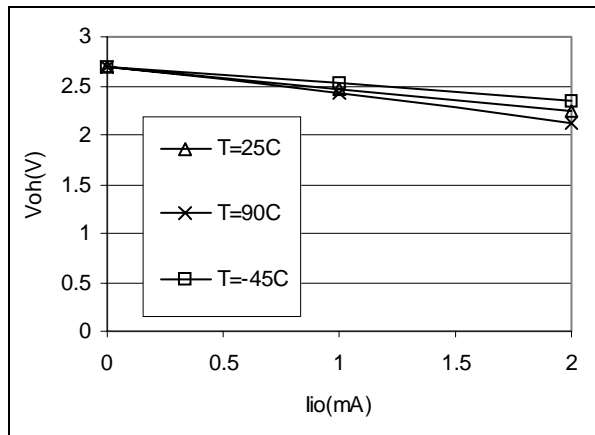


Figure 95. Typ.  $V_{OH}$  at  $V_{DD}=3V$

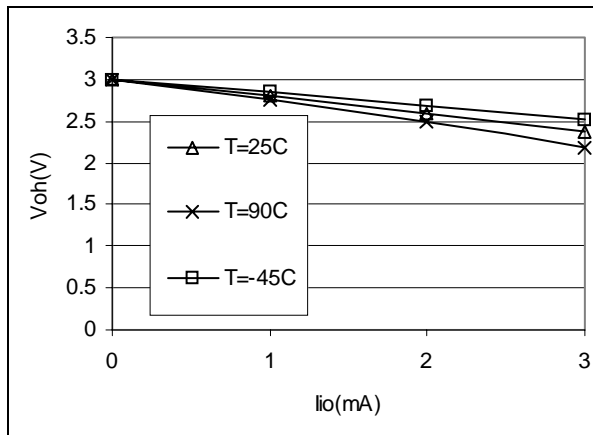


Figure 93. Typ.  $V_{OH}$  at  $V_{DD}=4V$

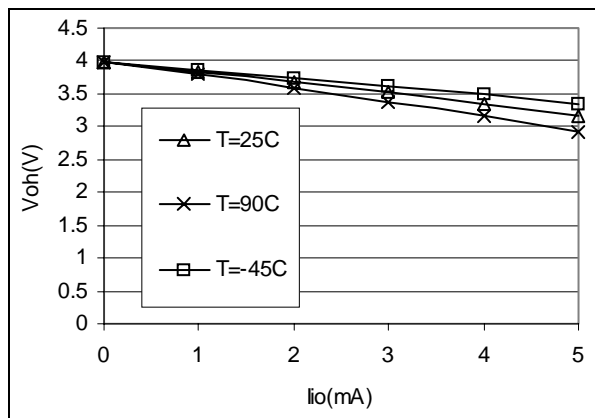


Figure 96. Typ.  $V_{OH}$  at  $V_{DD}=5V$

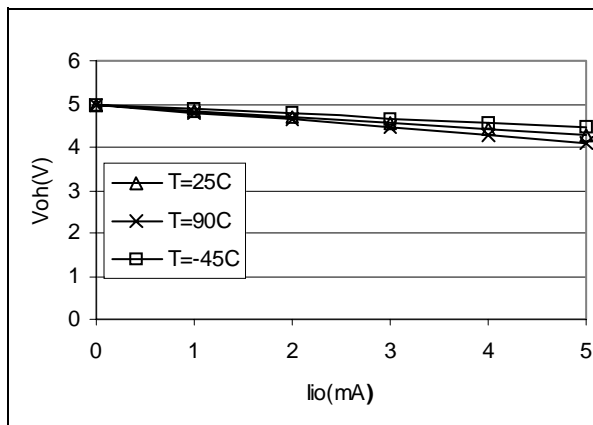
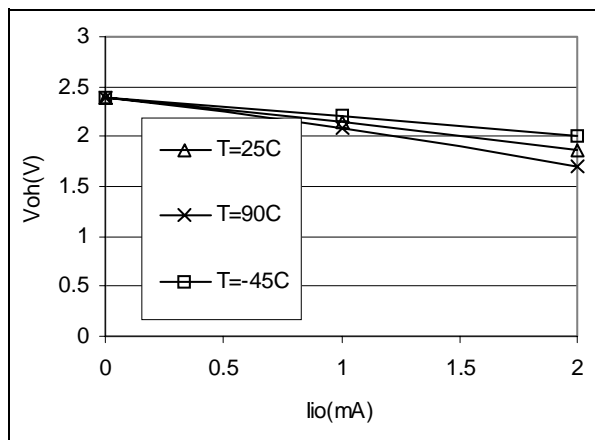


Figure 94. Typ.  $V_{OH}$  at  $V_{DD}=2.4V$



Notes:

1. The  $I_{IO}$  current sunk must always respect the absolute maximum rating specified in Section 13.2.2 and the sum of  $I_{IO}$  (I/O ports and control pins) must not exceed  $I_{VSS}$ .
2. The  $I_{IO}$  current sourced must always respect the absolute maximum rating specified in Section 13.2.2 and the sum of  $I_{IO}$  (I/O ports and control pins) must not exceed  $I_{VDD}$ . True open drain I/O pins does not have  $V_{OH}$ .

I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 97. Typical  $V_{OL}$  vs.  $V_{DD}$  on standard I/O port (Ports B and C)

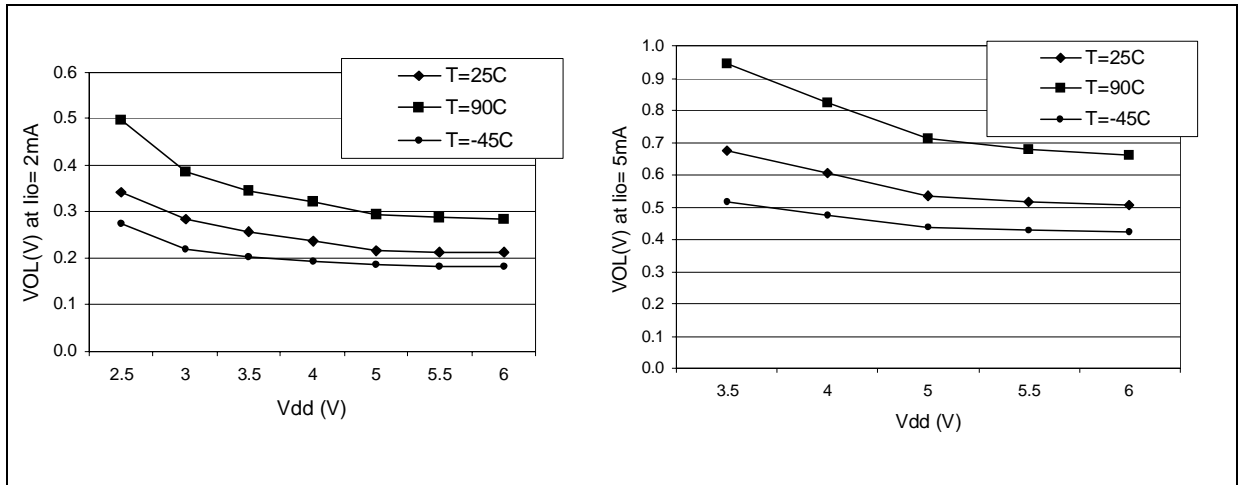
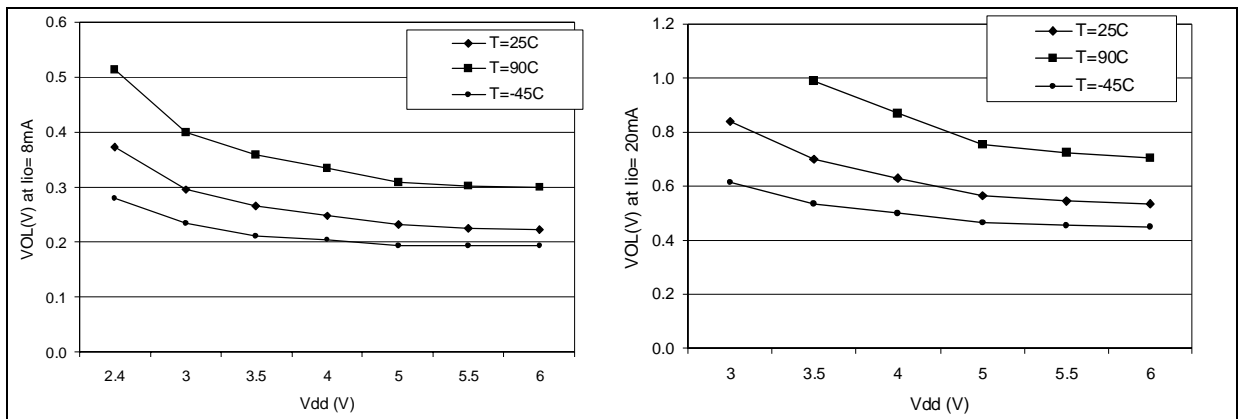


Figure 98. Typical  $V_{OL}$  vs.  $V_{DD}$  on high sink I/O port (Port A)



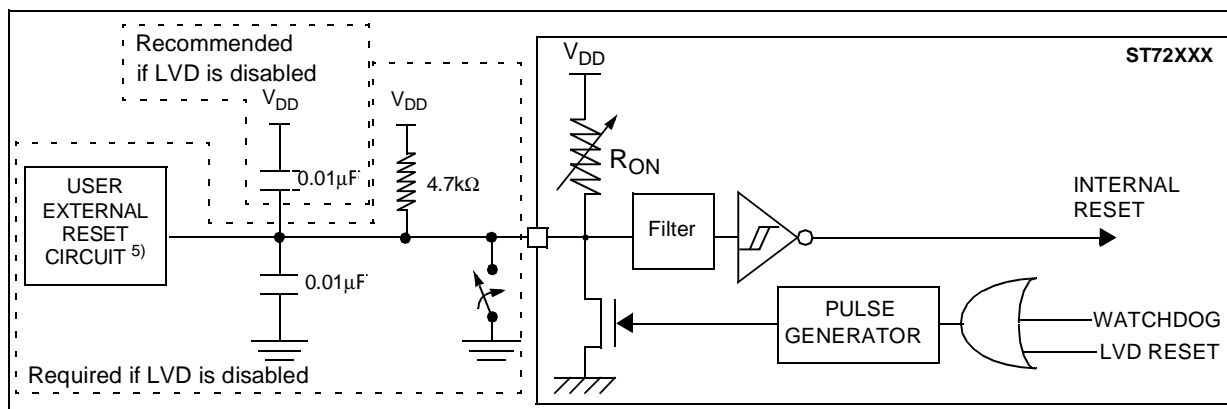
## 13.9 CONTROL PIN CHARACTERISTICS

### 13.9.1 Asynchronous $\overline{\text{RESET}}$ Pin

$T_A = -40$  to  $+85^\circ\text{C}$  unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IL}$	Input low level voltage				$0.16 \times V_{DD}$	V
$V_{IH}$	Input high level voltage		$0.85 \times V_{DD}$			
$V_{hys}$	Schmitt trigger voltage hysteresis <sup>1)</sup>			2.5		V
$V_{OL}$	Output low level voltage <sup>2)</sup>	$V_{DD}=5\text{V}$ $I_{IO}=+5\text{mA}$		0.68	0.95	V
			$I_{IO}=+2\text{mA}$	0.28	0.45	
$R_{ON}$	Pull-up equivalent resistor	$V_{DD}=5\text{V}$	20	40	80	k $\Omega$
		$V_{DD}=3\text{V}$		85		
$t_{w(\text{RSTL})\text{out}}$	Generated reset pulse duration	Internal reset sources		30		$\mu\text{s}$
$t_{h(\text{RSTL})\text{in}}$	External reset pulse hold time <sup>4)</sup>		20			$\mu\text{s}$
$t_{g(\text{RSTL})\text{in}}$	Filtered glitch duration <sup>5)</sup>			200		ns

Figure 99. Typical Application with  $\overline{\text{RESET}}$  pin<sup>6)7)8)</sup>

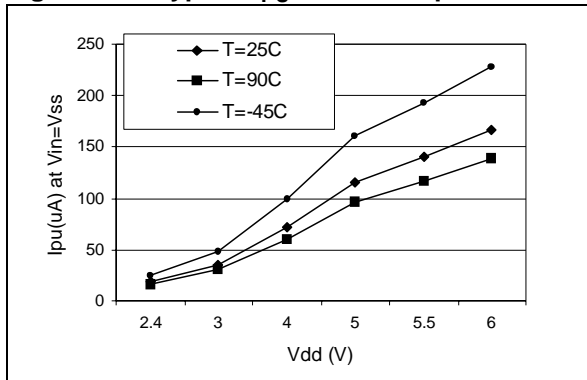


#### Notes:

1. Data based on characterization results, not tested in production.
2. The  $I_{IO}$  current sunk must always respect the absolute maximum rating specified in Section 13.2.2 and the sum of  $I_{IO}$  (I/O ports and control pins) must not exceed  $I_{VSS}$ .
3. To guarantee the reset of the device, a minimum pulse has to be applied to the  $\overline{\text{RESET}}$  pin. All short pulses applied on  $\overline{\text{RESET}}$  pin with a duration below  $t_{h(\text{RSTL})\text{in}}$  can be ignored.
4. The reset network protects the device against parasitic resets.
5. The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).
6. Whatever the reset source is (internal or external), the user must ensure that the level on the  $\overline{\text{RESET}}$  pin can go below the  $V_{IL}$  max. level specified in Section 13.9.1. Otherwise the reset will not be taken into account internally.
7. Because the reset circuit is designed to allow the internal  $\overline{\text{RESET}}$  to be output in the  $\overline{\text{RESET}}$  pin, the user must ensure that the current sunk on the  $\overline{\text{RESET}}$  pin (by an external pull-up for example) is less than the absolute maximum value specified for  $I_{\text{INJ}(\overline{\text{RESET}})}$  in Section 13.2.2 on page 123.

**CONTROL PIN CHARACTERISTICS (Cont'd)**

**Figure 100. Typical  $I_{PU}$  on RESET pin**



**13.10 TIMER PERIPHERAL CHARACTERISTICS**

Subject to general operating conditions for  $V_{DD}$ ,  $f_{OSC}$ , and  $T_A$  unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (output compare, input capture, external clock, PWM output...).

**13.10.1 16-Bit Timer**

$T_A = -40$  to  $+85^\circ\text{C}$  unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{w(ICAP)in}$	Input capture pulse time		1			$t_{CPU}$
$t_{res(PWM)}$	PWM resolution time		2			$t_{CPU}$
		$f_{CPU}=8\text{MHz}$	250			ns
$f_{EXT}$	Timer external clock frequency		0		$f_{CPU}/4$	MHz
$f_{PWM}$	PWM repetition rate		0		$f_{CPU}/4$	MHz
$Res_{PWM}$	PWM resolution				16	bit

### 13.11 COMMUNICATION INTERFACE CHARACTERISTICS

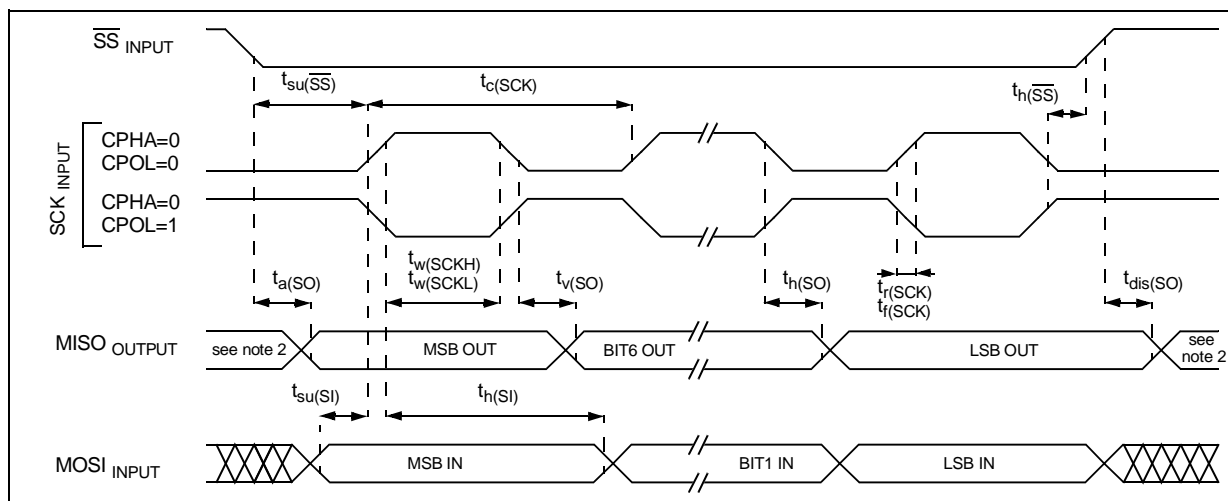
#### 13.11.1 SPI - Serial Peripheral Interface

Subject to general operating conditions for  $V_{DD}$ ,  $f_{OSC}$ , and  $T_A$  unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (SS, SCK, MOSI, MISO).

Symbol	Parameter	Conditions	Min	Max	Unit
$f_{SCK}$ $1/t_{c(SCK)}$	SPI clock frequency	Master $f_{CPU}=8MHz$	$f_{CPU}/128$ 0.0625	$f_{CPU}/4$ 2	MHz
		Slave $f_{CPU}=8MHz$	0	$f_{CPU}/2$ 4	
$t_{r(SCK)}$ $t_{f(SCK)}$	SPI clock rise and fall time		see I/O port pin description		
$t_{su}(\overline{SS})$	SS setup time	Slave	120		ns
$t_{h}(\overline{SS})$	SS hold time	Slave	120		
$t_w(SCKH)$ $t_w(SCKL)$	SCK high and low time	Master	100		
		Slave	90		
$t_{su}(MI)$ $t_{su}(SI)$	Data input setup time	Master	100		
		Slave	100		
$t_{h}(MI)$ $t_{h}(SI)$	Data input hold time	Master	100		
		Slave	100		
$t_a(SO)$	Data output access time	Slave	0	120	
$t_{dis}(SO)$	Data output disable time	Slave		240	
$t_v(SO)$	Data output valid time	Slave (after enable edge)		120	
$t_h(SO)$	Data output hold time		0		
$t_v(MO)$	Data output valid time	Master (before capture edge)	0.25		$t_{CPU}$
$t_h(MO)$	Data output hold time		0.25		

Figure 101. SPI Slave Timing Diagram with CPHA=0<sup>3)</sup>



**Notes:**

1. Data based on design simulation and/or characterisation results, not tested in production.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration.
3. Measurement points are done at CMOS levels:  $0.3 \times V_{DD}$  and  $0.7 \times V_{DD}$ .

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

Figure 102. SPI Slave Timing Diagram with CPHA=1<sup>1)</sup>

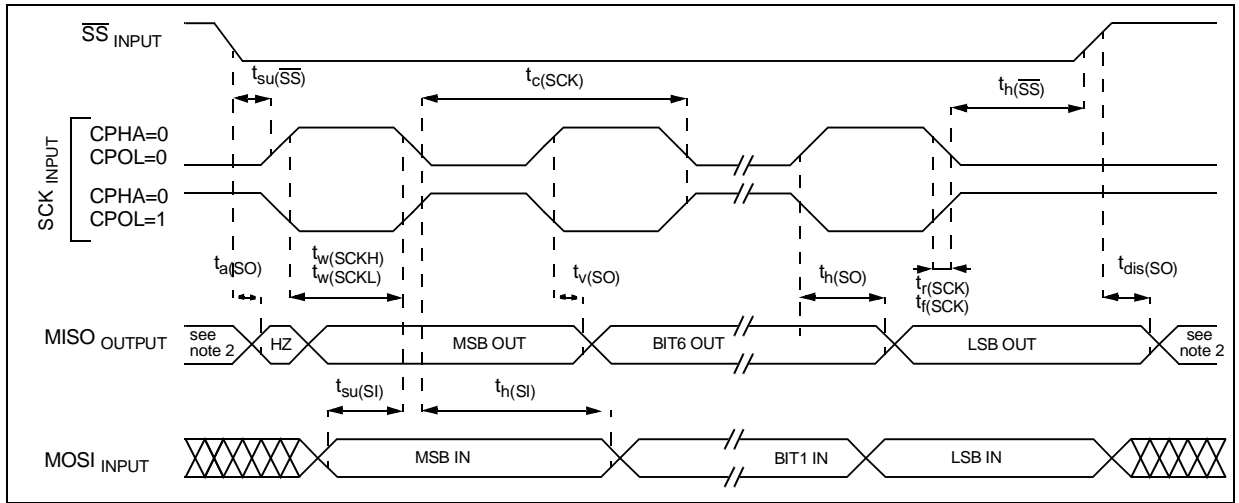
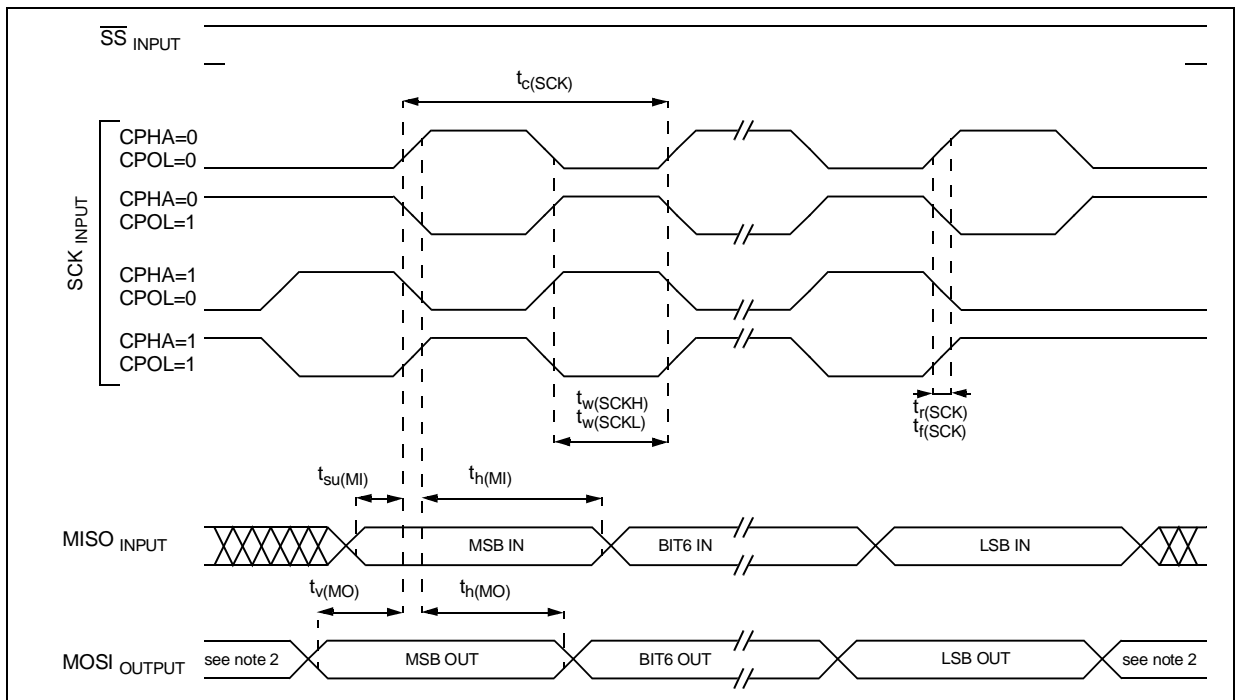


Figure 103. SPI Master Timing Diagram<sup>1)</sup>



Notes:

1. Measurement points are done at CMOS levels:  $0.3 \times V_{DD}$  and  $0.7 \times V_{DD}$ .
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

## COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

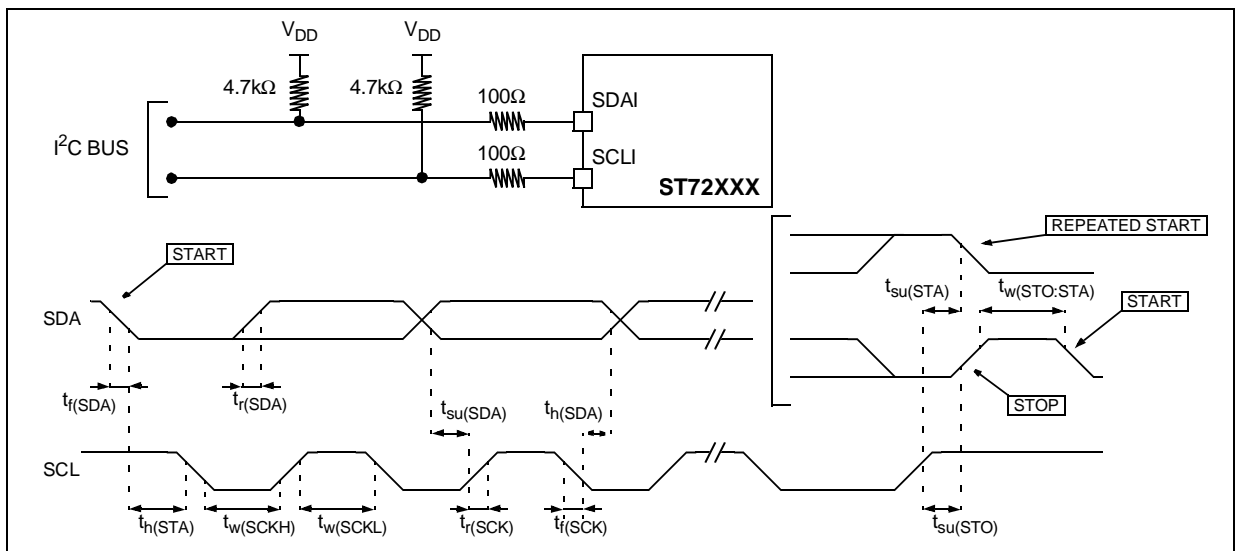
13.11.2 I<sup>2</sup>C - Inter IC Control Interface

Subject to general operating conditions for  $V_{DD}$ ,  $f_{OSC}$ , and  $T_A$  unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (SDAI and SCLI). The ST7 I<sup>2</sup>C interface meets the requirements of the Standard I<sup>2</sup>C communication protocol described in the following table.

Symbol	Parameter	Standard mode I <sup>2</sup> C		Fast mode I <sup>2</sup> C		Unit
		Min <sup>1)</sup>	Max <sup>1)</sup>	Min <sup>1)</sup>	Max <sup>1)</sup>	
$t_{w(SCLL)}$	SCL clock low time	4.7		1.3		$\mu s$
$t_{w(SCLH)}$	SCL clock high time	4.0		0.6		
$t_{su(SDA)}$	SDA setup time	250		100		ns
$t_{h(SDA)}$	SDA data hold time	0 <sup>3)</sup>		0 <sup>2)</sup>	900 <sup>3)</sup>	
$t_{r(SDA)}$ $t_{r(SCL)}$	SDA and SCL rise time		1000	$20+0.1C_b$	300	
$t_{f(SDA)}$ $t_{f(SCL)}$	SDA and SCL fall time		300	$20+0.1C_b$	300	
$t_{h(STA)}$	START condition hold time	4.0		0.6		$\mu s$
$t_{su(STA)}$	Repeated START condition setup time	4.7		0.6		
$t_{su(STO)}$	STOP condition setup time	4.0		0.6		ns
$t_{w(STO:STA)}$	STOP to START condition time (bus free)	4.7		1.3		ms
$C_b$	Capacitive load for each bus line		400		400	pF

Figure 104. Typical Application with I<sup>2</sup>C Bus and Timing Diagram <sup>4)</sup>



## Notes:

1. Data based on standard I<sup>2</sup>C protocol requirement, not tested in production.
2. The device must internally provide a hold time of at least 300ns for the SDA signal in order to bridge the undefined region of the falling edge of SCL.
3. The maximum hold time of the START condition has only to be met if the interface does not stretch the low period of SCL signal.
4. Measurement points are done at CMOS levels:  $0.3 \times V_{DD}$  and  $0.7 \times V_{DD}$ .

13.12 10-BIT ADC CHARACTERISTICS

T<sub>A</sub> = -40°C to 85°C, unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f <sub>ADC</sub>	ADC clock frequency		0.4 <sup>1)</sup>		4	MHz
V <sub>AIN</sub>	Conversion voltage range		V <sub>SS</sub>		V <sub>DD</sub>	V
C <sub>ADC</sub>	Internal sample and hold capacitor			6		pF
t <sub>CONV</sub>	Conversion time	f <sub>ADC</sub> =4MHz		28		μs
				112		1/f <sub>ADC</sub>
R <sub>AIN</sub>	External input impedance				see Figure 105 and Figure 106 <sup>2)3)4)</sup>	kΩ
C <sub>AIN</sub>	External capacitor on analog input					pF
f <sub>AIN</sub>	Variation frequency of analog input signal					Hz

Figure 105. R<sub>AIN</sub> max. vs f<sub>ADC</sub> with C<sub>AIN</sub>=0pF<sup>3)</sup>

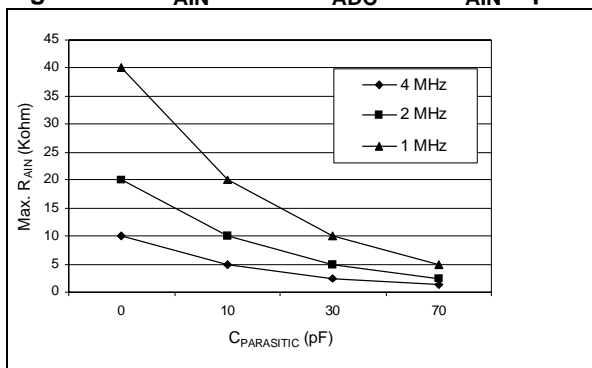


Figure 106. Recommended C<sub>AIN</sub>/R<sub>AIN</sub> values<sup>4)</sup>

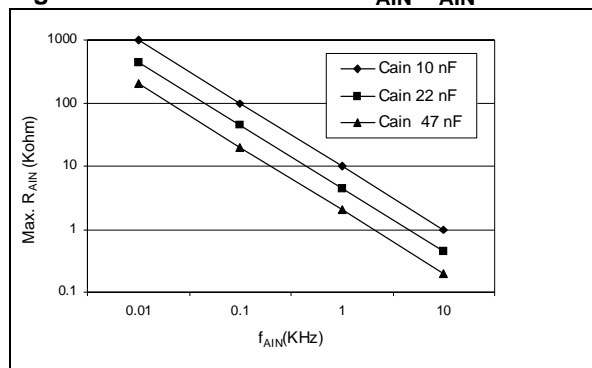
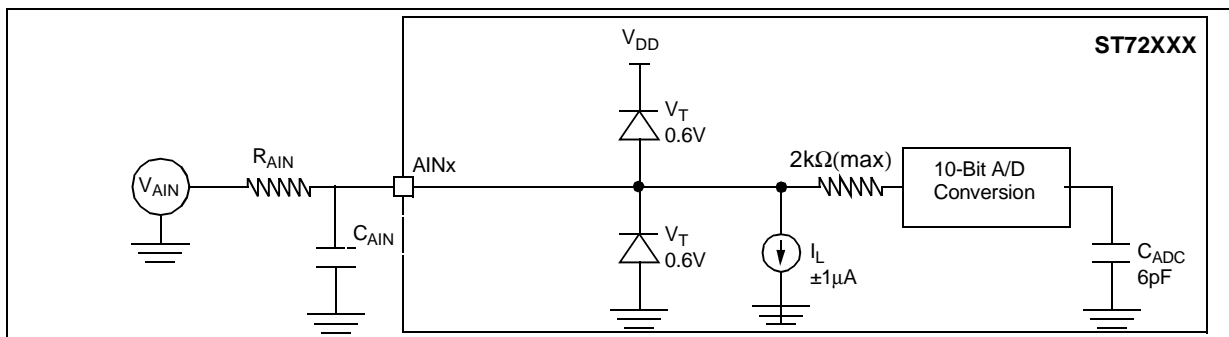


Figure 107. Analog Input equivalent circuit



Notes:

1. Data based on design simulation.
2. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than 10kΩ). Data based on characterization results, not tested in production.
3. C<sub>PARASITIC</sub> represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high C<sub>PARASITIC</sub> value will downgrade conversion accuracy. To remedy this, f<sub>ADC</sub> should be reduced.
4. This graph shows that depending on the input signal variation (f<sub>AIN</sub>), C<sub>AIN</sub> can be increased for stabilization time and decreased to allow the use of a larger serial resistor (R<sub>AIN</sub>). It is valid for all f<sub>ADC</sub> frequencies ≤ 4MHz.



## ADC CHARACTERISTICS (Cont'd)

## 13.12.0.1 General PCB Design Guidelines

To obtain best results, some general design and layout rules should be followed when designing the application PCB to shield the noise-sensitive, analog physical interface from noise-generating CMOS logic signals.

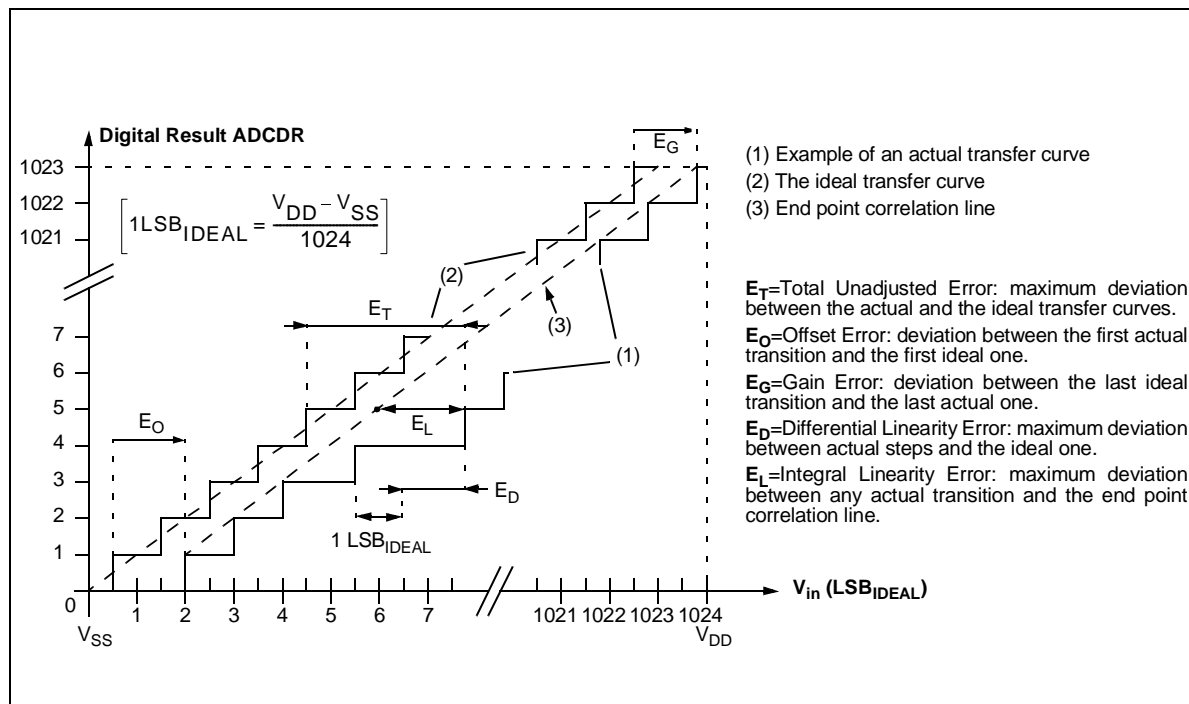
- Properly place components and route the signal traces on the PCB to shield the analog inputs.

Analog signals paths should run over the analog ground plane and be as short as possible. Isolate analog signals from digital signals that may switch while the analog inputs are being sampled by the A/D converter. Do not toggle digital outputs on the same I/O port as the A/D input being converted.

ADC Accuracy with  $f_{CPU}=8\text{ MHz}$ ,  $f_{ADC}=4\text{ MHz}$ ,  $R_{AIN}<10\text{ k}\Omega$ ,  $V_{DD}=4.5\text{V to }5.5\text{V}$

Symbol	Parameter	Conditions	Typ <sup>2)</sup>	Max	Unit
$ E_T $	Total unadjusted error <sup>1)</sup>		4		LSB
$E_O$	Offset error <sup>1)</sup>		1	-2.5/+2.5	
$E_G$	Gain Error <sup>1)</sup>		1	-1.5/+3	
$ E_D $	Differential linearity error <sup>1)</sup>		1.5	3	
$ E_L $	Integral linearity error <sup>1)</sup>		3	5	

Figure 108. ADC Accuracy Characteristics



## Notes:

1. ADC Accuracy vs. Negative Injection Current: Injecting negative current on any of the analog input pins significantly reduces the accuracy of the conversion being performed on another analog input.

For  $I_{INJ}=0.8\text{ mA}$ , the typical leakage induced inside the die is  $1.6\text{ }\mu\text{A}$  and the effect on the ADC accuracy is a loss of 4 LSB for each  $10\text{ k}\Omega$  increase of the external analog source impedance. It is recommended to add a Schottky diode (pin to ground) to analog pins which may potentially inject negative current. Any positive injection current within the limits specified for  $I_{INJ(PIN)}$  and  $\Sigma I_{INJ(PIN)}$  in Section 13.8 does not affect the ADC accuracy.

2. Refer to "Typical values" on page 122 for more information on typical ADC accuracy values.

## 14 PACKAGE CHARACTERISTICS

### 14.1 PACKAGE MECHANICAL DATA

Figure 109. 32-Pin Plastic Dual In-Line Package, Shrink 400-mil Width

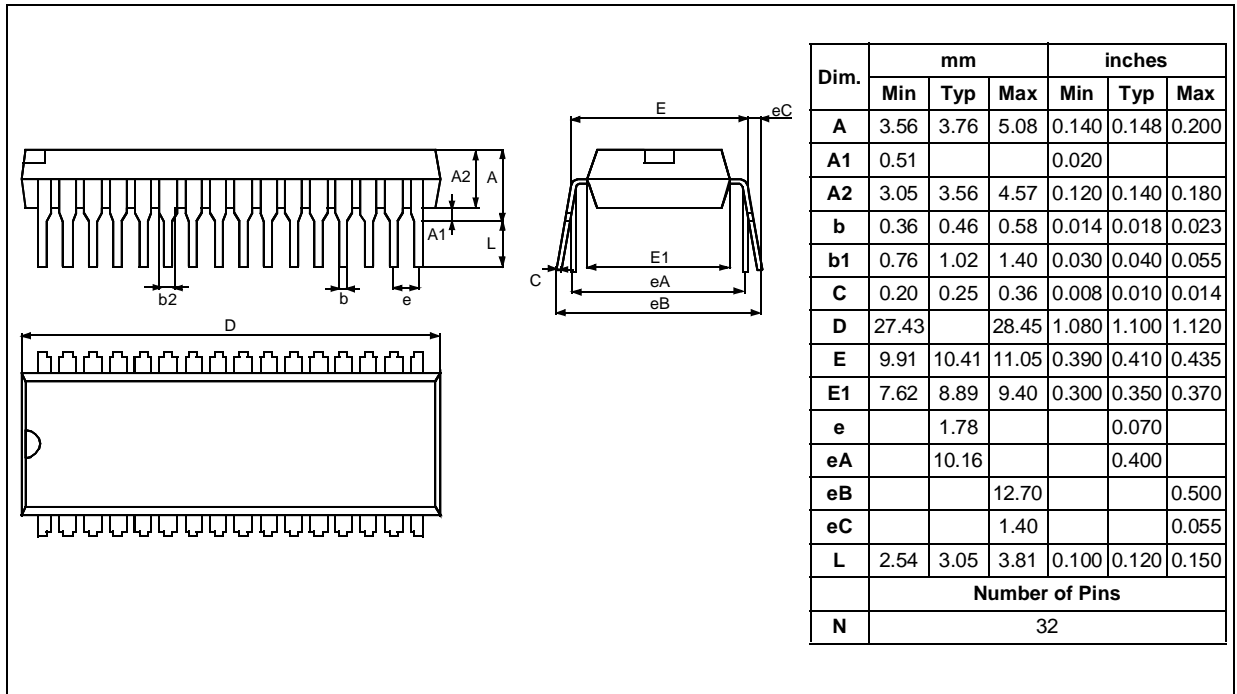


Figure 110. 28-Pin Plastic Small Outline Package, 300-mil Width

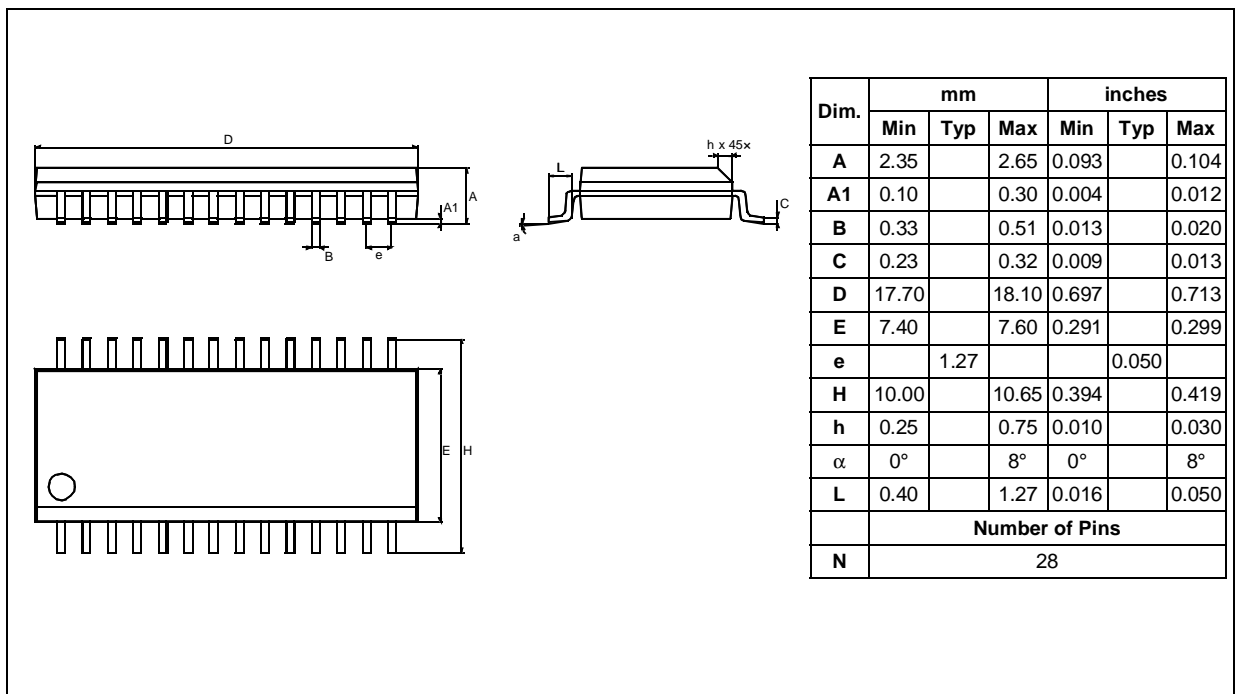
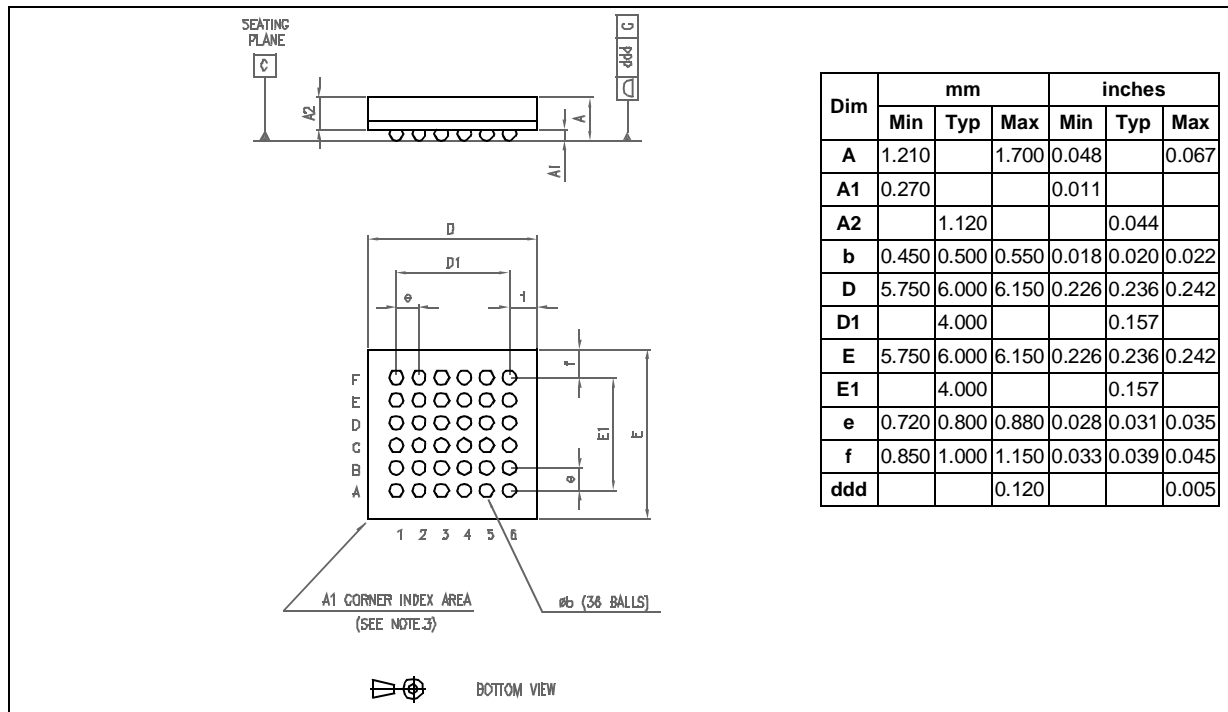


Figure 111. Low Profile Fine Pitch Ball Grid Array Package



14.2 THERMAL CHARACTERISTICS

Symbol	Ratings	Value	Unit
R <sub>thJA</sub>	Package thermal resistance (junction to ambient)		
	SDIP32	60	°C/W
	SO28	75	
	LFBGA 6x6 (on multilayer PCB)	56	
LFBGA 6x6 (on single-layer PCB)	72		
P <sub>D</sub>	Power dissipation <sup>1)</sup>	500	mW
T <sub>Jmax</sub>	Maximum junction temperature <sup>2)</sup>	150	°C

Notes:

- The power dissipation is obtained from the formula  $P_D = P_{INT} + P_{PORT}$  where  $P_{INT}$  is the chip internal power ( $I_{DD} \times V_{DD}$ ) and  $P_{PORT}$  is the port power dissipation determined by the user.
- The average chip-junction temperature can be obtained from the formula  $T_J = T_A + P_D \times R_{thJA}$ .

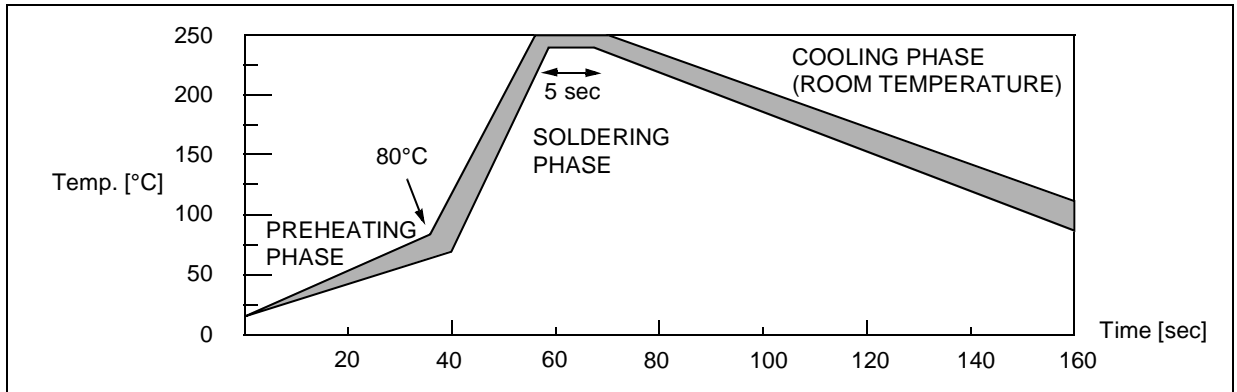
**14.3 SOLDERING AND GLUEABILITY INFORMATION**

Recommended soldering information given only as design guidelines in Figure 112 and Figure 113.

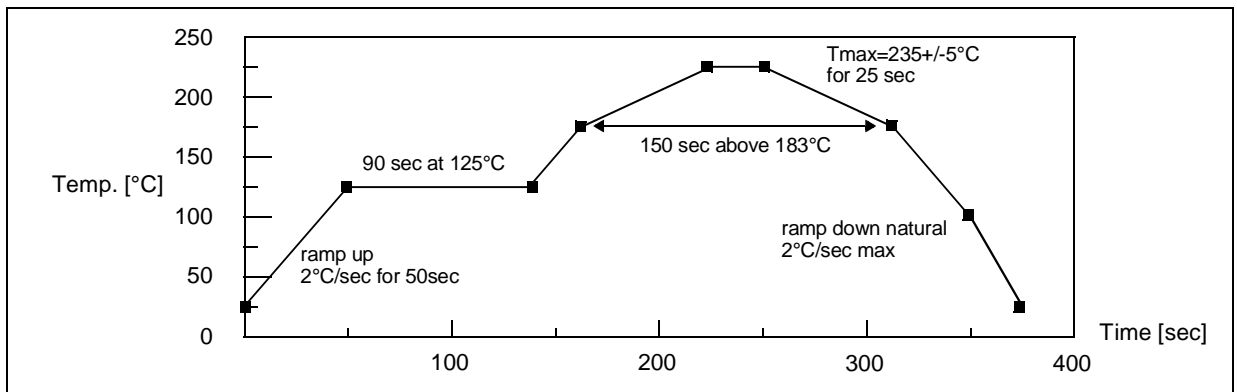
Recommended glue for SMD plastic packages dedicated to molding compound with silicone:

- Heraeus: PD945, PD955
- Loctite: 3615, 3298

**Figure 112. Recommended Wave Soldering Profile (with 37% Sn and 63% Pb)**



**Figure 113. Recommended Reflow Soldering Oven Profile (MID JEDEC)**



## 15 DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in user programmable versions (FLASH) as well as in factory coded versions (ROM/FASTROM).

ST7226x devices are ROM versions. ST72P26x devices are Factory Advanced Service Technique ROM (FASTROM) versions: they are factory-programmed XFlash devices.

ST72F26x XFlash devices are shipped to customers with a default program memory content (FFh). The option bytes are programmed to enable the internal RC oscillator. The ROM/FASTROM factory coded parts contain the code supplied by the customer. This implies that FLASH devices have to be configured by the customer using the Option Bytes while the ROM/FASTROM devices are factory-configured.

### 15.1 OPTION BYTES

The two option bytes allow the hardware configuration of the microcontroller to be selected.

The option bytes have no address in the memory map and can be accessed only in programming mode (for example using a standard ST7 programming tool). The default content of the FLASH is fixed to FFh.

In masked ROM devices, the option bytes are fixed in hardware by the ROM code (see option list).

#### USER OPTION BYTE 0

OPT 7 = **WDG HALT** *Watchdog reset on HALT*

This option bit determines if a RESET is generated when entering HALT mode while the Watchdog is active.

0: No Reset generation when entering Halt mode  
1: Reset generation when entering Halt mode

OPT 6 = **WDG SW** *Hardware or software watchdog*

This option bit selects the watchdog type.

	USER OPTION BYTE 0								USER OPTION BYTE 1							
	7							0	7							0
	WDG HALT	WDG SW	VD1	VD0	SEC 1	SEC 0	FMP R	FMP W	EXTIT	CSS	OSC TYPE 1	OSC TYPE 0	OSC RNGE 2	OSC RNGE 1	OSC RNGE 0	PLL OFF
Default Value	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	1

0: Hardware (watchdog always enabled)

1: Software (watchdog to be enabled by software)

OPT 5:4 = **VD[1:0]** *Voltage detection selection*

These option bits enable the voltage detection block (LVD and AVD) with a selected threshold of the LVD and AVD.

Configuration	VD1	VD0
LVD Off	1	1
Lowest Voltage Threshold (~-3.05V)	1	0
Medium Voltage Threshold (~-3.6V)	0	1
Highest Voltage Threshold (~-4.1V)	0	0

OPT 3:2 = **SEC[1:0]** *Sector 0 size definition*

These option bits indicate the size of sector 0 according to the following table.

Sector 0 Size	SEC1	SEC0
0.5k	0	0
1k	0	1
2	1	0
4k	1	1

OPT 1 = **FMP\_R** *Read-out protection*

This option indicates if the program memory is protected against piracy. See "Memory Protection" on page 16.. The read-out protection blocks access to the program memory in any mode except user mode and IAP mode. Erasing the option bytes when the FMP\_R option is selected will cause the whole memory to be erased first, and the device can be reprogrammed. Refer to Section 4.5 and the ST7 Flash Programming Reference Manual for more details.

0: Read-out protection off

1: Read-out protection on

**DEVICE CONFIGURATION (Cont'd)**

OPT 0 = **FMP\_W** FLASH write protection

This option indicates if the FLASH program memory is write protected.

**Warning:** When this option is selected, the program memory (and the option bit itself) can never be erased or programmed again.

0: Write protection off  
1: Write protection on

**USER OPTION BYTE 1**

OPT 7 = **EXTIT** Port C External Interrupt Configuration.

This option bit allows the Port C external interrupt mapping to be configured as ei0 or ei1.

**Table 26. External Interrupt Configuration**

ei0	ei1	EXTIT option bit
PA[7:0] Ports	PB[7:0] Ports PC[5:0] Ports	1
PA[7:0] Ports PC[5:0] Ports	PB[7:0] Ports	0

OPT 6 = **CSS** Clock Security System on/off

This option bit enables or disables the clock security system (CSS) which include the clock filter and the backup safe oscillator.

0: CSS enabled  
1: CSS disabled

**Caution:** The Clock Security System is not guaranteed. The features described in Section 6.4.3 on page 26 are subject to revision.

OPT 5:4 = **OSCTYPE[1:0]** Oscillator Type selection

These option bits select the Oscillator Type.

Clock Source	OSCTYPE1	OSCTYPE0
Resonator Oscillator	0	0
Reserved	0	1
Internal RC Oscillator	1	0
External Source	1	1

OPT 3:1 = **OSCRNGE[2:0]** Oscillator Range selection

These option bits select the oscillator range.

Typ. Freq. Range		OSC RNGE2	OSC RNGE1	OSC RNGE0
VLP	32~100kHz	1	x	x
LP	1~2MHz	0	0	0
MP	2~4MHz	0	0	1
MS	4~8MHz	0	1	0
HS	8~16MHz	0	1	1

OPT 0 = **PLL** PLL selection

This option bit selects the PLL which allows multiplication by two of the oscillator frequency. The PLL must not be used with the internal RC oscillator. It is guaranteed only with a f<sub>OSC</sub> input frequency between 2 and 4MHz.

0: PLL x2 enabled  
1: PLL x2 disabled

**CAUTION:** the PLL can be enabled only if the "OSC RANGE" (OPT3:1) bits are configured to "MP - 2~4MHz". Otherwise, the device functionality is not guaranteed.

## 15.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE

Customer code is made up of the ROM/FAS-TROM contents and the list of the selected options (if any). The ROM/FASTROM contents are to be sent on diskette, or by electronic means, with the S19 hexadecimal file generated by the development tool. All unused bytes must be set to FFh. The selected options are communicated to

STMicroelectronics using the correctly completed OPTION LIST appended.

Refer to application note AN1635 for information on the counter listing returned by ST after code has been transferred.

The STMicroelectronics Sales Organization will be pleased to provide detailed information on contractual points.

**Table 27. Supported Part Numbers**

Part Number	Program Memory (Bytes)	RAM (Bytes)	Temp. Range	Package
ST72F264G1B6	4K FLASH	256	-40°C +85°C	SDIP32
ST72F264G1M6				SO28
ST72F262G1B6				SDIP32
ST72F262G1M6				SO28
ST72F264G2B6	8K FLASH	256	0°C +70°C	SDIP32
ST72F264G2M6				SO28
ST72F264G2H1			0°C +70°C	LFBGA
ST72F262G2B6			-40°C +85°C	SDIP32
ST72F262G2M6	SO28			
ST72F262G1B6	SDIP32			
ST72F262G1M6	SO28			
ST72F260G1B6	4K FLASH	256	-40°C +85°C	SDIP32
ST72F260G1M6				SO28
ST72P264G2B6/xxx	8K FASTROM	256	-40°C +85°C	SDIP32
ST72P264G2M6/xxx				SO28
ST72P264G2H1/xxx			0°C +70°C	LFBGA
ST72P262G2B6/xxx			-40°C +85°C	SDIP32
ST72P262G2M6/xxx				SO28
ST72P262G1B6/xxx	SDIP32			
ST72P262G1M6/xxx	4K FASTROM	256	-40°C +85°C	SO28
ST72P260G1B6/xxx				SDIP32
ST72P260G1M6/xxx				SO28
ST72264G2B6/xxx				8K ROM
ST72264G2M6/xxx	SO28			
ST72262G2B6/xxx	SDIP32			
ST72262G2M6/xxx	SO28			
ST72262G1B6/xxx	4K ROM	256	-40°C +85°C	SDIP32
ST72262G1M6/xxx				SO28
ST72260G1B6/xxx				SDIP32
ST72260G1M6/xxx				SO28

TRANSFER OF CUSTOMER CODE (Cont'd)

ST72264 ROM/FASTROM MICROCONTROLLER OPTION LIST

Customer .....  
 Address .....  
 Contact .....  
 Phone No .....  
 Reference /ROM or FAST ROM Code\* .....  
 ROM or FASTROM code is assigned by STMicroelectronics.  
 Code must be sent in .S19 format. .Hex extension cannot be processed.

Package/Memory:  
 SO28:  8K (G2M)  4K (G1M)  
 DIP32:  8K (G2B)  4K (G1B)  
 LFBGA6x6  8K (G2H)  
 Die Form:  8K (as G2)  4K (as G1)

Conditioning:  
 SO package:  Tape & Reel  Tube  
 Die form:  Tape & Reel  Inked wafer  Sawn wafer on sticky foil

Special Marking  No  
 Yes "-----" ( DIP32 15 ch, SO28 13 ch, LFBGA6x6 7 ch max.)"

Authorized characters are letters, digits '.', ',', '/' and spaces only.

Temperature Range:  
 Packaged form:  0°C to + 70°C  
 - 10°C to + 85°C (except LFBGA)  
 - 40°C to + 85°C (except LFBGA)

Die form: Tested at 25°C only  
 Watchdog Selection:  Software Activation  Hardware Activation  
 Watchdog Reset on Halt:  Reset  No Reset  
 LVD Reset  Disabled  Enabled:  
 Highest threshold  
 Medium threshold  
 Lowest threshold

Sector 0 Size:  0.5K  1K  2K  4K  
 Readout Protection:  Disabled  Enabled  
 External Interrupt:  Port C mapped to ei0 interrupt vector  
 Port C mapped to ei1 interrupt vector  
 Clock Security System:  Disabled  Enabled  
 Clock Source Selection:  Resonator:  VLP: Very Low power resonator (32 to 100 kHz)  
 LP: Low power resonator (1 to 2 MHz)  
 MP: Medium power resonator (2 to 4 MHz)  
 MS: Medium speed resonator (4 to 8 MHz)  
 HS: High speed resonator (8 to 16 MHz)

Internal RC Network<sup>1</sup>:  
 External Clock  
 PLL<sup>1</sup>:  Disabled  Enabled

Comments : .....

Supply Operating Range in the application: .....

Notes: .....

Signature: .....

<sup>1</sup>Use of the PLL with the internal RC oscillator is not supported.

**Important note:** Not all configurations are available. See Table 27 on page 159 for the list of supported part numbers.



### 15.3 DEVELOPMENT TOOLS

STMicroelectronics offers a range of hardware and software development tools for the ST7 microcontroller family. Full details of tools available for the ST7 from third party manufacturers can be obtained from the STMicroelectronics Internet site:

→ <http://www.stmcu.com>.

#### Third Party Tools

Tools from these manufacturers include C compilers, emulators and gang programmers.

#### ST Emulators

The emulator is delivered with everything (probes, TEB, adapters etc.) needed to start emulating the devices. To configure the emulator to emulate different ST7 subfamily devices, the active probe for the ST7 EMU3 can be changed and the ST7EMU3 probe is designed for easy interchange of TEBs (Target Emulation Board). See Table 28 for more details.

#### 15.3.1 Socket and Emulator Adapter Information

For information on the type of socket that is supplied with the emulator, refer to the suggested list of sockets in Table 29 and Table 30.

**Table 28. STMicroelectronics Development Tools**

Supported Products	Evaluation Board	ST7 Emulator	Active Probe & TEB	ST7 Programming Board
ST72F264, ST72F262, ST72F260	ST7FOPTIONS-EVAL	ST7MDT10-EMU3	ST7MDT10-TEB <sup>1</sup>	ST7MDT10-EPB <sup>2</sup>

#### Notes:

1. BGA adapter not available for ST7MDT10-EMU3.
2. ST7MDT10-EPB has no BGA socket, it can program BGA devices via ICC only.

## 15.3.2 PACKAGE/SOCKET FOOTPRINT PROPOSAL

Table 29. Suggested List of SDIP32 Socket Types

Package / Probe	Adaptor / Socket Reference	Same Footprint	Socket Type
SDIP32 EMU PROBE	TEXTTOOL                      232-1291-00	X	Textool

Table 30. Suggested List of SO28 Socket Types

Package / Probe	Adaptor / Socket Reference	Same Footprint	Socket Type
SO28	YAMAICHI                      IC51-0282-334-1		Clamshell
EMU PROBE	Adapter from SO28 to SDIP32 footprint (delivered with emulator)	X	SMD to SDIP

Table 31. Suggested LFBGA Socket Type

Package	Socket Reference
LFBGA 6 X6	ENPLAS OTB-36(144)-0.8-04

## 15.4 ST7 APPLICATION NOTES

IDENTIFICATION	DESCRIPTION
<b>EXAMPLE DRIVERS</b>	
AN 969	SCI COMMUNICATION BETWEEN ST7 AND PC
AN 970	SPI COMMUNICATION BETWEEN ST7 AND EEPROM
AN 971	I <sup>2</sup> C COMMUNICATING BETWEEN ST7 AND M24CXX EEPROM
AN 972	ST7 SOFTWARE SPI MASTER COMMUNICATION
AN 973	SCI SOFTWARE COMMUNICATION WITH A PC USING ST72251 16-BIT TIMER
AN 974	REAL TIME CLOCK WITH ST7 TIMER OUTPUT COMPARE
AN 976	DRIVING A BUZZER THROUGH ST7 TIMER PWM FUNCTION
AN 979	DRIVING AN ANALOG KEYBOARD WITH THE ST7 ADC
AN 980	ST7 KEYPAD DECODING TECHNIQUES, IMPLEMENTING WAKE-UP ON KEYSTROKE
AN1017	USING THE ST7 UNIVERSAL SERIAL BUS MICROCONTROLLER
AN1041	USING ST7 PWM SIGNAL TO GENERATE ANALOG OUTPUT (SINUSOID)
AN1042	ST7 ROUTINE FOR I <sup>2</sup> C SLAVE MODE MANAGEMENT
AN1044	MULTIPLE INTERRUPT SOURCES MANAGEMENT FOR ST7 MCUS
AN1045	ST7 S/W IMPLEMENTATION OF I <sup>2</sup> C BUS MASTER
AN1046	UART EMULATION SOFTWARE
AN1047	MANAGING RECEPTION ERRORS WITH THE ST7 SCI PERIPHERALS
AN1048	ST7 SOFTWARE LCD DRIVER
AN1078	PWM DUTY CYCLE SWITCH IMPLEMENTING TRUE 0% & 100% DUTY CYCLE
AN1082	DESCRIPTION OF THE ST72141 MOTOR CONTROL PERIPHERAL REGISTERS
AN1083	ST72141 BLDC MOTOR CONTROL SOFTWARE AND FLOWCHART EXAMPLE
AN1105	ST7 PCAN PERIPHERAL DRIVER
AN1129	PERMANENT MAGNET DC MOTOR DRIVE.
AN1130	AN INTRODUCTION TO SENSORLESS BRUSHLESS DC MOTOR DRIVE APPLICATIONS WITH THE ST72141
AN1148	USING THE ST7263 FOR DESIGNING A USB MOUSE
AN1149	HANDLING SUSPEND MODE ON A USB MOUSE
AN1180	USING THE ST7263 KIT TO IMPLEMENT A USB GAME PAD
AN1276	BLDC MOTOR START ROUTINE FOR THE ST72141 MICROCONTROLLER
AN1321	USING THE ST72141 MOTOR CONTROL MCU IN SENSOR MODE
AN1325	USING THE ST7 USB LOW-SPEED FIRMWARE V4.X
AN1445	USING THE ST7 SPI TO EMULATE A 16-BIT SLAVE
AN1475	DEVELOPING AN ST7265X MASS STORAGE APPLICATION
AN1504	STARTING A PWM SIGNAL DIRECTLY AT HIGH LEVEL USING THE ST7 16-BIT TIMER
<b>PRODUCT EVALUATION</b>	
AN 910	PERFORMANCE BENCHMARKING
AN 990	ST7 BENEFITS VERSUS INDUSTRY STANDARD
AN1077	OVERVIEW OF ENHANCED CAN CONTROLLERS FOR ST7 AND ST9 MCUS
AN1086	U435 CAN-DO SOLUTIONS FOR CAR MULTIPLEXING
AN1150	BENCHMARK ST72 VS PC16
AN1151	PERFORMANCE COMPARISON BETWEEN ST72254 & PC16F876
AN1278	LIN (LOCAL INTERCONNECT NETWORK) SOLUTIONS
<b>PRODUCT MIGRATION</b>	
AN1131	MIGRATING APPLICATIONS FROM ST72511/311/214/124 TO ST72521/321/324
AN1322	MIGRATING AN APPLICATION FROM ST7263 REV.B TO ST7263B
AN1365	GUIDELINES FOR MIGRATING ST72C254 APPLICATION TO ST72F264
<b>PRODUCT OPTIMIZATION</b>	

IDENTIFICATION	DESCRIPTION
AN 982	USING ST7 WITH CERAMIC RESONATOR
AN1014	HOW TO MINIMIZE THE ST7 POWER CONSUMPTION
AN1015	SOFTWARE TECHNIQUES FOR IMPROVING MICROCONTROLLER EMC PERFORMANCE
AN1040	MONITORING THE VBUS SIGNAL FOR USB SELF-POWERED DEVICES
AN1070	ST7 CHECKSUM SELF-CHECKING CAPABILITY
AN1324	CALIBRATING THE RC OSCILLATOR OF THE ST7FLITE0 MCU USING THE MAINS
AN1477	EMULATED DATA EEPROM WITH XFLASH MEMORY
AN1502	EMULATED DATA EEPROM WITH ST7 HDFLASH MEMORY
AN1529	EXTENDING THE CURRENT & VOLTAGE CAPABILITY ON THE ST7265 VDDF SUPPLY
AN1530	ACCURATE TIMEBASE FOR LOW-COST ST7 APPLICATIONS WITH INTERNAL RC OSCILLATOR
<b>PROGRAMMING AND TOOLS</b>	
AN 978	KEY FEATURES OF THE STVD7 ST7 VISUAL DEBUG PACKAGE
AN 983	KEY FEATURES OF THE COSMIC ST7 C-COMPILER PACKAGE
AN 985	EXECUTING CODE IN ST7 RAM
AN 986	USING THE INDIRECT ADDRESSING MODE WITH ST7
AN 987	ST7 SERIAL TEST CONTROLLER PROGRAMMING
AN 988	STARTING WITH ST7 ASSEMBLY TOOL CHAIN
AN 989	GETTING STARTED WITH THE ST7 HIWARE C TOOLCHAIN
AN1039	ST7 MATH UTILITY ROUTINES
AN1064	WRITING OPTIMIZED HIWARE C LANGUAGE FOR ST7
AN1071	HALF DUPLEX USB-TO-SERIAL BRIDGE USING THE ST72611 USB MICROCONTROLLER
AN1106	TRANSLATING ASSEMBLY CODE FROM HC05 TO ST7
AN1179	PROGRAMMING ST7 FLASH MICROCONTROLLERS IN REMOTE ISP MODE (IN-SITU PROGRAMMING)
AN1446	USING THE ST72521 EMULATOR TO DEBUG A ST72324 TARGET APPLICATION
AN1478	PORTING AN ST7 PANTA PROJECT TO CODEWARRIOR IDE
AN1527	DEVELOPING A USB SMARTCARD READER WITH ST7SCR
AN1575	ON-BOARD PROGRAMMING METHODS FOR XFLASH AND HDFLASH ST7 MCUS

## 16 SUMMARY OF CHANGES

Description of the changes between the current release of the specification and the previous one.

Rev.	Main changes	Date
1.6	Added LFBGA package on page 1 and throughout document Added RTC timer on page 1. Removed External RC oscillator option on page 1 and throughout document Added "can be reprogrammed" to "Memory Protection" on page 16 and "OPTION BYTES" on page 157. Added "freerunning" to description of counter in "WATCHDOG TIMER (WDG)" on page 48 Changed reset value of SCIBRR register in "Hardware Register Map" on page 11 Changed SLOW and SPEED bit table in ADC "Register Description" on page 114 Removed references to Temp. range -40 to +125°C and -40 to +105°C in "OPERATING CONDITIONS" on page 124 and throughout section. Changed Vtpor values in "Low Voltage Detector (LVD) Thresholds" on page 125 Changed EMS and EMI values in "EMC CHARACTERISTICS" on page 136 Changed option list and added mention of FASTROM to "DEVICE CONFIGURATION AND ORDERING INFORMATION" on page 157 Added "ERRATA SHEET" on page 166	April-03
1.7	Modified description of internal RC oscillator in Section 6.2 on page 21. Added Caution about disconnecting OSC pins in Section 6.2 on page 21 Added note on GCAL to I2C Section 11.6 on page 99. Added note on ADC data register in Table 2 on page 11 Updated Errata sheet	Aug-03



## ST72264 LIMITATIONS AND CORRECTIONS

### 17 SILICON IDENTIFICATION

This document refers to ST72F264 devices and subsets (ST72F260, ST72F262).

They are identifiable:

- On the device package, by the last letter of the **Trace code** marked on the device package
- On the box, by the last 3 digits of the **Internal Sales Type** printed on the box label.

**Table 32. Device Identification**

	Trace Code marked on device	Internal Sales Type on box label
<b>Flash Devices:</b>	"XXXXXXXXXZ"	72F264xxxx\$U2 72F264xxxx\$A2

### 18 REFERENCE SPECIFICATION

Limitations in this document are with reference to the ST72264 Datasheet Revision 1.7 (August 2003).

### 19 SILICON LIMITATIONS

This section lists the known limitations of the devices referenced in Table 32.

#### 19.1 EXECUTION OF BTJX INSTRUCTION

When testing the address \$FF with the "BTJT" or "BTJF" instructions, the CPU may perform an incorrect operation when the relative jump is negative and performs an address page change.

To avoid this issue, including when using a C compiler, it is recommended to never use address \$00FF as a variable (using the linker parameter for example).

#### 19.2 I/O PORT B AND C CONFIGURATION

When using an external quartz crystal or ceramic resonator, the  $f_{OSC2}$  clock may be disturbed because the device goes into reserved mode controlled by Port B and C.

This happens with either one of the following configurations:

- PB1=0, PC2=1, PB3=0 while CSS and PLL options are both disabled and PC4 is toggling
- PB1=0, PC2=1, PB3=0, PC4=1 while CSS or PLL options are enabled

This is detailed in the following table:

CSS	PLL	PB1	PC2	PB3	PC4	Clock Disturbance
x	x	0	1	0	Toggling	Max. 2 clock cycles lost at each rising or falling edge of PC4
x	ON	0	1	0	1	Max. 1 clock cycle lost out of every 16
ON	x					

As a consequence, for cycle-accurate operations, these configurations are prohibited in either input or output mode.

#### Workaround:

To avoid this occurring, it is recommended to connect one of these pins to GND (PC2 or PC4) or V<sub>DD</sub> (PB1 or PB3).

### 19.3 16-BIT TIMER PWM MODE

In PWM mode, the first PWM pulse is missed after writing the value FFFCh in the OC12R register.

### 19.4 SPI MULTIMASTER MODE

Multi master mode is not supported.

### 19.5 MINIMUM OPERATING VOLTAGE

The minimum V<sub>DD</sub> voltage is 2.7V.

### 19.6 CSS FUNCTION

The Clock Security System is not guaranteed. The features described in Section 6.4.3 are subject to revision.

### 19.7 INTERNAL AND EXTERNAL RC OSCILLATOR WITH LVD

If the LVD is disabled, the internal or external RC oscillator clock source cannot be used.

In ICP mode, new flash devices must be programmed with an external clock connected to the OSC1 pin or using a crystal or ceramic resonator. In the STVP7 programming tool software, select the “OPTIONS DISABLED” mode.

### 19.8 EXTERNAL CLOCK WITH PLL

The PLL option is not supported for use with external clock source.

### 19.9 HALT MODE POWER CONSUMPTION WITH ADC ON

If the A/D converter is being used when Halt mode is entered, the power consumption in Halt Mode may exceed the maximum specified in the datasheet.

#### Workaround

Switch off the ADC by software (ADON=0) before executing a HALT instruction.

### 19.10 ACTIVE HALT WAKE-UP BY EXTERNAL INTERRUPT

External interrupts are not able to wake-up the MCU from Active Halt mode. The MCU can only exit from Active Halt mode by means of an MCC/RTC interrupt or a reset.

#### Workaround

Use WAIT mode if external interrupt capability is required in low power mode.

### 19.11 A/D CONVERTER ACCURACY FOR FIRST CONVERSION

When the ADC is enabled after being powered down (for example when waking up from HALT, ACTIVE-HALT or setting the ADON bit in the ADCCSR register), the first conversion (8-bit or 10-bit) accuracy does not meet the accuracy specified in the data sheet.

#### Workaround

In order to have the accuracy specified in the datasheet, the first conversion after a ADC switch-on has to be ignored.

### 19.12 NEGATIVE INJECTION IMPACT ON ADC ACCURACY

Injecting a negative current on an analog input pins significantly reduces the accuracy of the AD Converter. Whenever necessary, the negative injection should be prevented by the addition of a Schottky diode between the concerned I/Os and ground.

Injecting a negative current on digital input pins degrades ADC accuracy especially if performed on a pin close to ADC channel in use.

### 19.13 ADC CONVERSION SPURIOUS RESULTS

Spurious conversions occur with a rate lower than 50 per million. Such conversions happen when the measured voltage is just between 2 consecutive digital values.

#### Workaround

A software filter should be implemented to remove erratic conversion results whenever they may cause unwanted consequences.



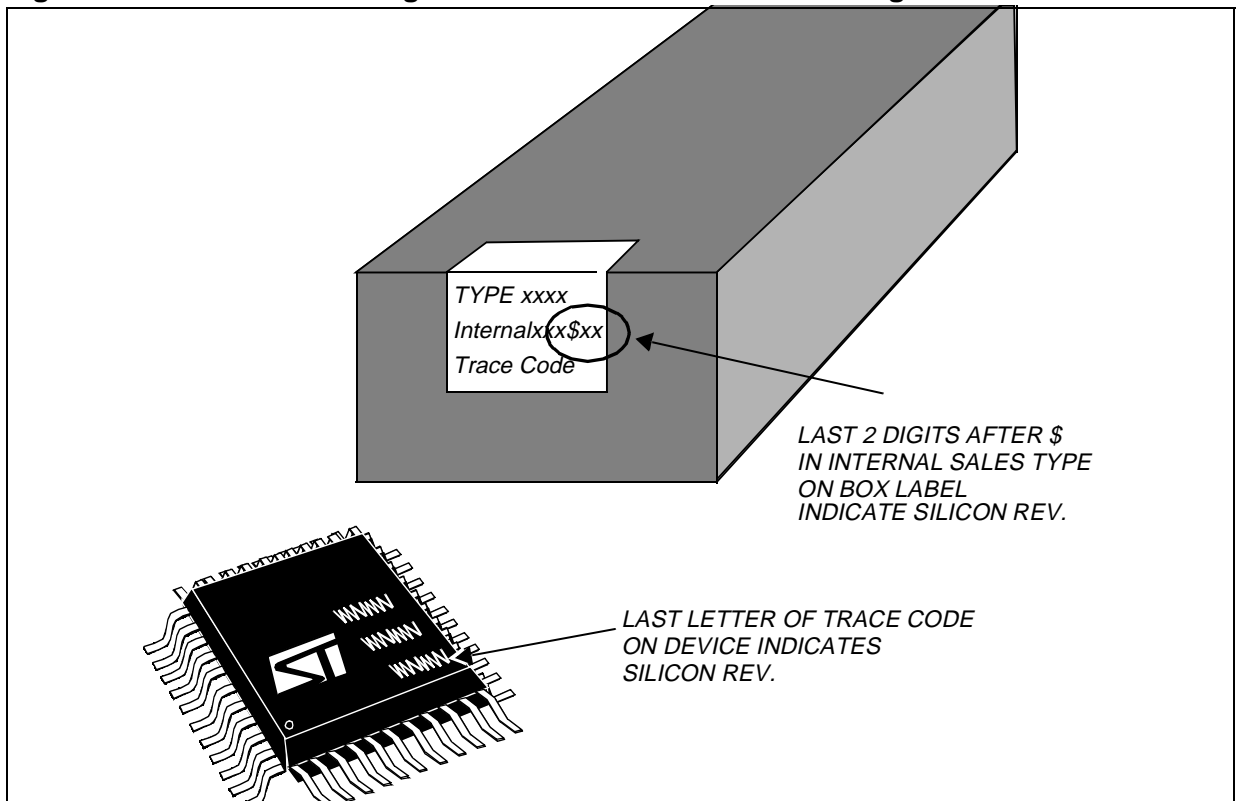
## 19.14 FUNCTIONAL EMS

Functional EMS (Electro Magnetic Susceptibility) levels do not reach the ST standards.

Special care should be taken when designing the PCB layout and firmware (refer to application notes AN898, AN901 and AN1015) in sensitive applications (that use switches for instance). For more information refer to application note AN1637.

## 20 DEVICE MARKING

Figure 114. Revision Marking on Box Label and Device Marking



## 21 ERRATA SHEET REVISION HISTORY

Revision	Main Changes	Date
2.3	Modified Section 19.5 "MINIMUM OPERATING VOLTAGE" on page 167 Added "16-BIT TIMER PWM MODE" on page 167 Added "FUNCTIONAL EMS" on page 169	05/06/03
2.4	Removed External RC Oscillator section Added "CSS Function not guaranteed" section	08/07/03

**Notes:**

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without the express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics

©2003 STMicroelectronics - All Rights Reserved.

Purchase of I<sup>2</sup>C Components by STMicroelectronics conveys a license under the Philips I<sup>2</sup>C Patent. Rights to use these components in an I<sup>2</sup>C system is granted provided that the system conforms to the I<sup>2</sup>C Standard Specification as defined by Philips.

STMicroelectronics Group of Companies

Australia - Brazil - Canada - China - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan  
Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - U.S.A.

<http://www.st.com>