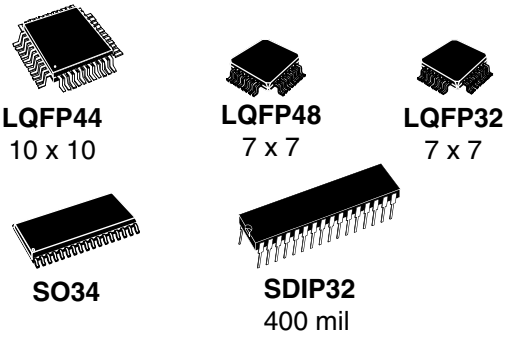
ST72323
ST72323L

3 V/5 V range 8-bit MCU with 4/8 Kbyte ROM, 10-bit ADC, four timers and SPI

Features

- Memories
  - 4/8K ROM with read-out protection capability
  - 384 bytes RAM
  - Compatible with Flash superset ST72F324B
- Clock, reset and supply management
  - Clock sources: crystal/ceramic resonator oscillators, internal RC oscillator and bypass for external clock
  - Four power saving modes: Halt, Active-Halt, Wait and Slow
- Interrupt management
  - Nested interrupt controller
  - 10 interrupt vectors plus TRAP and RESET
  - 9/6 external interrupt lines (on 4 vectors)
- Up to 32 I/O ports
  - 32/24 multifunctional bidirectional I/O lines
  - 22/17 alternate function lines
  - 12/10 high sink outputs
- 4 Timers
  - Main clock controller with: real-time base, beep and clock-out capabilities
  - Configurable watchdog timer
  - 16-bit timer A with: 1 input capture, 1 output compare, external clock input, PWM and pulse generator modes
  - 16-bit timer B with: 2 input captures, 2 output compares, PWM and pulse generator modes



- Communications interface
  - SPI synchronous serial interface
- 1 analog peripheral (low-current coupling)
  - 10-bit ADC with up to 12 robust input ports
- 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
  - In-circuit testing capability

Contents

List of tables

List of figures

1 Description

ST72323L and ST72323 ROM devices are members of the ST7 microcontroller family designed for the 3 V and 5 V operating range.

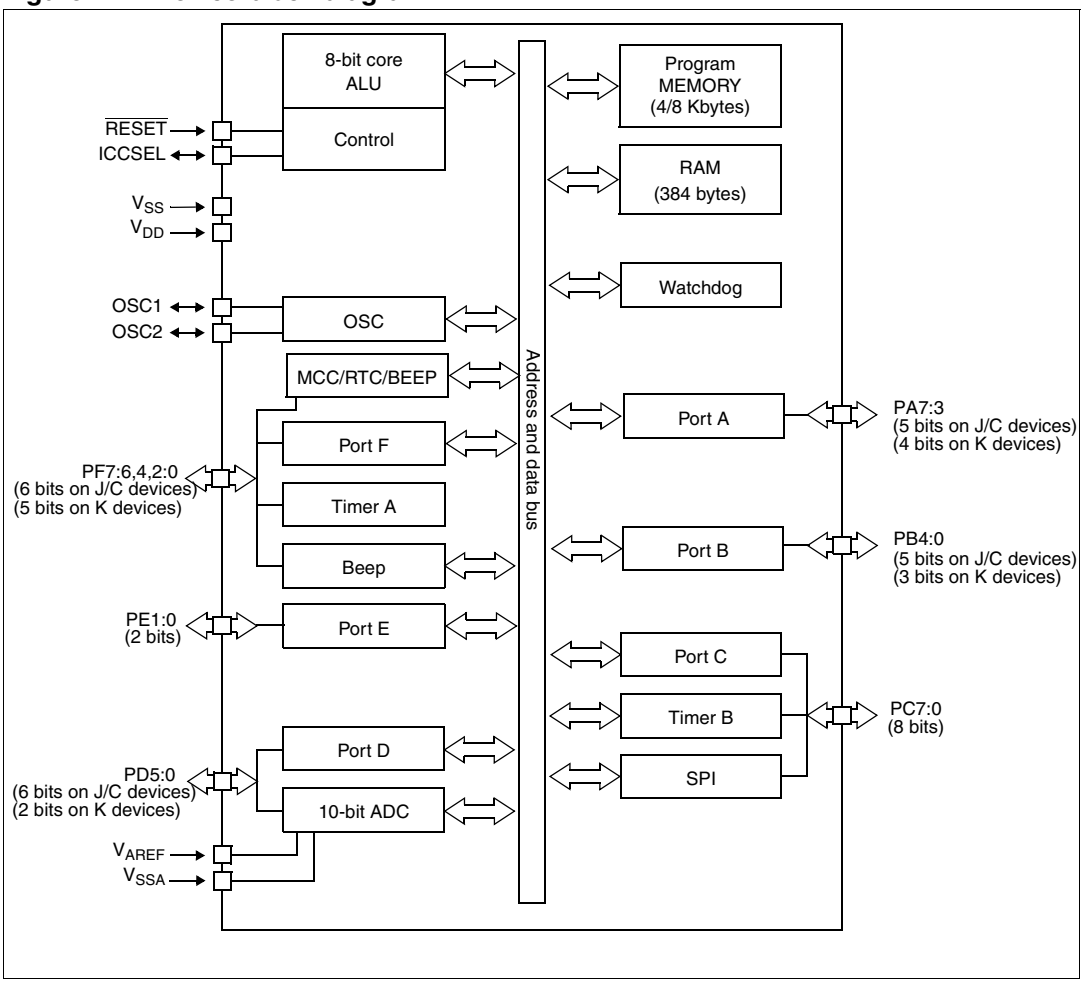
- The 32/34-pin (K) devices are designed for mid-range applications
- The 42/44/48-pin (J and C) devices target the same range of applications requiring more than 24 I/O ports.

All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set and are available with ROM program memory. The ST7 family architecture offers both power and flexibility to software developers, enabling the design of highly efficient and compact application code.

The on-chip peripherals include an A/D converter, two general purpose timers and an SPI interface. For power economy, the microcontroller can switch dynamically into, Slow, Wait, Active Halt or Halt mode when the application is in idle or stand-by state.

Typical applications include consumer, home, office and industrial products.

Figure 1. Device block diagram



2 Pin description

Figure 2. 48-pin device pinout

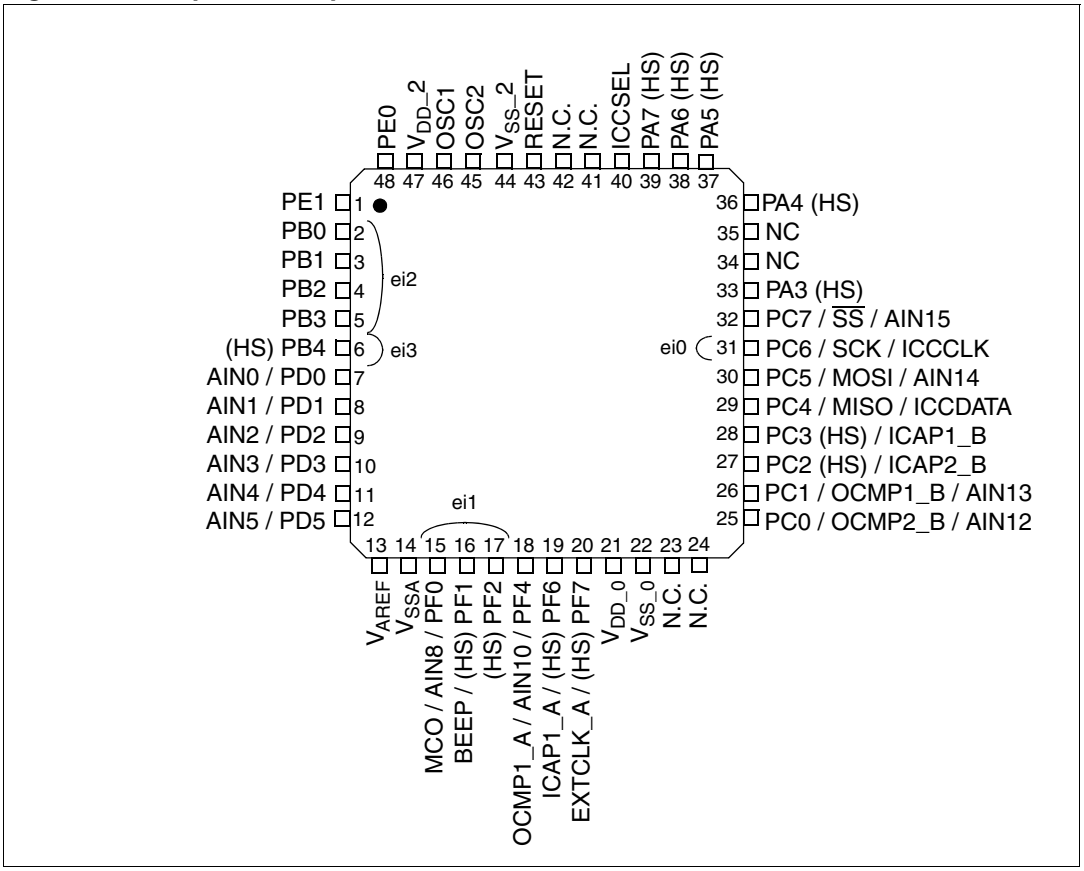
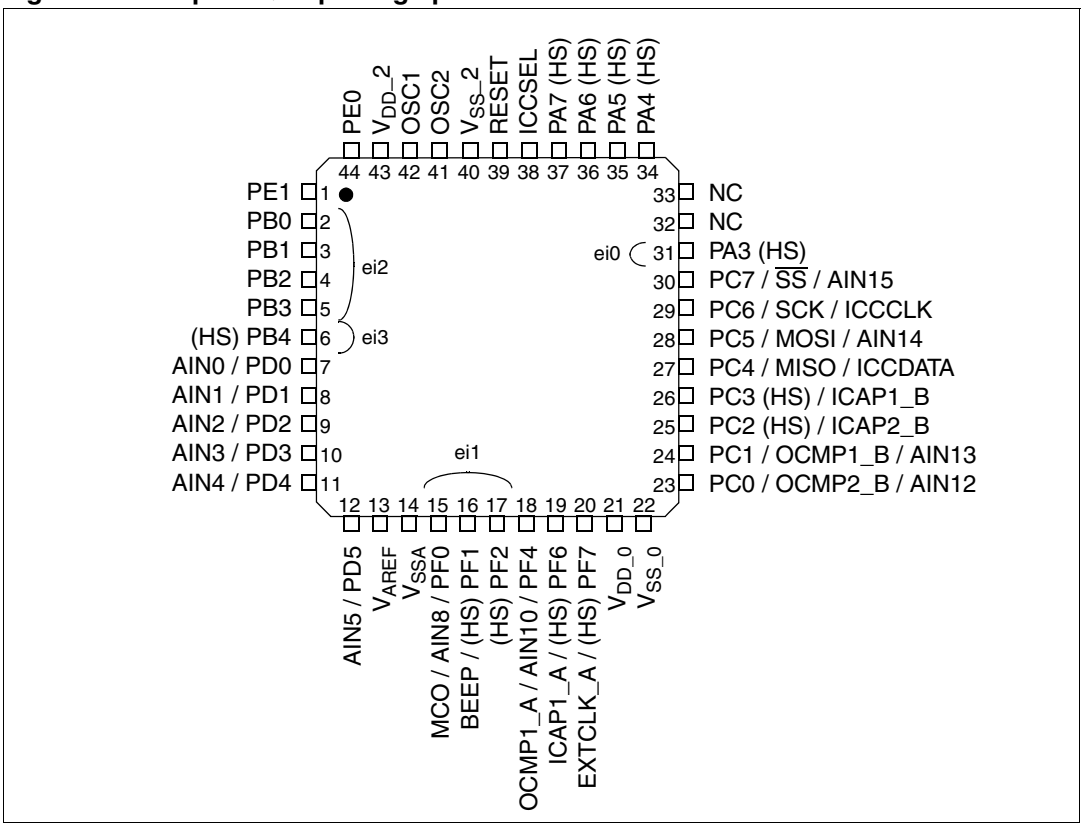
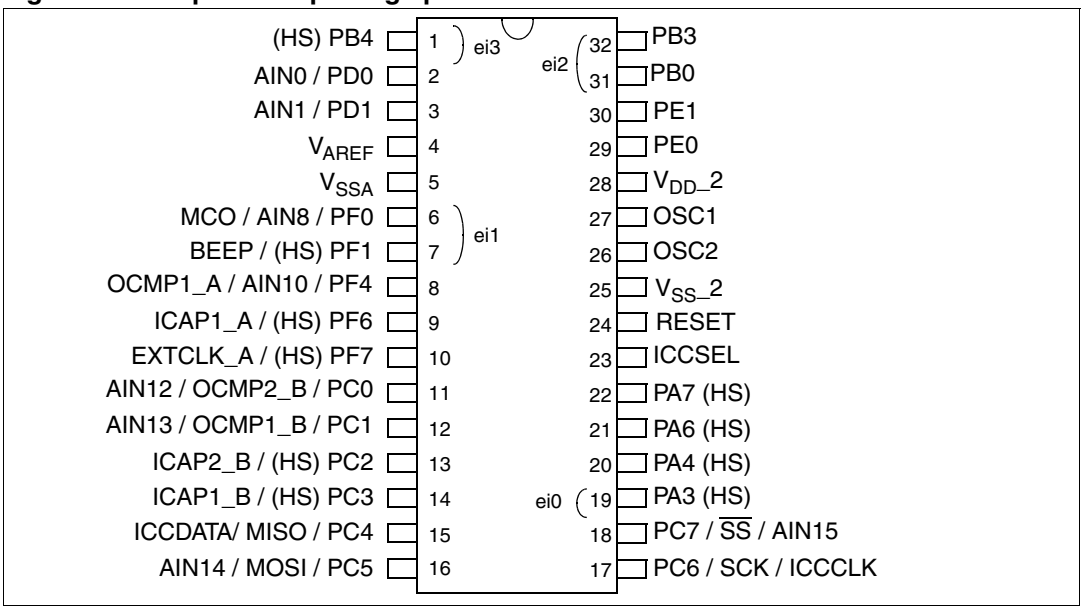


Figure 3. 44-pin LQFP package pinouts



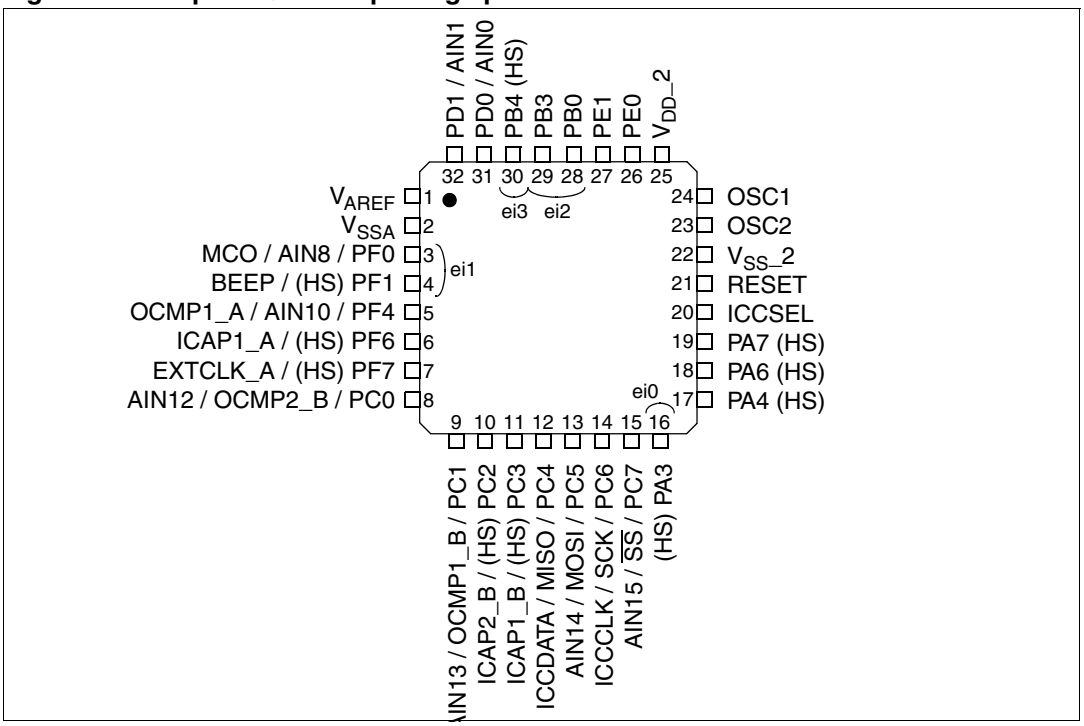
1. HS: 20 mA high-sink capability.
2. eix: associated external interrupt vector.

Figure 4. 32-pin SDIP package pinout



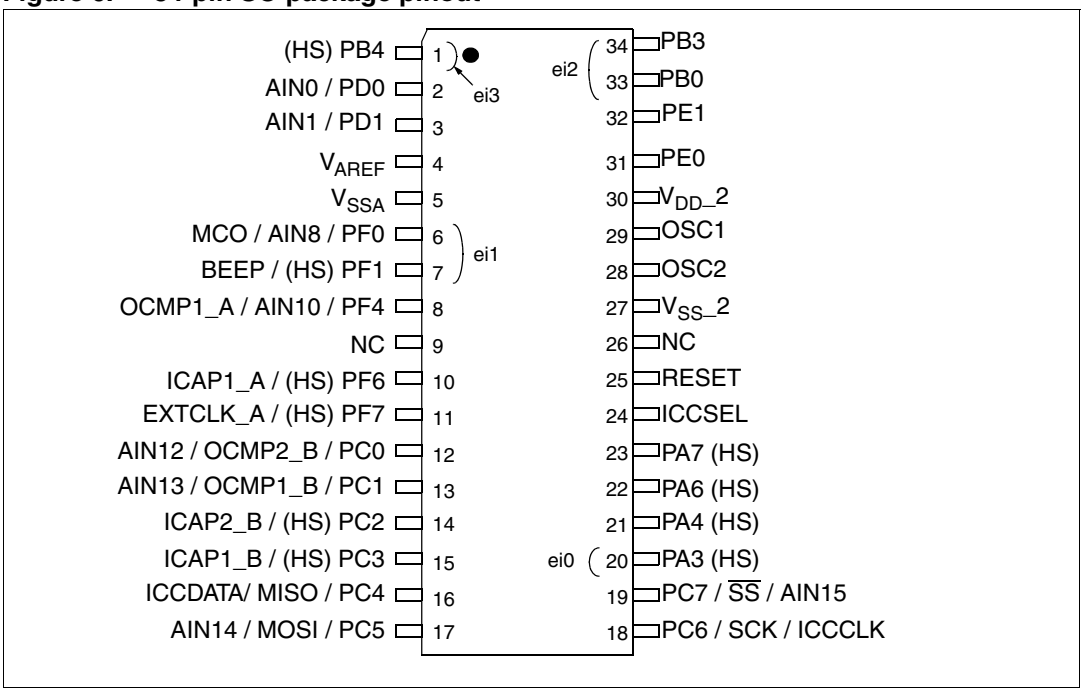
1. HS: 20 mA high-sink capability.
2. eix: associated external interrupt vector.

Figure 5. 32-pin LQFP 7x7 package pinout



1. HS: 20 mA high-sink capability.
2. eix: associated external interrupt vector.

Figure 6. 34-pin SO package pinout



1. HS: 20 mA high-sink capability.
2. eix: associated external interrupt vector.

For external pin connection guidelines, refer to *Section 11: Electrical characteristics on page 114*

Refer to *I/O ports on page 49* 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 no.					Pin name	Type(1)	Level(2)		Port(3)						Main function (after reset)	Alternate function	
									Input(5)				Output(6)				
LQFP48	LQFP44	LQFP32	SDIP32	SO34			Input	Output(4)	float	wpu	int	ana	OD(7)	PP			
6	6	30	1	1	PB4 (HS)	I/O	C_T	HS	X	ei3			X	X	Port B4		
7	7	31	2	2	PD0/AIN0	I/O	C_T		X	X		X	X	X	Port D0	ADC analog input 0	
8	8	32	3	3	PD1/AIN1	I/O	C_T		X	X		X	X	X	Port D1	ADC analog input 1	
9	9				PD2/AIN2	I/O	C_T		X	X		X	X	X	Port D2	ADC analog input 2	
10	10				PD3/AIN3	I/O	C_T		X	X		X	X	X	Port D3	ADC analog input 3	
11	11				PD4/AIN4	I/O	C_T		X	X		X	X	X	Port D4	ADC analog input 4	
12	12				PD5/AIN5	I/O	C_T		X	X		X	X	X	Port D5	ADC analog input 5	
13	13	1	4	4	V_{AREF}	S									Analog reference voltage for ADC		
14	14	2	5	5	V_{SSA}(8)	S									Analog ground voltage		
15	15	3	6	6	PF0/MCO/AIN8	I/O	C_T		X	ei1		X	X	X	Port F0	Main clock out (f_{CPU})	ADC analog input 8
16	16	4	7	7	PF1 (HS)/BEEP	I/O	C_T	HS	X	ei1			X	X	Port F1	Beep signal output	
17	17				PF2 (HS)	I/O	C_T	HS	X		ei1		X	X	Port F2		
18	18	5	8	8	PF4/OCMP1_A/ AIN10	I/O	C_T		X	X		X	X	X	Port F4	Timer A output compare 1	ADC analog input 10
19	19	6	9	10	PF6 (HS)/ ICAP1_A	I/O	C_T	HS	X	X			X	X	Port F6	Timer A input capture 1	
20	20	7	10	11	PF7 (HS)/ EXTCLK_A	I/O	C_T	HS	X	X			X	X	Port F7	Timer A external clock source	
21	21				V_{DD_0}(8)	S									Digital main supply voltage		
22	22				V_{SS_0}(8)	S									Digital ground voltage		
25	23	8	11	12	PC0/OCMP2_B/ AIN12	I/O	C_T		X	X		X	X	X	Port C0	Timer B output compare 2	ADC analog input 12
26	24	9	12	13	PC1/OCMP1_B/ AIN13	I/O	C_T		X	X		X	X	X	Port C1	Timer B output compare 1	ADC analog input 13

Table 1. Device pin description (continued)

Pin no.					Pin name	Type[1]	Level[2]		Port[3]						Main function (after reset)	Alternate function	
							Input	Output[4]	Input[5]				Output[6]				
LQFP48	LQFP44	LQFP32	SDIP32	SO34					float	wpu	int	ana	OD[7]	PP			
27	25	10	13	14	PC2 (HS)/ ICAP2_B	I/O	C_T	HS	X	X			X	X	Port C2	Timer B input capture 2	
28	26	11	14	15	PC3 (HS)/ ICAP1_B	I/O	C_T	HS	X	X			X	X	Port C3	Timer B input capture 1	
29	27	12	15	16	PC4/MISO/ ICCDATA	I/O	C_T		X	X			X	X	Port C4	SPI master in / slave out data	ICC data input
30	28	13	16	17	PC5/MOSI/ AIN14	I/O	C_T		X	X		X	X	X	Port C5	SPI master out / slave in data	ADC analog input 14
31	29	14	17	18	PC6/SCK/ ICCCLK	I/O	C_T		X	X			X	X	Port C6	SPI serial clock	ICC clock output
32	30	15	18	19	PC7/$\overline{SS}$/AIN15	I/O	C_T		X	X		X	X	X	Port C7	SPI Slave Select (active low)	ADC analog input 15
33	31	16	19	20	PA3 (HS)	I/O	C_T	HS	X		ei0		X	X	Port A3		
34	32				NC	S									Not connected		
35	33				NC	S									Not connected		
36	34	17	20	21	PA4 (HS)	I/O	C_T	HS	X	X			X	X	Port A4		
37	35				PA5 (HS)	I/O	C_T	HS	X	X			X	X	Port A5		
38	36	18	21	22	PA6 (HS)	I/O	C_T	HS	X				T		Port A6[9]		
39	37	19	22	23	PA7 (HS)	I/O	C_T	HS	X				T		Port A7[9]		
40	38	20	23	24	ICCSEL	I									Must be tied low.		
43	39	21	24	25	RESET	I/O	C_T								Top priority non maskable interrupt.		
44	40	22	25	27	V_{SS_2}[8]	S									Digital ground voltage		
45	41	23	26	28	OSC2[10]	O									Resonator oscillator inverter output		
46	42	24	27	29	OSC1[10]	I									External clock input or Resonator oscillator inverter input		
47	43	25	28	30	V_{DD_2}[8]	S									Digital main supply voltage		
48	44	26	29	31	PE0	I/O	C_T		X	X			X	X	Port E0		
1	1	27	30	32	PE1	I/O	C_T		X	X			X	X	Port E1		
2	2	28	31	33	PB0	I/O	C_T		X	ei2			X	X	Port B0		
3	3				PB1	I/O	C_T		X	ei2			X	X	Port B1		

Table 1. Device pin description (continued)

Pin no.					Pin name	Type(1)	Level(2)		Port(3)						Main function (after reset)	Alternate function
									Input(5)				Output(6)			
LQFP48	LQFP44	LQFP32	SDIP32	SO34			Input	Output(4)	float	wpu	int	ana	OD(7)	PP		
4	4				PB2	I/O	C_T		X	ei2			X	X	Port B2	
5	5	29	32	34	PB3	I/O	C_T		X		ei2		X	X	Port B3	

1. I = input, O = output, S = supply.
2. C = CMOS $0.3V_{DD}/0.7V_{DD}$

   CT= CMOS $0.3V_{DD}/0.7V_{DD}$ with input trigger
3. On the chip, each I/O port has 8 pads. Pads that are not bonded to external pins are in input pull-up configuration after reset. The configuration of these pads must be kept at reset state to avoid added current consumption.
4. HS = 20 mA high sink (on N-buffer only)
5. float = floating, wpu = weak pull-up, int = interrupt 1), ana = analog ports
6. OD = open drain 2), PP = push-pull.
7. In the open drain output column, “T” defines a true open drain I/O (P-Buffer and protection diode to VDD are not implemented). See*Section 8: I/O ports on page 49*. and *Section 11: Electrical characteristics on page 114* for more details.
8. It is mandatory to connect all available V_{DD} and V_{DD_x} pins to the supply voltage and all V_{SS} and V_{SSA} pins to ground.
9. 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 pull-up interrupt input, else the configuration is floating interrupt input.
10. OSC1 and OSC2 pins connect a crystal/ceramic resonator, or an external source to the on-chip oscillator; see *Section 2: Pin description* and *Section 11.5: Clock and timing characteristics* for more details.

3 Register and memory map

As shown in *Figure 7*, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 128 bytes of register locations, 386 bytes of RAM and 8Kbytes of user program memory. The RAM space includes up to 256 bytes for the stack from 0100h to 01FFh.

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

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

Figure 7. Memory map

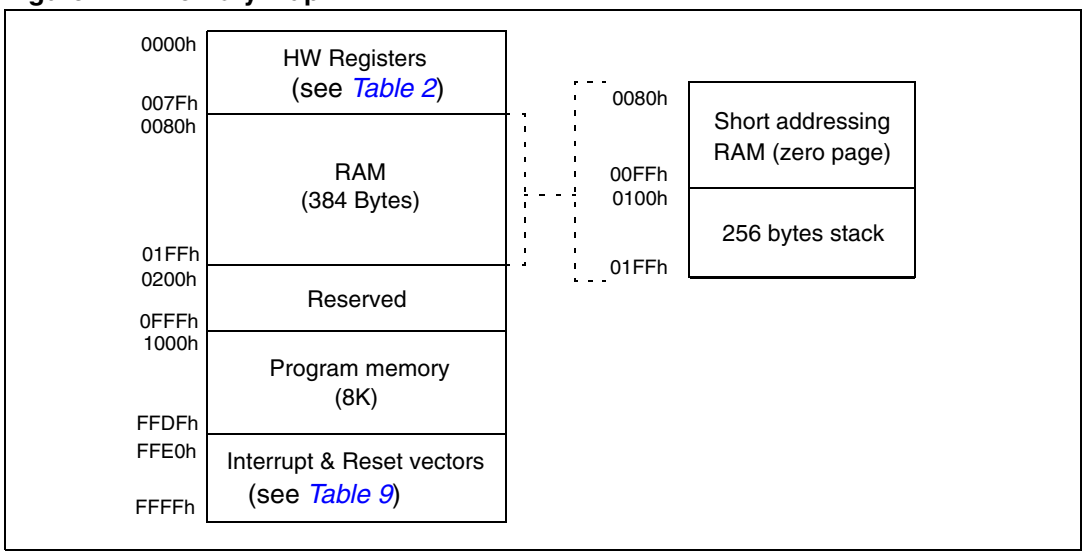


Table 2. Hardware register map(1)

Address	Block	Register label	Register name	Reset status	Remarks
0000h 0001h 0002h	Port A(3)	PADR PADDR PAOR	Port A Data Register Port A Data Direction Register Port A Option Register	00h(2) 00h 00h	R/W R/W R/W
0003h 0004h 0005h	Port B (3)	PBDR PBDDR PBOR	Port B Data Register Port B Data Direction Register Port B Option Register	00h(2) 00h 00h	R/W R/W R/W
0006h 0007h 0008h	Port C	PCDR PCDDR PCOR	Port C Data Register Port C Data Direction Register Port C Option Register	00h(2) 00h 00h	R/W R/W R/W
0009h 000Ah 000Bh	Port D (3)	PDADR PDDDR PDOR	Port D Data Register Port D Data Direction Register Port D Option Register	00h(2) 00h 00h	R/W R/W R/W
000Ch 000Dh 000Eh	Port E (3)	PEDR PEDDR PEOR	Port E Data Register Port E Data Direction Register Port E Option Register	00h(2) 00h 00h	R/W R/W(3) R/W(3)

Table 2. Hardware register map(1) (continued)

Address	Block	Register label	Register name	Reset status	Remarks
000Fh	Port F (3)	PFDR	Port F Data Register	00h(2)	R/W
0010h		PFDDR	Port F Data Direction Register	00h	R/W
0011h		PFOR	Port F Option Register	00h	R/W
0012h to 0020h	Reserved area (15 bytes)				
0021h	SPI	SPIDR	SPI Data I/O Register	xxh	R/W
0022h		SPICR	SPI Control Register	0xh	R/W
0023h		SPICSR	SPI Control/Status Register	00h	R/W
0024h	ITC	ISPR0	Interrupt Software Priority Register 0	FFh	R/W
0025h		ISPR1	Interrupt Software Priority Register 1	FFh	R/W
0026h		ISPR2	Interrupt Software Priority Register 2	FFh	R/W
0027h		ISPR3	Interrupt Software Priority Register 3	FFh	R/W
0028h		EICR	External Interrupt Control Register	00h	R/W
0029h	Reserved Area (1 byte)				
002Ah	WATCHDOG	WDGCR	Watchdog Control Register	7Fh	R/W
002Bh	Reserved area (1 byte)				
002Ch	MCC	MCCSR	Main Clock Control / Status Register	00h	R/W
002Dh		MCCBCR	Main Clock Controller: Beep Control Register	00h	R/W
002Eh to 0030h	Reserved area (3 bytes)				
0031h	TIMER A	TACR2	Timer A Control Register 2	00h	R/W
0032h		TACR1	Timer A Control Register 1	00h	R/W
0033h		TACSR	Timer A Control/Status Register	xxxx x0xxb	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		TACLR	Timer A Counter Low Register	FCh	Read Only
003Ah		TAACHR	Timer A Alternate Counter High Register	FFh	Read Only
003Bh		TAACLR	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	Reserved Area (1 Byte)				

Table 2. Hardware register map(1) (continued)

Address	Block	Register label	Register name	Reset status	Remarks
0041h	TIMER B	TBCR2	Timer B Control Register 2	00h	R/W
0042h		TBCR1	Timer B Control Register 1	00h	R/W
0043h		TBCSR	Timer B Control/Status Register	xxxx x0xxb	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		TBCLR	Timer B Counter Low Register	FCh	Read Only
004Ah		TBACHR	Timer B Alternate Counter High Register	FFh	Read Only
004Bh		TBACLR	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 to 006Fh	Reserved area (32 bytes)				
0070h	ADC	ADCCSR	Control/Status Register	00h	R/W
0071h		ADCDRH	Data High Register	00h	Read Only
0072h		ADCDRL	Data Low Register	00h	Read Only
0073h 007Fh	Reserved area (13 bytes)				

1. x = undefined, R/W = read/write
2. 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.
3. The bits associated with unavailable pins must always keep their reset value.

4 Central processing unit

4.1 Introduction

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

4.2 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

4.3 CPU registers

The 6 CPU registers shown in *Figure 8* are not present in the memory mapping and are accessed by specific instructions.

4.3.1 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.

4.3.2 Index registers (X and Y)

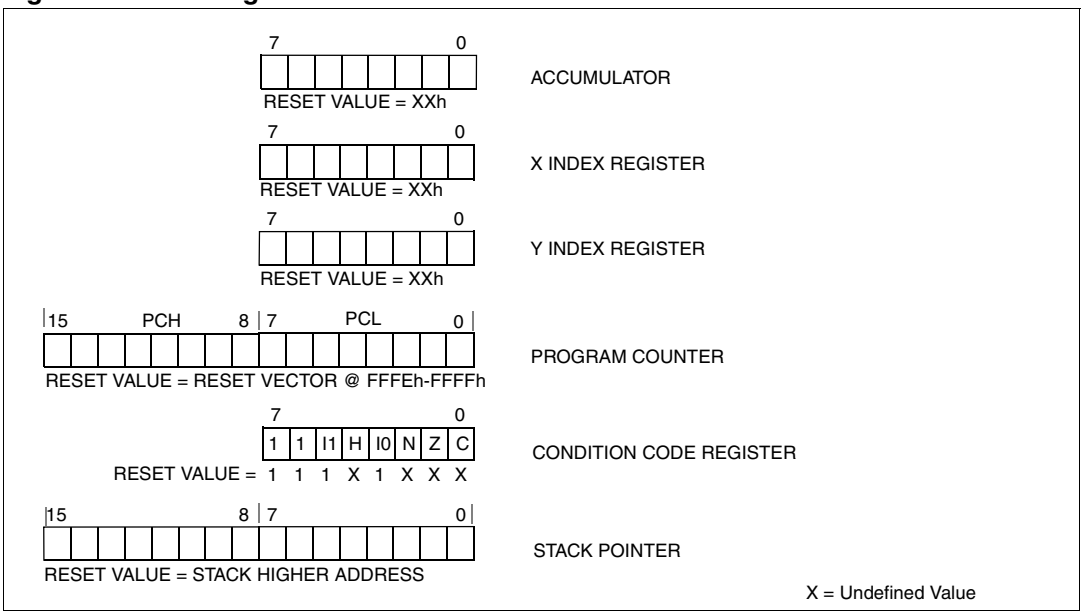
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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

4.3.3 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 8. CPU registers



4.3.4 Condition code register (CC)

Reset: 111x1xxx

7							0
1	1	I1	H	I0	N	Z	C
Read/write							

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^{th} 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

Bits 5 and 3 = **I1**, **I0** *Interrupt*

The combination of the I1 and I0 bits gives the current interrupt software priority (see *Table 3*).

Table 3. 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 *Section 6: Interrupts on page 29* for more details.

4.3.5 Stack pointer (SP)

Reset: 01 FFh

15							8	7							0
0	0	0	0	0	0	0	1	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
Read/Write															

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 9*).

Since the stack is 256 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.

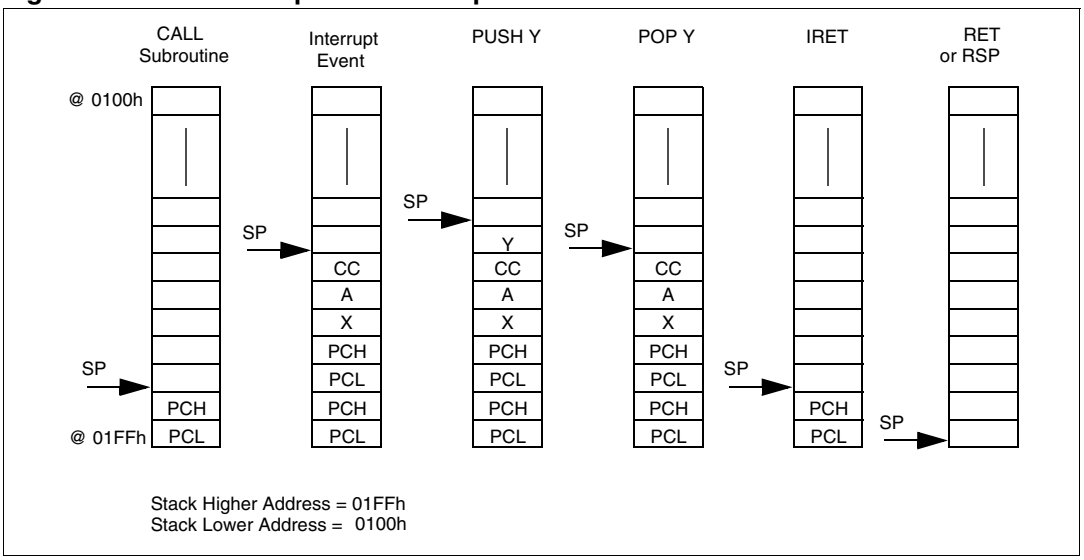
Note: When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

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 9*.

- 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 9. Stack manipulation example



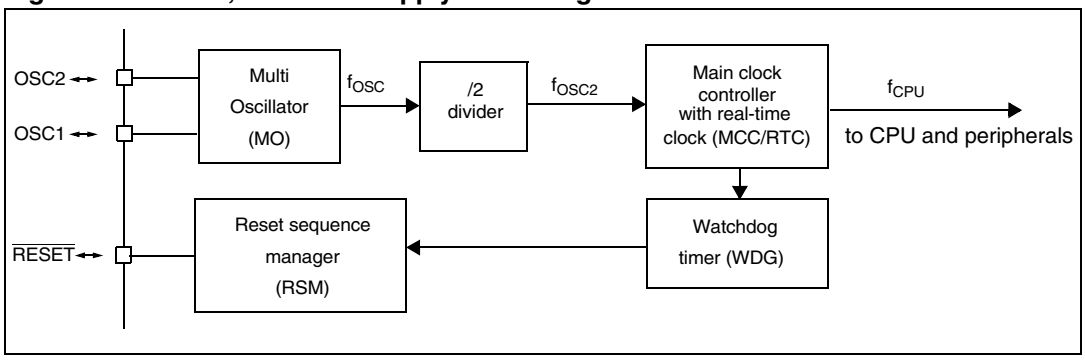
5 Supply, reset and clock management

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*.

Main features

- Reset sequence manager (RSM)
- Multioscillator clock management (MO)
  - 5 crystal/ceramic resonator oscillators
  - 1 Internal RC oscillator

Figure 10. Clock, reset and supply block diagram



5.1 Multioscillator (MO)

The main clock of the ST7 can be generated by three different source types coming from the multioscillator block:

- an external source
- 4 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 4*. Refer to *Section 11: Electrical characteristics* for more details.

Caution: The OSC1 and/or OSC2 pins must not be left unconnected. For the purposes of Failure Mode and Effect 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 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 ~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 4 oscillators with different frequency ranges has to be done by option byte in order to reduce consumption (refer to *Section 13.1 on page 153* for more details on the frequency ranges). In this mode of the multioscillator, 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.

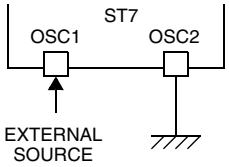
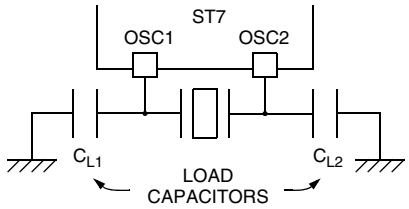
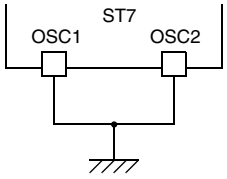
These oscillators are not stopped during the RESET phase to avoid losing time in the oscillator start-up phase.

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 4. ST7 clock sources

Clock sources	Hardware configuration
External clock	
Crystal/ceramic resonators	
Internal RC oscillator	

5.2 Reset sequence manager (RSM)

5.2.1 Introduction

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

- External $\overline{\text{RESET}}$ source pulse
- 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*:

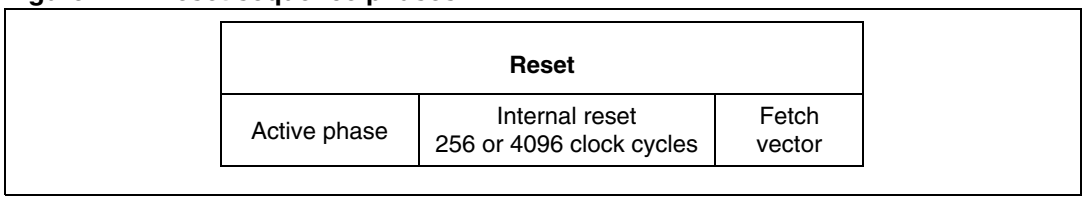
- Active Phase depending on the RESET source
- 256 or 4096 CPU clock cycle delay (selected by option byte)
- RESET vector fetch

Caution: When the ST72323 is unprogrammed or fully erased, the Flash is blank and the RESET vector is not programmed. For this reason, it is recommended to keep the $\overline{\text{RESET}}$ pin in low state until programming mode is entered, in order to avoid unwanted behavior.

The 256 or 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.

The RESET vector fetch phase duration is 2 clock cycles.

Figure 11. Reset sequence phases



5.2.2 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 *Section 11: Electrical characteristics* 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. This detection is asynchronous and therefore the MCU can enter reset state even in Halt mode.

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 *Section 11: Electrical characteristics*.

5.2.3 External power-on reset (POR)

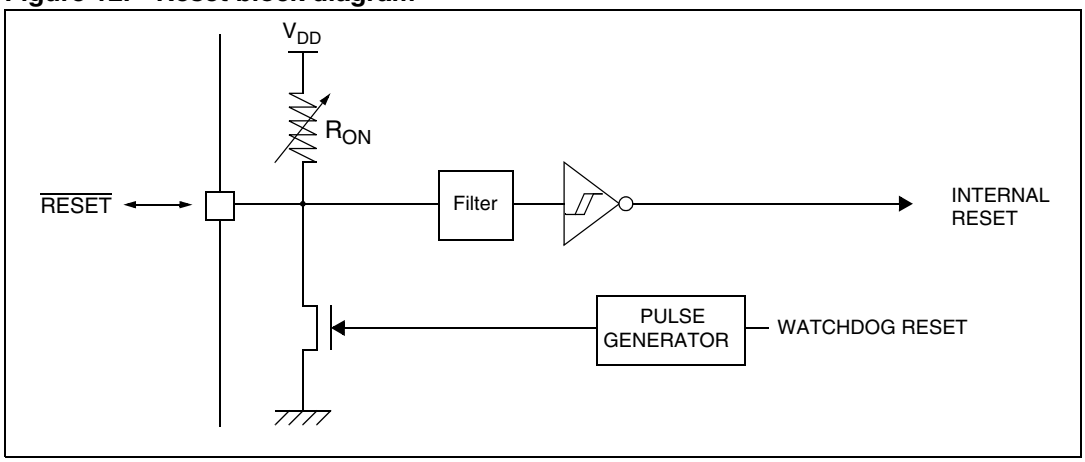
To start up the microcontroller correctly, the user must ensure by means of an external reset circuit that the reset signal is held low until V_{DD} is over the minimum level specified for the selected f_{OSC} frequency.

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

5.2.4 Internal watchdog reset

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

Figure 12. Reset block diagram



6 Interrupts

6.1 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, 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.

6.2 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 5*). The processing flow is shown in *Figure 13*

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).

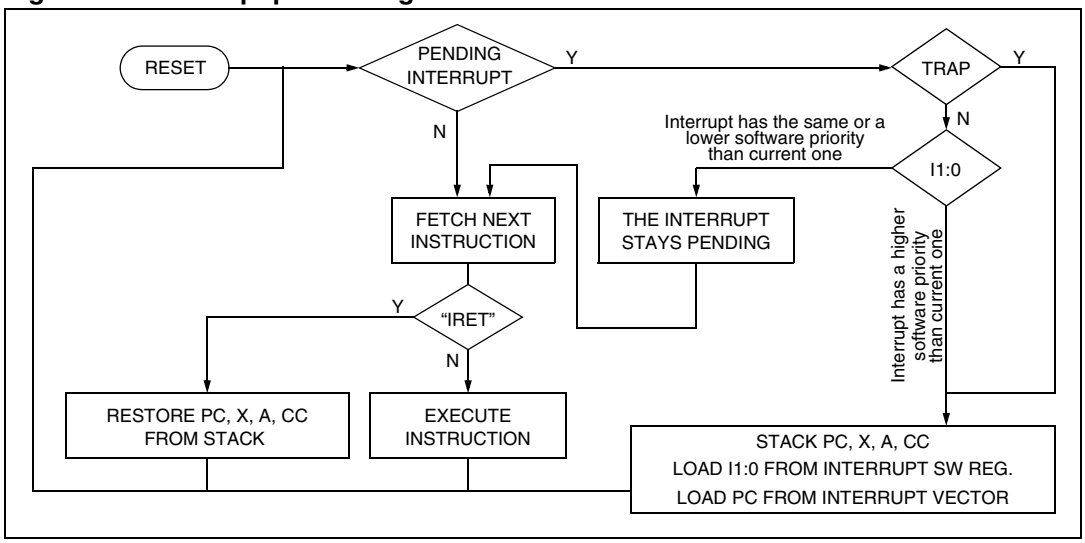
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

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 5. 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 13. Interrupt processing flowchart



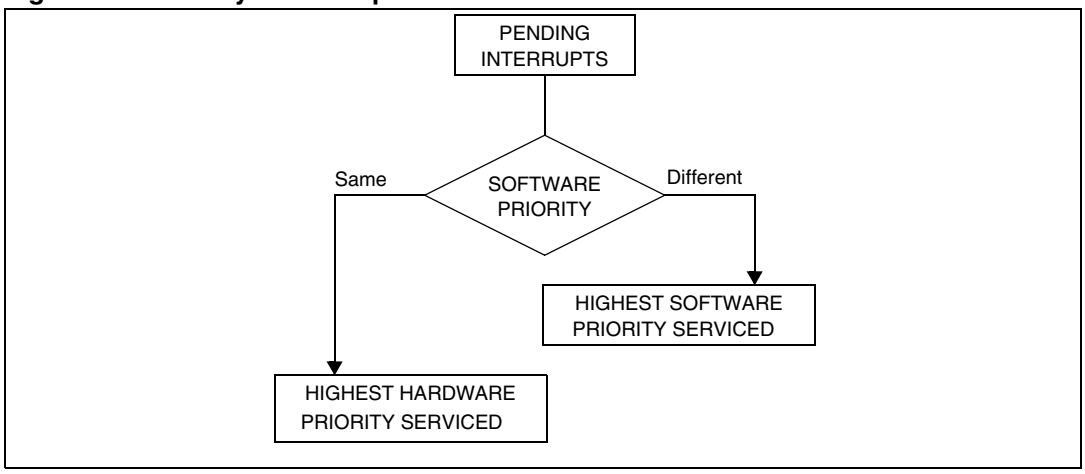
Servicing pending interrupts

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 14 describes this decision process.

Figure 14. 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.

2 RESET and TRAP 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, TRAP) and the maskable type (external or from internal peripherals).

Nonmaskable sources

These sources are processed regardless of the state of the I1 and I0 bits of the CC register (see *Figure 13*). 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 in *Figure 13*.

- 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 *Section 5.2: Reset sequence manager (RSM)* 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 External Interrupt Control register (EICR).
  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 line are selected simultaneously, these will be logically ORed.

- 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.

6.3 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 from 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 14*.

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.

6.4 Concurrent and nested management

The following *Figure 15* and *Figure 16* 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 16*. 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.

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

Figure 15. Concurrent interrupt management

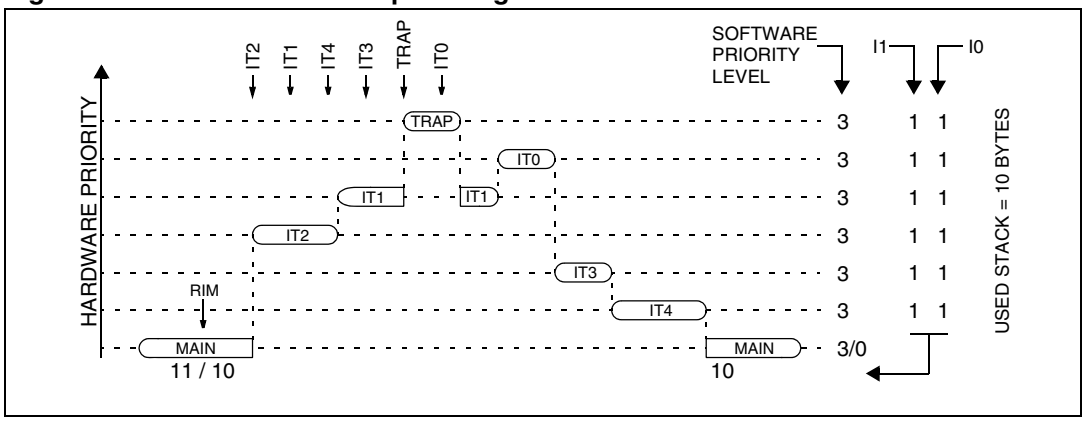
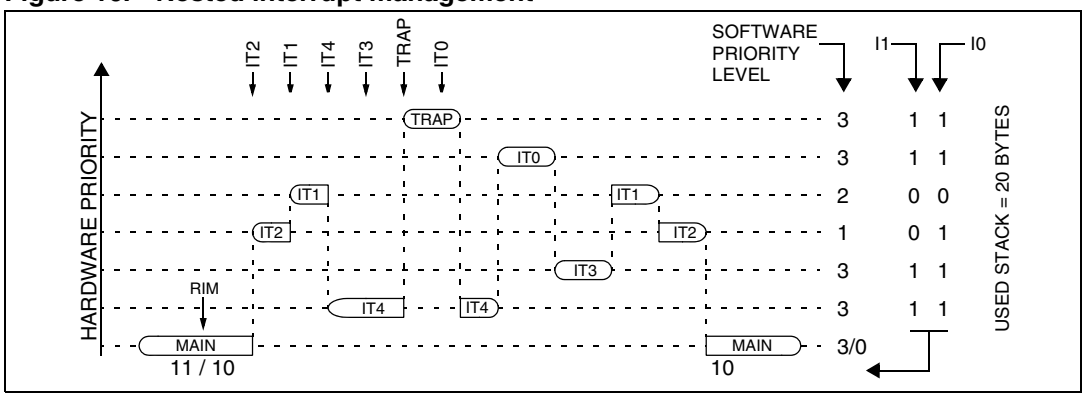


Figure 16. Nested interrupt management



6.5 Interrupt register description

6.5.1 CPU CC register interrupt bits

Reset value: 111x 1010 (xAh)

7							0
1	1	I1	H	I0	N	Z	C
Read only		Read/Write					

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

These two bits indicate the current interrupt software priority as shown in *Table 6*.

Table 6. Current interrupt software priority

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	1

1. TRAP and RESET events can interrupt a level 3 program.

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).

6.5.2 Interrupt software priority registers (ISPRx)

Reset value: 1111 1111 (FFh)

	7							0
ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1	I1_0	I0_0
ISPR1	I1_7	I0_7	I1_6	I0_6	I1_5	I0_5	I1_4	I0_4
ISPR2	I1_11	I0_11	I1_10	I0_10	I1_9	I0_9	I1_8	I0_8
Read/write								
ISPR3	1	1	1	1	I1_13	I0_13	I1_12	I0_12
Read only					Read/write			

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 *Table 7*.

Table 7. ISPRx interrupt vector correspondence

Vector address	ISPRx bits
FFFBh-FFFAh	I1_0 and I0_0 bits*
FFF9h-FFF8h	I1_1 and I0_1 bits
...	...
FFE1h-FFE0h	I1_13 and I0_13 bits

- 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 behavior 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 8. Dedicated interrupt instruction set

Instruction	New description	Function/example	I1	H	I0	N	Z	C
Halt	Entering Halt mode		1		0			
IRET	Interrupt routine return	Pop CC, A, X, PC	I1	H	I0	N	Z	C
JRM	Jump if I1:0=11 (level 3)	I1:0=11 ?						
JRNM	Jump if I1:0<>11	I1:0<>11 ?						

Table 8. Dedicated interrupt instruction set (continued)

Instruction	New description	Function/example	I1	H	I0	N	Z	C
POP CC	Pop CC from the Stack	Mem => CC	I1	H	I0	N	Z	C
RIM	Enable interrupt (level 0 set)	Load 10 in I1:0 of CC	1		0			
SIM	Disable interrupt (level 3 set)	Load 11 in I1:0 of CC	1		1			
TRAP	Software trap	Software NMI	1		1			
WFI	Wait for interrupt		1		0			

Note: During the execution of an interrupt routine, the Halt, POPCC, RIM, SIM and WFI instructions change the current software priority up to the next IRET instruction or one of the previously mentioned instructions.

Table 9. Interrupt mapping

No	Source block	Description	Register label	Priority order	Exit from Halt/Active-Halt	Address vector
	RESET	Reset	N/A		yes	FFFEh-FFFFh
	TRAP	Software interrupt			no	FFFCh-FFFDh
0	Not used					FFFAh-FFFBh
1	MCC/RTC	Main clock controller time base interrupt	MCCSR	Higher Priority	yes	FFF8h-FFF9h
2	ei0	External interrupt port A3..0	N/A		yes	FFF6h-FFF7h
3	ei1	External interrupt port F2..0			yes	FFF4h-FFF5h
4	ei2	External interrupt port B3..0			yes	FFF2h-FFF3h
5	ei3	External interrupt port B7..4			yes	FFF0h-FFF1h
6	Not used					FFEEh-FFEFh
7	SPI	SPI peripheral interrupts	SPICSR		yes	FFECh-FFEDh
8	TIMER A	TIMER A peripheral interrupts	TASR		no	FFEAh-FFEBh
9	TIMER B	TIMER B peripheral interrupts	TBSR		no	FFE8h-FFE9h
10	Not used			Lower Priority		FFE6h-FFE7h
11	Not used					FFE4h-FFE5h

6.6 External interrupts

6.6.1 I/O port interrupt sensitivity

The external interrupt sensitivity is controlled by the IPA, IPB and ISxx bits of the EICR register (*Figure 17*). This control allows to have up to 4 fully independent external interrupt source sensitivities.

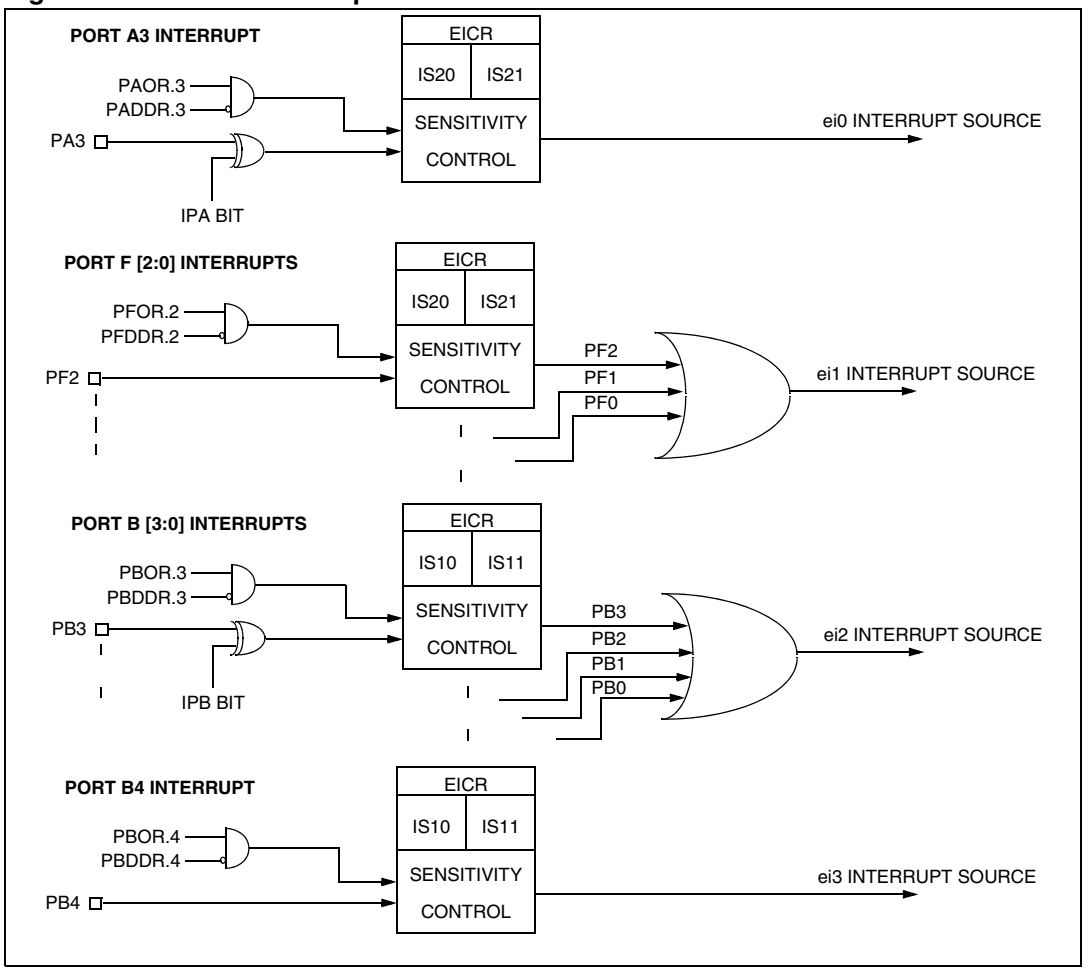
Each external interrupt source can be generated on four (or five) different events on the pin:

- Falling edge
- Rising edge
- Falling and rising edge
- Falling edge and low level
- Rising edge and high level (only for ei0 and ei2)

To guarantee correct functionality, the sensitivity bits in the EICR register can be modified only when the I1 and I0 bits of the CC register are both set to 1 (level 3). This means that interrupts must be disabled before changing sensitivity.

The pending interrupts are cleared by writing a different value in the ISx[1:0], IPA or IPB bits of the EICR.

Figure 17. External interrupt control bits



6.7 External interrupt control register (EICR)

Reset value: 0000 0000 (00h)

7							0
IS11	IS10	IPB	IS21	IS20	IPA	0	0
Read / Write						Read only	

Bit 7:6 = **IS1[1:0]** *ei2 and ei3 sensitivity*

The interrupt sensitivity, defined using the IS1[1:0] bits, is applied to the following external interrupts:

– ei2 (port B3..0) as shown in *Table 10*

– ei3 (port B4) as shown in *Table 11*

Table 10. Interrupt sensitivity - ei2

IS11	IS10	External interrupt sensitivity	
		IPB bit = 0	IPB bit = 1
0	0	Falling edge & low level	Rising edge & high level
0	1	Rising edge only	Falling edge only
1	0	Falling edge only	Rising edge only
1	1	Rising and falling edge	

Table 11. Interrupt sensitivity - ei3

IS11	IS10	External interrupt sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

Bits 7:6 can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

Bit 5 = **IPB** *Interrupt polarity for port B*

This bit is used to invert the sensitivity of the port B [3:0] external interrupts. It can be set and cleared by software only when I1 and I0 of the CC register are both set to 1 (level 3).

0: No sensitivity inversion

1: Sensitivity inversion

Bit 4:3 = **IS2[1:0]** *ei0 and ei1 sensitivity*

The interrupt sensitivity, defined using the IS2[1:0] bits, is applied to the following external interrupts:

– ei0 (port A3..0) as shown in *Table 12*
– ei1 (port F2..0) as shown in *Table 13*

Table 12. Interrupt sensitivity - ei0

IS21	IS20	External interrupt sensitivity	
		IPA bit = 0	IPA bit = 1
0	0	Falling edge & low level	Rising edge & high level
0	1	Rising edge only	Falling edge only
1	0	Falling edge only	Rising edge only
1	1	Rising and falling edge	

Table 13. Interrupt sensitivity - ei1

IS21	IS20	External interrupt sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

Bit 4:3 can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

Bit 2 = **IPA** *Interrupt polarity for port A*

This bit is used to invert the sensitivity of the port A [3:0] external interrupts. It can be set and cleared by software only when I1 and I0 of the CC register are both set to 1 (level 3).
0: No sensitivity inversion
1: Sensitivity inversion

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

Table 14. Nested interrupts register map and reset values

Address (hex.)	Register label	7	6	5	4	3	2	1	0
0024h		ei1		ei0		MCC			
	ISPR0 Reset value	I1_3 1	I0_3 1	I1_2 1	I0_2 1	I1_1 1	I0_1 1	1	1
0025h		SPI				ei3		ei2	
	ISPR1 Reset value	I1_7 1	I0_7 1	I1_6 1	I0_6 1	I1_5 1	I0_5 1	I1_4 1	I0_4 1
0026h						TIMER B		TIMER A	
	ISPR2 Reset value	I1_11 1	I0_11 1	I1_10 1	I0_10 1	I1_9 1	I0_9 1	I1_8 1	I0_8 1
0027h									
	ISPR3 Reset value	1	1	1	1	I1_13 1	I0_13 1	I1_12 1	I0_12 1
0028h	**EICR** Reset value	IS11 0	IS10 0	IPB 0	IS21 0	IS20 0	IPA 0	0	0

7 Power saving modes

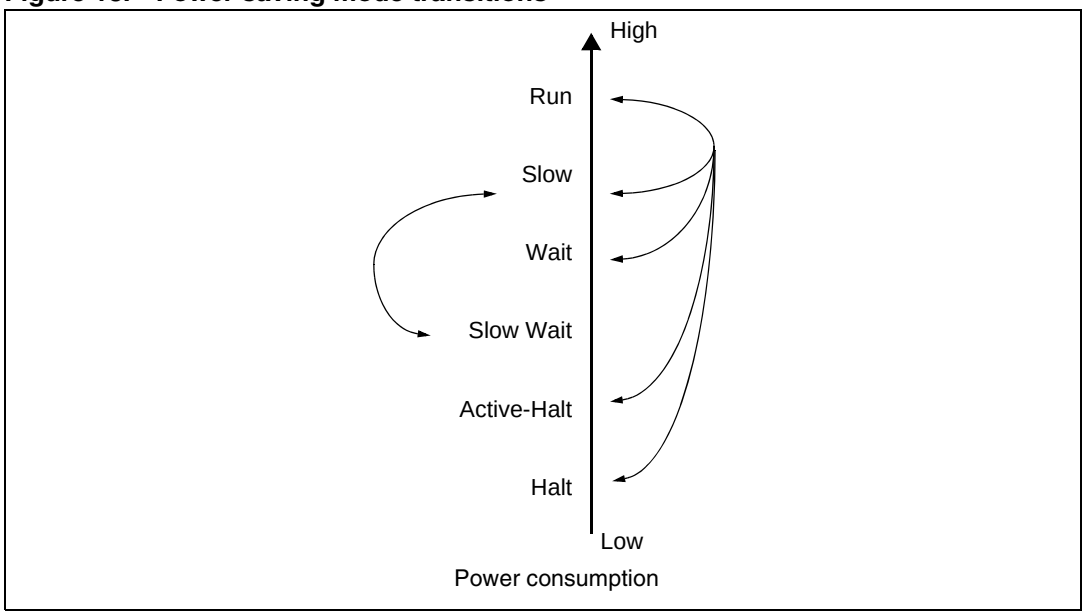
7.1 Introduction

To give a large measure of flexibility to the application in terms of power consumption, four main power saving modes are implemented in the ST7 (see *Figure 18*): Slow, Wait (Slow Wait), Active-Halt and Halt.

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 or multiplied by 2 (f_{OSC2}).

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 18. Power saving mode transitions



7.2 Slow mode

This mode has two targets:

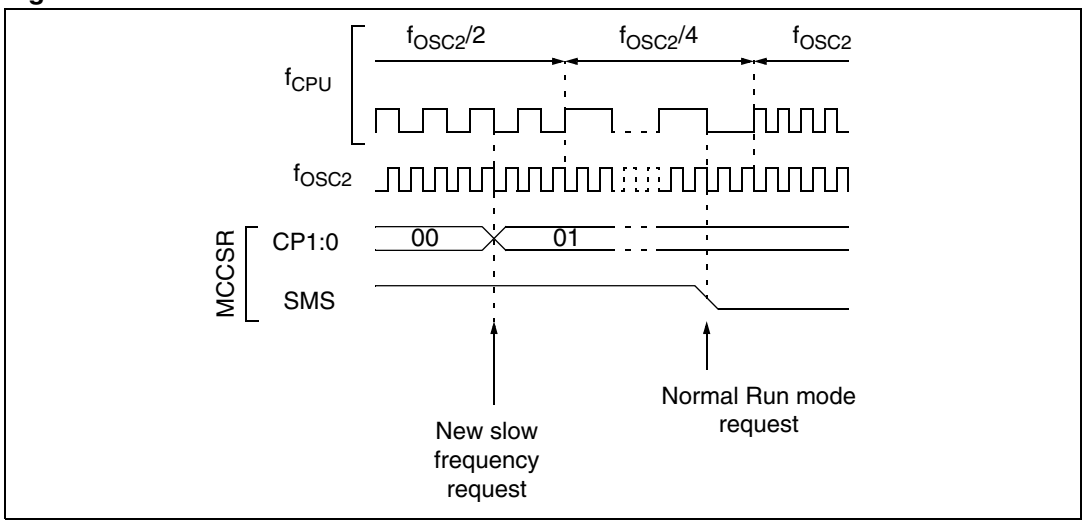
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

Slow mode is controlled by three bits in the MCCSR 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 master clock frequency (f_{OSC2}) can be divided by 2, 4, 8 or 16. The CPU and peripherals are clocked at this lower frequency (f_{CPU}).

Note: Slow-Wait mode is activated when entering the Wait mode while the device is already in Slow mode.

Figure 19. Slow mode clock transitions



7.3 Wait mode

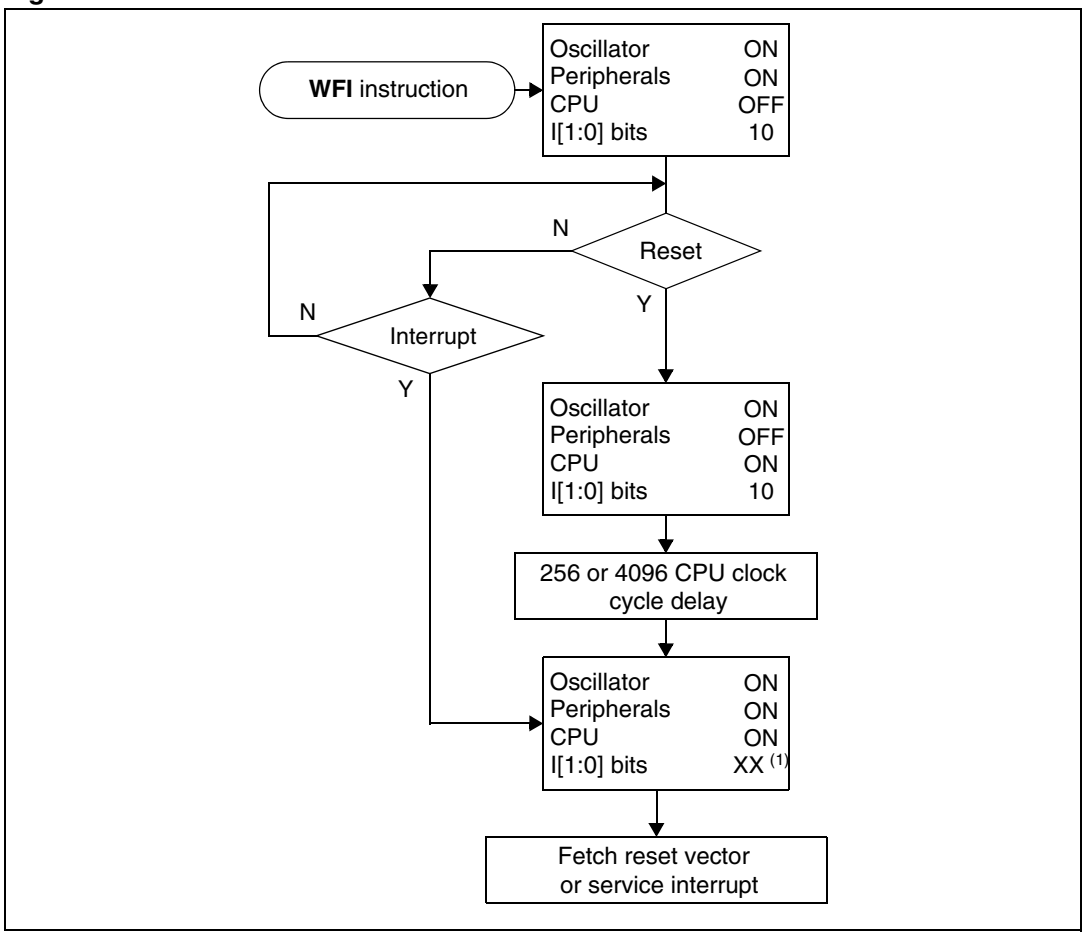
Wait mode places the MCU in a low power consumption mode by stopping the CPU.

This power saving mode is selected by calling the 'WFI' instruction.

All peripherals remain active. During Wait mode, the I[1:0] bits of the CC register are forced to '10', 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 20*.

Figure 20. Wait mode flowchart



Note: 1 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 recovered when the CC register is popped.

7.4 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 MCCSR register).

Table 15. MCC/RTC low-power mode selection

MCCSR OIE bit	Power saving mode entered when Halt instruction is executed
0	Halt mode
1	Active-Halt mode

7.4.1 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 (MCCSR) is set (see *Section 9.2 on page 61* for more details on the MCCSR register).

The MCU can exit Active-Halt mode on reception of either an MCC/RTC interrupt, a specific interrupt (see *Table 9: Interrupt mapping on page 36*) or a Reset. When exiting Active-Halt mode by means of an interrupt, no 256 or 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 22*).

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 (MCCSR.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.

Caution: When exiting Active-Halt mode following an interrupt, OIE bit of MCCSR register must not be cleared before t_{DELAY} after the interrupt occurs (t_{DELAY} = 256 or 4096 t_{CPU} delay depending on option byte). Otherwise, the ST7 enters Halt mode for the remaining t_{DELAY} period.

Figure 21. Active-Halt timing overview

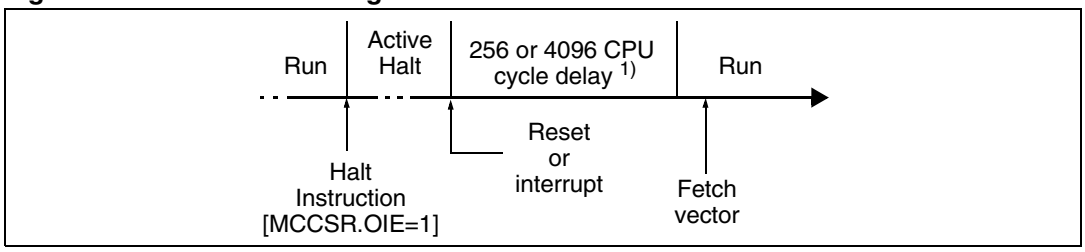
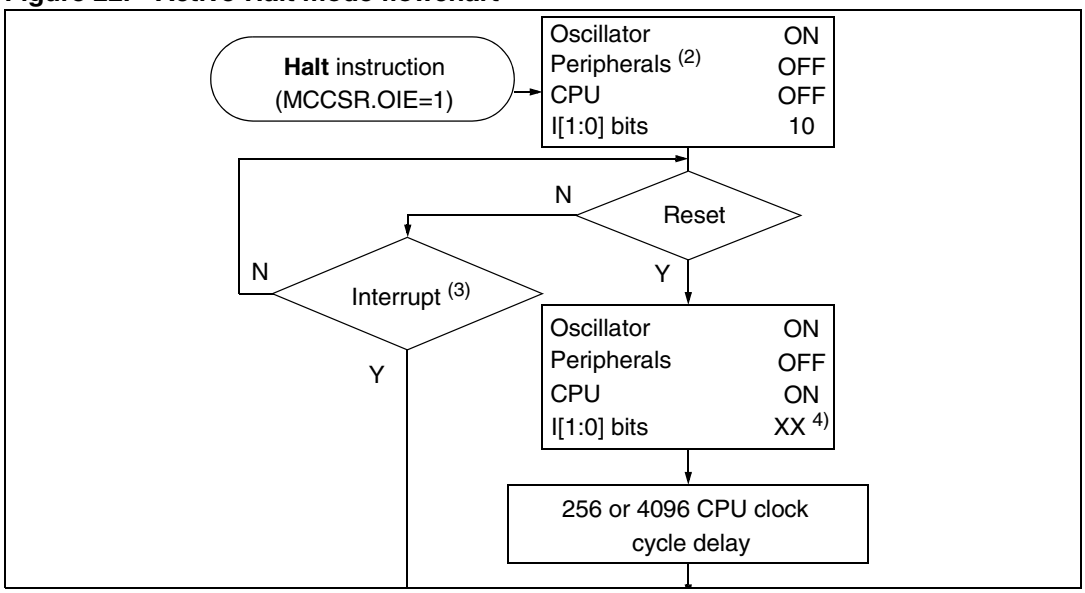


Figure 22. Active-Halt mode flowchart



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 9: Interrupt mapping on page 36* 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.

7.4.2 Halt mode

The Halt mode is the lowest power consumption mode of the MCU. It is entered by executing the 'Halt' instruction when the OIE bit of the Main Clock Controller Status register (MCCSR) is cleared (see *Section 9.2 on page 61* for more details on the MCCSR register).

The MCU can exit Halt mode on reception of either a specific interrupt (see *Table 9: Interrupt mapping on page 36*) or a Reset. When exiting Halt mode by means of a Reset or an interrupt, the oscillator is immediately turned on and the 256 or 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 24*).

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 up immediately.

In 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 13.1 on page 153*) for more details.

Figure 23. Halt timing overview

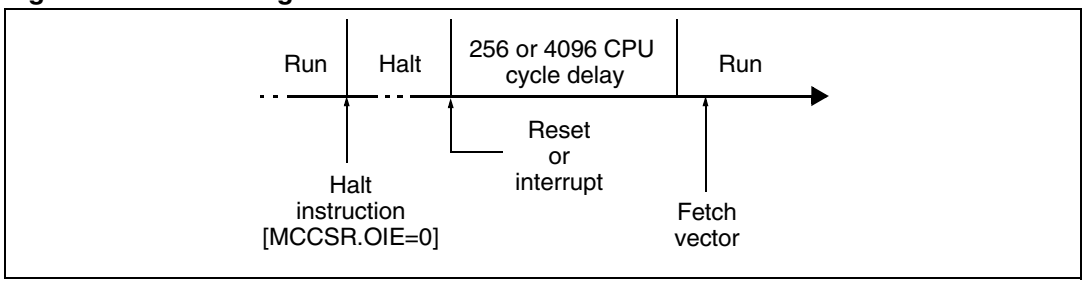
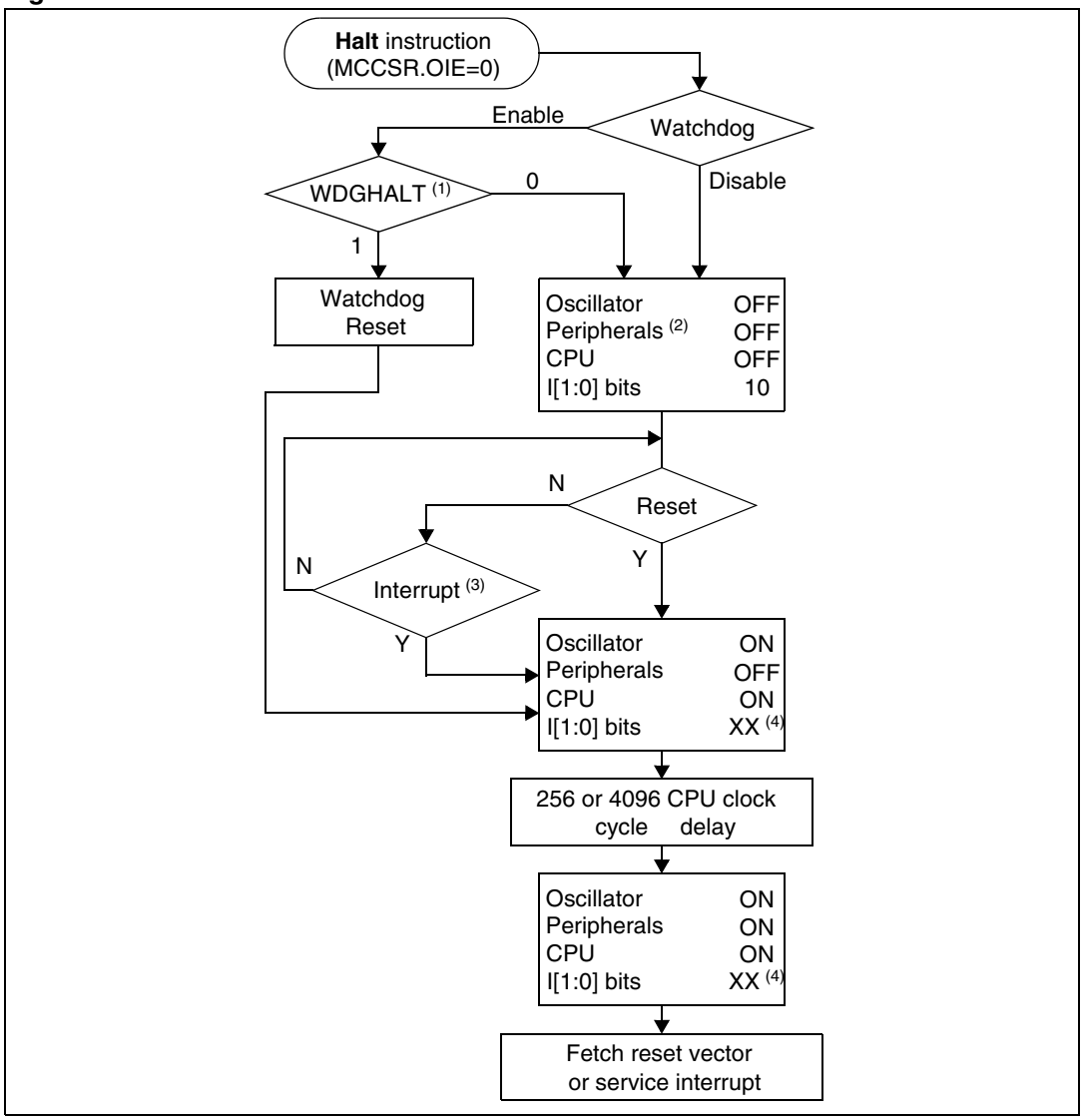


Figure 24. Halt mode flowchart



1. WDGHALT is an option bit. See *Option byte 0* and *Option byte 1* sections 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 9: Interrupt mapping on page 36* 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 recovered when the CC register is popped.

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 sensitiveness 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).

8 I/O ports

8.1 Introduction

The I/O ports offer different functional modes:

- transfer of data through digital inputs and outputs

and for specific pins:

- external interrupt generation
- alternate signal input/output for the on-chip peripherals.

An I/O port contains up to 8 pins. Each pin can be programmed independently as digital input (with or without interrupt generation) or digital output.

8.2 Functional description

Each port has 2 main registers:

- Data Register (DR)
- Data Direction Register (DDR)

and one optional register:

- Option Register (OR)

Each I/O pin may be programmed using the corresponding register bits in the DDR and OR registers: bit X corresponding to pin X of the port. The same correspondence is used for the DR register.

The following description takes into account the OR register, (for specific ports which do not provide this register refer to the I/O Port Implementation section). The generic I/O block diagram is shown in *Figure 25*

8.2.1 Input modes

The input configuration is selected by clearing the corresponding DDR register bit.

In this case, reading the DR register returns the digital value applied to the external I/O pin.

Different input modes can be selected by software through the OR register.

Note: 1 Writing the DR register modifies the latch value but does not affect the pin status.

2 When switching from input to output mode, the DR register has to be written first to drive the correct level on the pin as soon as the port is configured as an output.

3 Do not use read/modify/write instructions (BSET or BRES) to modify the DR register as this might corrupt the DR content for I/Os configured as input.

External interrupt function

When an I/O is configured as Input with Interrupt, an event on this I/O can generate an external interrupt request to the CPU.

Each pin can independently generate an interrupt request. The interrupt sensitivity is independently programmable using the sensitivity bits in the EICR register.

Each external interrupt vector is linked to a dedicated group of I/O port pins (see pinout description and interrupt section). If several input pins are selected simultaneously as interrupt sources, these are first detected according to the sensitivity bits in the EICR register and then logically ORed.

The external interrupts are hardware interrupts, which means that the request latch (not accessible directly by the application) is automatically cleared when the corresponding interrupt vector is fetched. To clear an unwanted pending interrupt by software, the sensitivity bits in the EICR register must be modified.

8.2.2 Output modes

The output configuration is selected by setting the corresponding DDR register bit. In this case, writing the DR register applies this digital value to the I/O pin through the latch. Then reading the DR register returns the previously stored value.

Two different output modes can be selected by software through the OR register: Output push-pull and open-drain as shown in *Table 16*.

Table 16. DR register value and output pin status

DR	Push-pull	Open-drain
0	V_{SS}	V_{SS}
1	V_{DD}	Floating

8.2.3 Alternate functions

When an on-chip peripheral is configured to use a pin, the alternate function is automatically selected. This alternate function takes priority over the standard I/O programming.

When the signal is coming from an on-chip peripheral, the I/O pin is automatically configured in output mode (push-pull or open drain according to the peripheral).

When the signal is going to an on-chip peripheral, the I/O pin must be configured in input mode. In this case, the pin state is also digitally readable by addressing the DR register.

Note: Input pull-up configuration can cause unexpected value at the input of the alternate peripheral input. When an on-chip peripheral use a pin as input and output, this pin has to be configured in input floating mode.

Figure 25. I/O port general block diagram

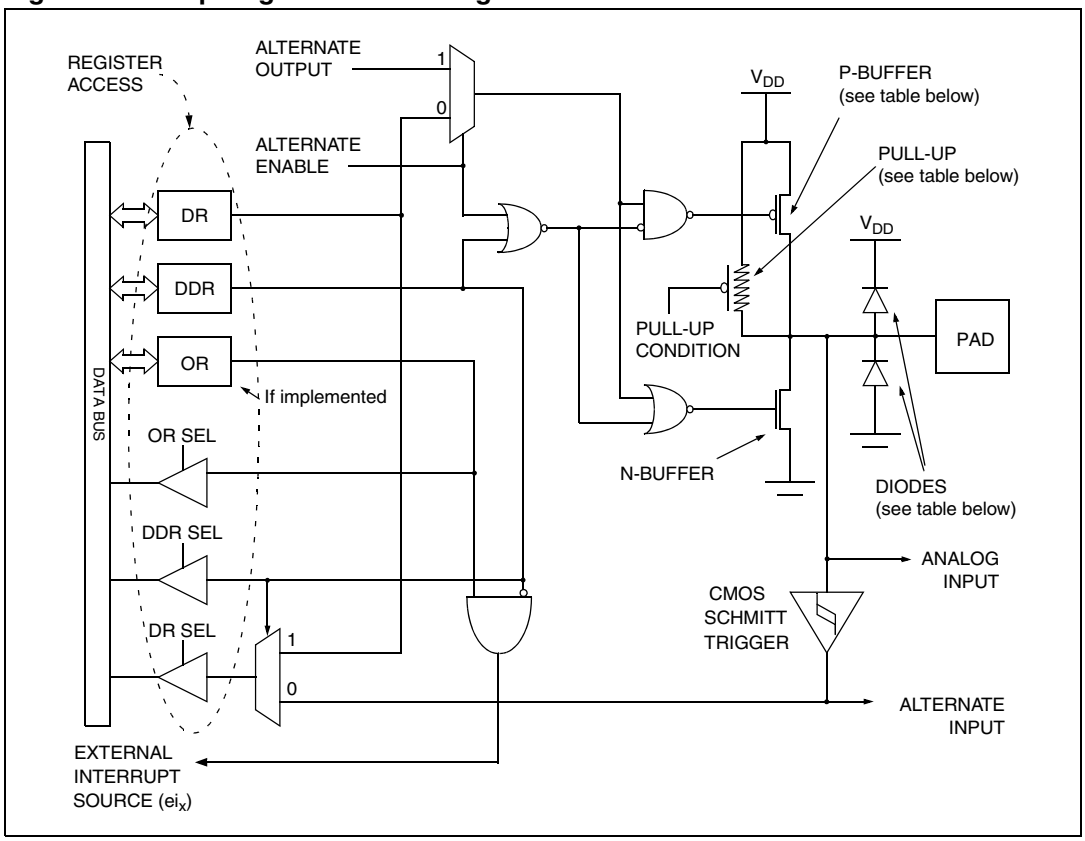


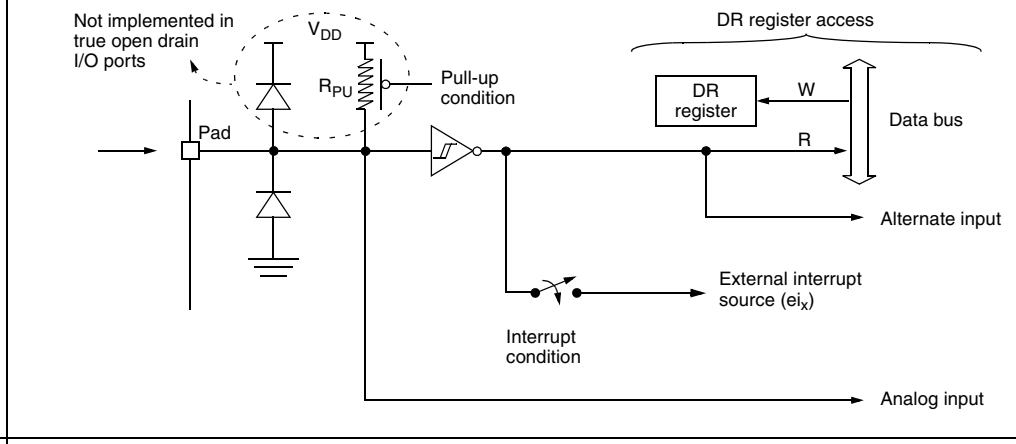
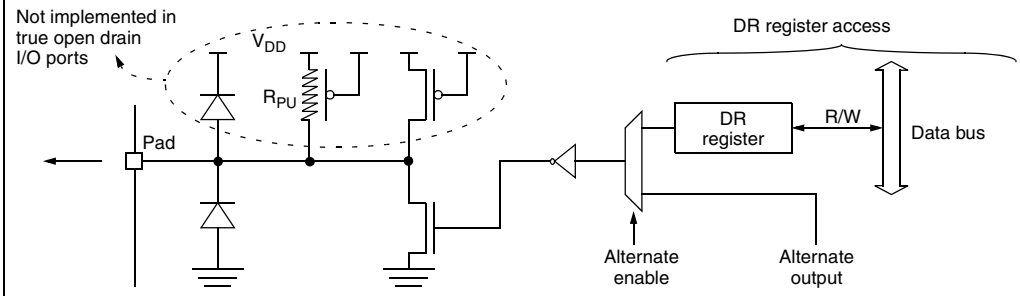
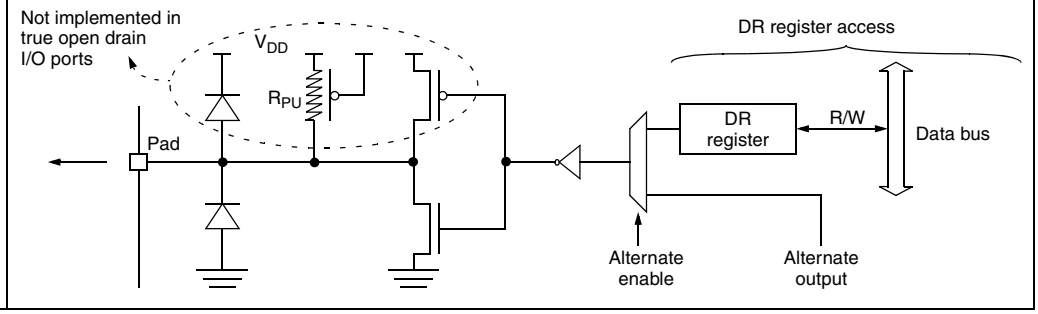
Table 17. I/O port mode options(1)

Configuration mode		Pull-up	P-buffer	Diodes	
				to V_{DD}	to V_{SS}
Input	Floating with/without Interrupt	Off	Off	On	On
	Pull-up with/without Interrupt	On			
Output	Push-pull	Off	On		
	Open Drain (logic level)		Off		
	True Open Drain	NI	NI	NI (2)	

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

2. The diode to V_{DD} is not implemented in the true open drain pads. A local protection between the pad and V_{SS} is implemented to protect the device against positive stress.

Table 18. I/O port configurations

	Hardware configuration
Input (1)	
OPEN-DRAIN OUTPUT (2)	
PUSH-PULL OUTPUT (2)	

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 function reads the pin status given by the DR register content.

Caution: The alternate function must not be activated as long as the pin is configured as input with interrupt, in order to avoid generating spurious interrupts.

Analog alternate function

When the pin is used as an ADC input, the I/O must be configured as floating input. The analog multiplexer (controlled by the ADC registers) switches the analog voltage present on the selected pin to the common analog rail which is connected to the ADC input.

It is recommended not to change the voltage level or loading on any port pin while conversion is in progress. Furthermore it is recommended not to have clocking pins located close to a selected analog pin.

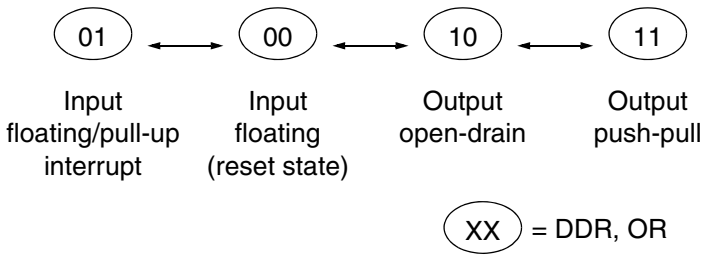
Caution: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

8.3 I/O port implementation

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific feature of the I/O port such as ADC Input or true 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 26* Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 26. Interrupt I/O port state transitions



8.4 Low-power modes

Table 19. Effect of low-power modes on I/O ports

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.

8.5 Interrupts

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and OR registers and the interrupt mask in the CC register is not active (RIM instruction).

Table 20. I/O port interrupt control/wake-up capability

Interrupt event	Event flag	Enable control bit	Exit from Wait	Exit from Halt
External interrupt on selected external event	-	DDRx ORx	Yes	Yes

8.5.1 I/O port Implementation

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

Table 21. PA5:4, PC7:0, PD5:0, PE1:0, PF7:6, 4 standard ports

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

Table 22. PB4, PB2:0, PF1:0 interrupt ports (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

Table 23. PA3, PB3, PF2 interrupt ports (without pull-up)

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

Table 24. PA7:6 true open-drain ports

Mode	DDR
floating input	0
open drain (high sink ports)	1

Table 25. Port configuration

Port	Pin name	Input		Output	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7:6	floating		true open-drain	
	PA5:4	floating	pull-up	open drain	push-pull
	PA3	floating	floating interrupt	open drain	push-pull
Port B	PB3	floating	floating interrupt	open drain	push-pull
	PB4, PB2:0	floating	pull-up interrupt	open drain	push-pull
Port C	PC7:0	floating	pull-up	open drain	push-pull
Port D	PD5:0	floating	pull-up	open drain	push-pull

Table 25. Port configuration (continued)

Port	Pin name	Input		Output	
		OR = 0	OR = 1	OR = 0	OR = 1
Port E	PE1:0	floating	pull-up	open drain	push-pull
Port F	PF7:6, 4	floating	pull-up	open drain	push-pull
	PF2	floating	floating interrupt	open drain	push-pull
	PF1:0	floating	pull-up interrupt	open drain	push-pull

Table 26. 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	PADR	MSB							LSB
0001h	PADDR								
0002h	PAOR								
0003h	PBDR	MSB							LSB
0004h	PBDDR								
0005h	PBOR								
0006h	PCDR	MSB							LSB
0007h	PCDDR								
0008h	PCOR								
0009h	PDDR	MSB							LSB
000Ah	PDDDR								
000Bh	PDOR								
000Ch	PEDR	MSB							LSB
000Dh	PEDDR								
000Eh	PEOR								
000Fh	PFDR	MSB							LSB
0010h	PFDDR								
0011h	PFOR								

9 On-chip peripherals

9.1 Watchdog timer (WDG)

9.1.1 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.

9.1.2 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

9.1.3 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 $t_{w(RSTL)out}$ (see *Table 80: Asynchronous RESET pin characteristics on page 136* for a value of $t_{w(RSTL)out}$).

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:

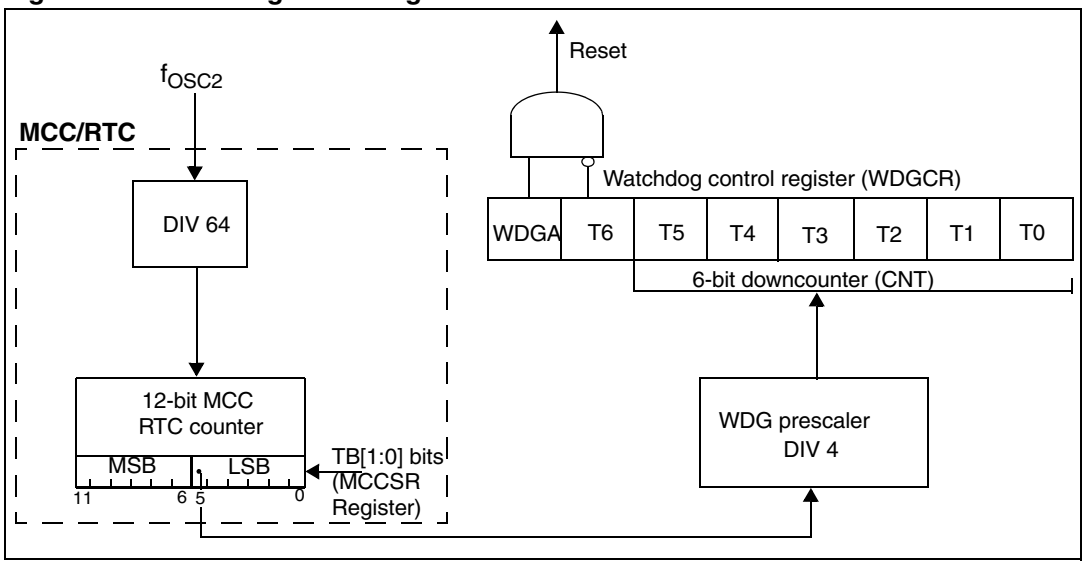
- The WDGA bit is set (watchdog enabled)
- The T6 bit is set to prevent generating an immediate reset
- The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset (see *Figure 28: Approximate timeout duration*). The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WDGCR register (see *Figure*).

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 27. Watchdog block diagram

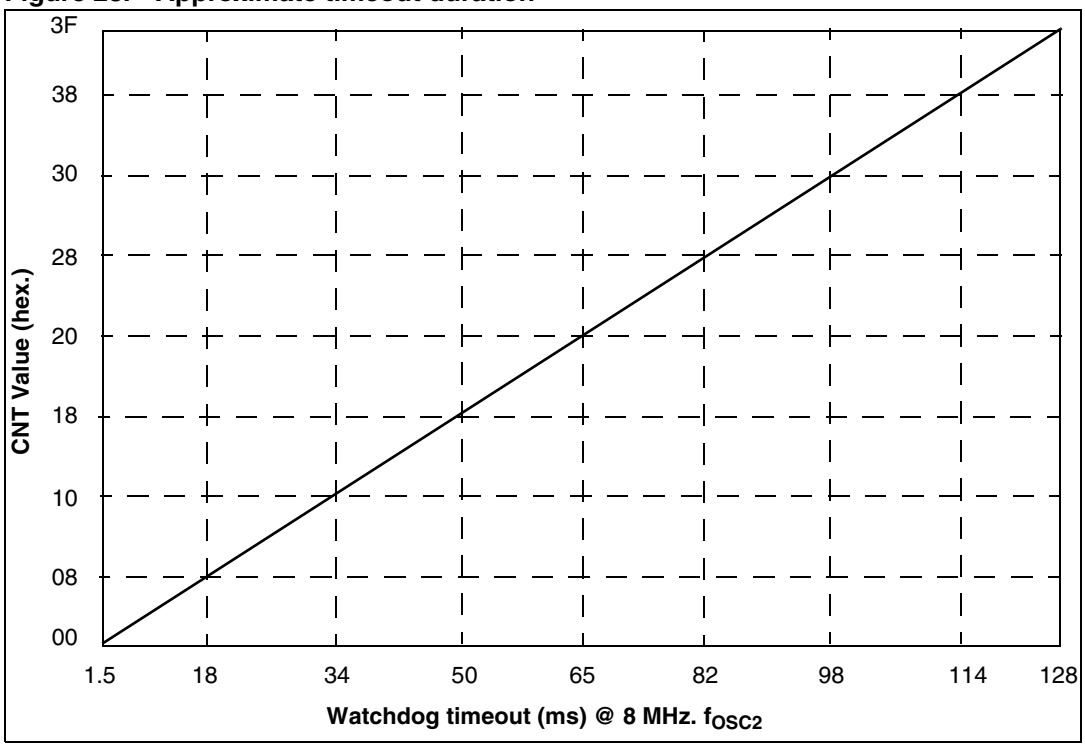


9.1.4 How to program the watchdog timeout

Figure 28 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 *Exact timeout duration (tmin and tmax) on page 58*.

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

Figure 28. Approximate timeout duration



Exact timeout duration (t_{min} and t_{max})

Where:

t_{min0} = (LSB + 128) x 64 x t_{OSC2}

t_{max0} = 16384 x t_{OSC2}

t_{OSC2} = 125ns if f_{OSC2}=8 MHz

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

MSB and LSB are values shown in *Table 27*, that depend on the time base selected by the TB[1:0] bits in the MCCSR register.

Table 27. Time base selection

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

To calculate the minimum Watchdog timeout (t_{min}):

If $CNT < \left[\frac{MSB}{4}\right]$ then $t_{min} = t_{min0} + 16384 \times CNT \times t_{osc2}$

else $t_{min0} + \left[16384 \times \left(CNT - \left[\frac{4CNT}{MSB}\right]\right) + (192 + LSB) \times 64 \times \left[\frac{4CNT}{MSB}\right]\right]$

To calculate the maximum Watchdog timeout (t_{max}):

If $CNT \leq \left[\frac{MSB}{4}\right]$ then $t_{max} = t_{max0} + 16384 \times CNT \times t_{osc2}$

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

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

Example: with 2 ms timeout selected in the MCCSR register:

Table 28. t_{min} and t_{max} values for a time base of 2 ms

Value of T[5:0] bits in WDGCR register (hex.)	Min. watchdog timeout (ms) t_{min}	Max. watchdog timeout (ms) t_{max}
00	1.496	2.048
3F	128	128.552

9.1.5 Low- power modes

The low-power modes are shown in *Table 29*.

Table 29. Low-power modes

Mode	Description		
Slow	No effect on Watchdog.		
Wait	No effect on Watchdog.		
Halt	**OIE bit in MCCSR 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 9.1.7* 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.

9.1.6 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.

9.1.7 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.

9.1.8 Interrupts

None.

9.1.9 Register description

Control register (WDGCR)

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0
Read/Write							

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 30. Watchdog timer register map and reset values

Address (hex.)	Register label	7	6	5	4	3	2	1	0
002Ah	WDGCR reset value	WDGA 0	T6 1	T5 1	T4 1	T3 1	T2 1	T1 1	T0 1

9.2 Main clock controller with real-time clock and beeper (MCC/RTC)

The main clock controller consists of three different functions:

- a programmable CPU clock prescaler
- a clock-out signal to supply external devices
- a real-time clock timer with interrupt capability

Each function can be used independently and simultaneously.

9.2.1 Programmable CPU clock prescaler

The programmable CPU clock prescaler supplies the clock for the ST7 CPU and its internal peripherals. It manages Slow power saving mode (See *Section 7.2: Slow mode* for more details).

The prescaler selects the f_{CPU} main clock frequency and is controlled by three bits in the MCCSR register: CP[1:0] and SMS.

9.2.2 Clock-out capability

The clock-out capability is an alternate function of an I/O port pin that outputs the f_{CPU} clock to drive external devices. It is controlled by the MCO bit in the MCCSR register.

Caution: When selected, the clock out pin suspends the clock during Active-Halt mode.

9.2.3 Real-time clock timer (RTC)

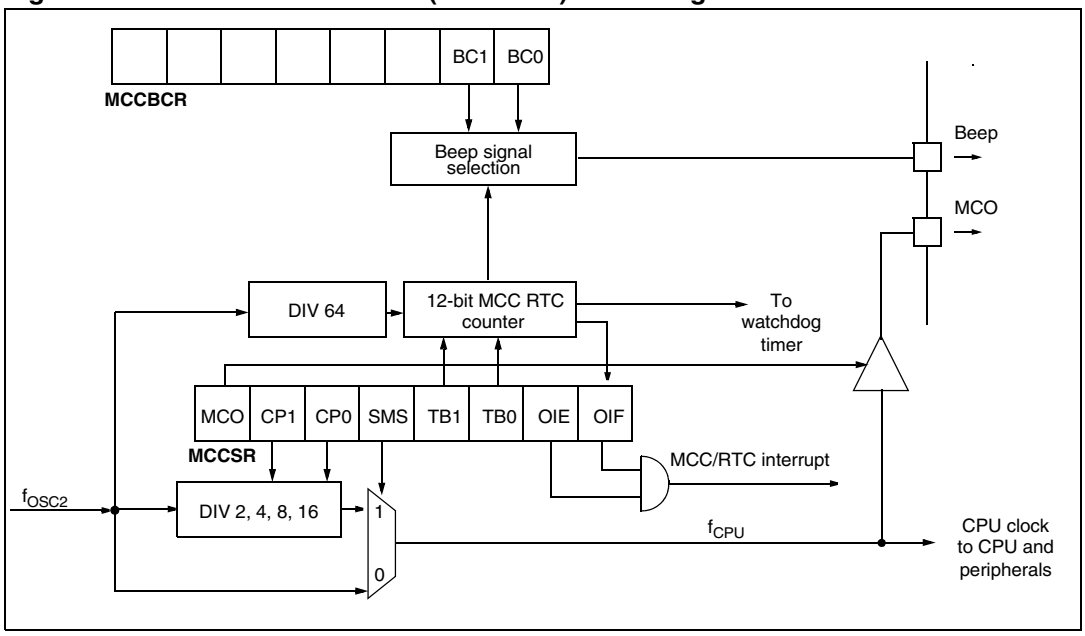
The counter of the real-time clock timer allows an interrupt to be generated based on an accurate real-time clock. Four different time bases depending directly on f_{OSC2} are available. The whole functionality is controlled by four bits of the MCCSR register: TB[1:0], OIE and OIF.

When the RTC interrupt is enabled (OIE bit set), the ST7 enters Active-Halt mode when the Halt instruction is executed. See *Section 7.4: Active-Halt and Halt modes* for more details.

9.2.4 Beeper

The beep function is controlled by the MCCBCR register. It can output three selectable frequencies on the BEEP pin (I/O port alternate function).

Figure 29. Main clock controller (MCC/RTC) block diagram



9.2.5 Low-power modes

Table 31. 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.

9.2.6 Interrupts

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

Table 32. MCC/RTC interrupt control/wake-up capability

Interrupt event	Event flag	Enable Control bit	Exit from Wait	Exit from Halt
Time base overflow event	OIF	OIE	Yes	No(1)

1. The MCC/RTC interrupt wakes up the MCU from Active-Halt mode, not from Halt mode.

9.2.7 Register description

MCC control/status register (MCCSR)

Reset Value: 0000 0000 (00h)

7							0
MCO	CP1	CP0	SMS	TB1	TB0	OIE	OIF
Read/Write							

Bit 7 = **MCO** *Main clock out selection*

This bit enables the MCO alternate function on the PF0 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)

Note: To reduce power consumption, the MCO function is not active in Active-Halt mode.

Bit 6:5 = **CP[1:0]** *CPU clock prescaler*

These bits select the CPU clock prescaler which is applied in the different slow modes. Their action is conditioned by the setting of the SMS bit. These two bits are set and cleared by software. *Table 33* shows how to select the CPU clock prescaler.

Table 33. CPU clock prescaler selection

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

Bit 4 = **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 *Section 7.2: Slow mode* and *Section 9.2: Main clock controller with real-time clock and beeper (MCC/RTC)* for more details.

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

These bits select the programmable divider time base. They are set and cleared by software. *Table 34* shows how to select the programmable divider time base.

Table 34. Time base selection

Counter prescaler	Time base		TB1	TB0
	f_{OSC2} = 4 MHz	f_{OSC2} = 8 MHz		
16000	4 ms	2 ms	0	0
32000	8 ms	4 ms	0	1
80000	20 ms	10 ms	1	0
200000	50 ms	25 ms	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.

Bit 0 = **OIF** *Oscillator interrupt flag*

This bit is set by hardware and cleared by software reading the MCCSR 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 MCCSR register to avoid unintentionally clearing the OIF bit.

MCC beep control register (MCCBCR)

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	BC1	BC0
Read/Write							

Bit 7:2 = Reserved, must be kept cleared.

Bit 1:0 = **BC[1:0]** *Beep control*

These 2 bits select the PF1 pin beep capability.

Table 35. Selecting the PF1 pin peep capability

BC1	BC0	Beep mode with f_{OSC2} = 8 MHz
0	0	Off

Table 35. Selecting the PF1 pin peep capability (continued)

BC1	BC0	Beep mode with f_{OSC2} = 8 MHz	
0	1	~2-kHz	Output beep signal ~50% duty cycle
1	0	~1-kHz	
1	1	~500-Hz	

The beep output signal is available in Active-Halt mode but has to be disabled to reduce the consumption.

Table 36. Main clock controller register map and reset values

Address (hex.)	Register label	7	6	5	4	3	2	1	0
002Ch	MCCSR reset value	MCO 0	CP1 0	CP0 0	SMS 0	TB1 0	TB0 0	OIE 0	OIF 0
002Dh	**MCCBCR** reset value	0	0	0	0	0	0	BC1 0	BC0 0

9.3 16-bit timer

9.3.1 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).

9.3.2 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 30*.

Note: Some timer pins may not be available (not bonded) in some ST7 devices. Refer to the device pinout description.
When reading an input signal on a non-bonded pin, the value will always be '1'.

9.3.3 Functional description

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)

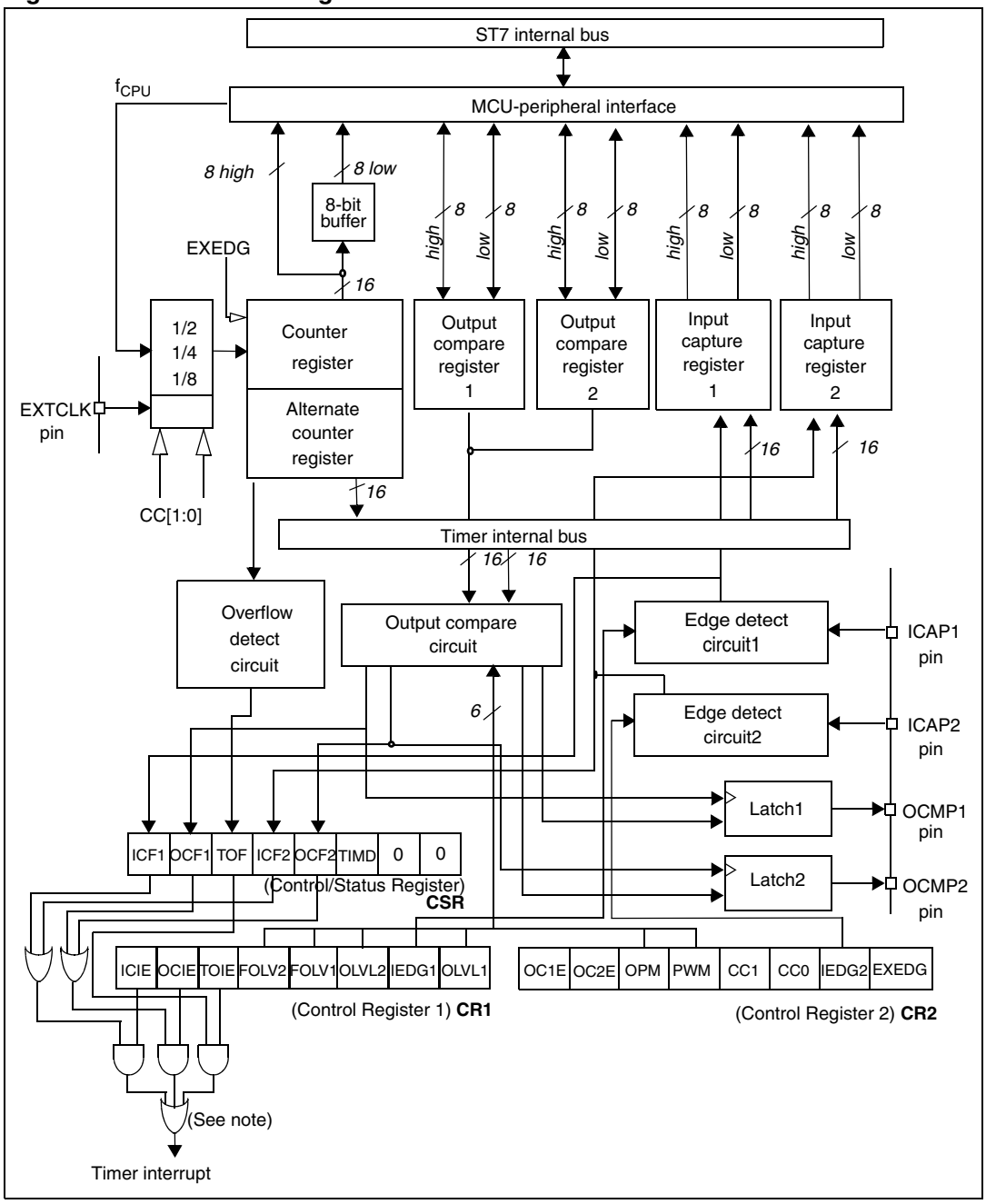
- Alternate Counter High Register (ACHR) is the most significant byte (MS Byte).
- Alternate Counter Low Register (ACLR) is the least significant byte (LS Byte).

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.

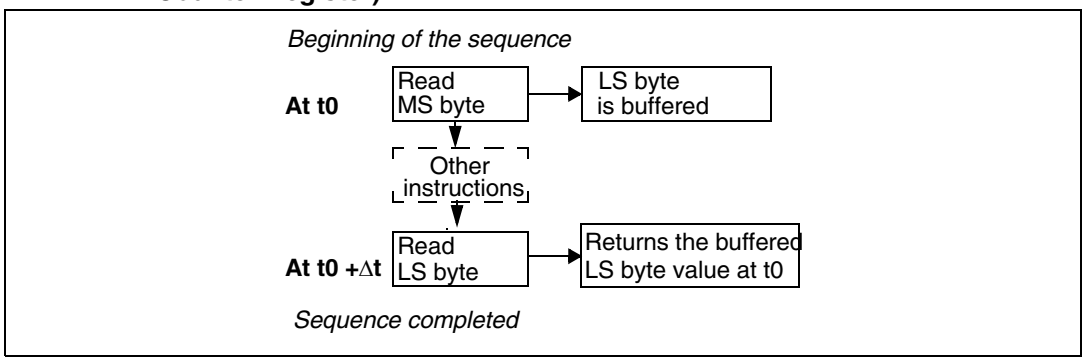
The timer clock depends on the clock control bits of the CR2 register, as illustrated in *Table 42*. The value in the counter register repeats every 131072, 262144 or 524288 CPU clock cycles depending on the CC[1:0] bits.
The timer frequency can be $f_{CPU}/2$, $f_{CPU}/4$, $f_{CPU}/8$ or an external frequency.

Figure 30. Timer block diagram



1. If IC, OC and TO interrupt requests have separate vectors then the last OR is not present (see device interrupt vector table).

Figure 31. 16-bit read sequence (from either the Counter Register or the Alternate Counter Register).



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.

Note: 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).

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.

Figure 32. Counter timing diagram, internal clock divided by 2

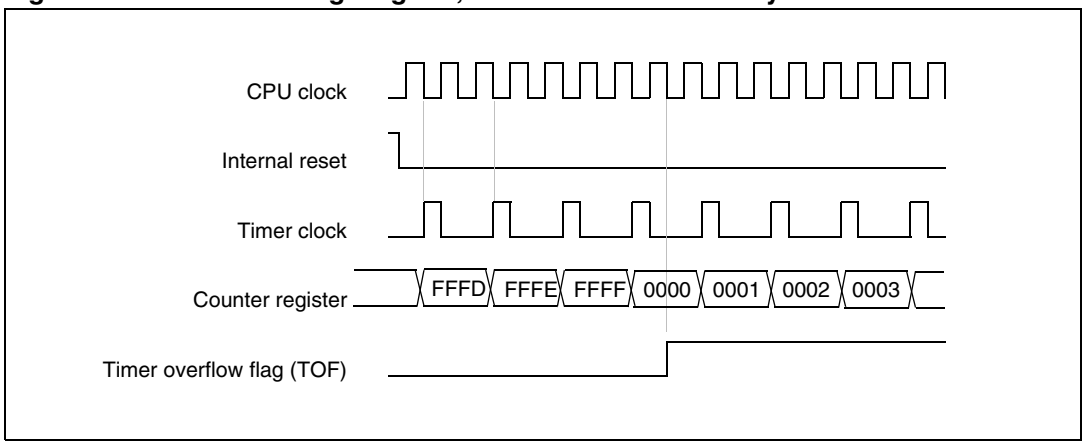


Figure 33. Counter timing diagram, internal clock divided by 4

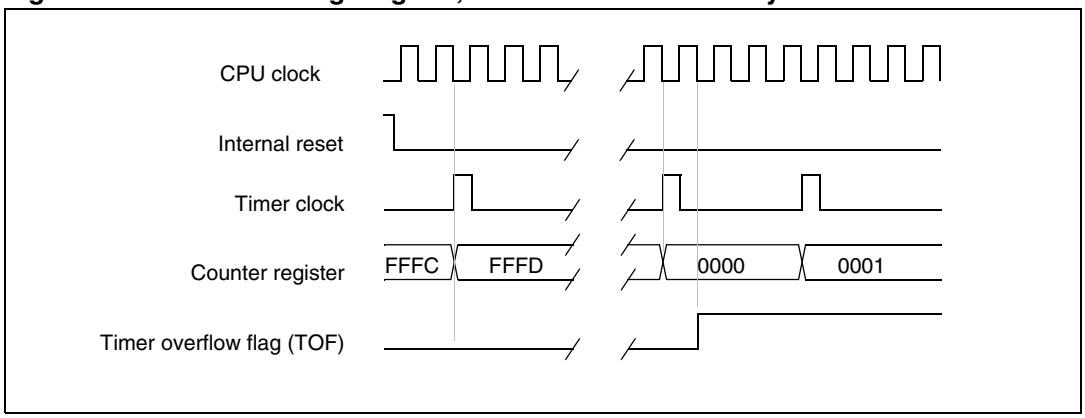
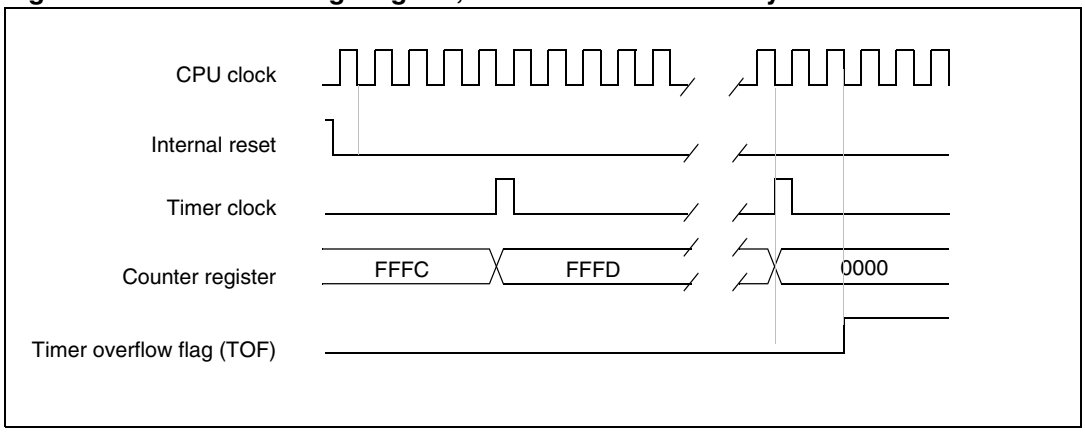


Figure 34. Counter timing diagram, internal clock divided by 8



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

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 35*).

Table 37. Input capture byte distribution

IC1R/IC2R	MS byte	LS byte
ICiR	IC*i*HR	IC*i*LR

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

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

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 42*).
- 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 ICAP1pin 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 36*).
- 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.

Note: 1 After reading the ICiHR register, transfer of input capture data is inhibited and ICFi will never be set until the ICiLR register is also read.

2 The ICiR 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 ICAPi 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 ICiHR (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).

Figure 35. Input capture block diagram

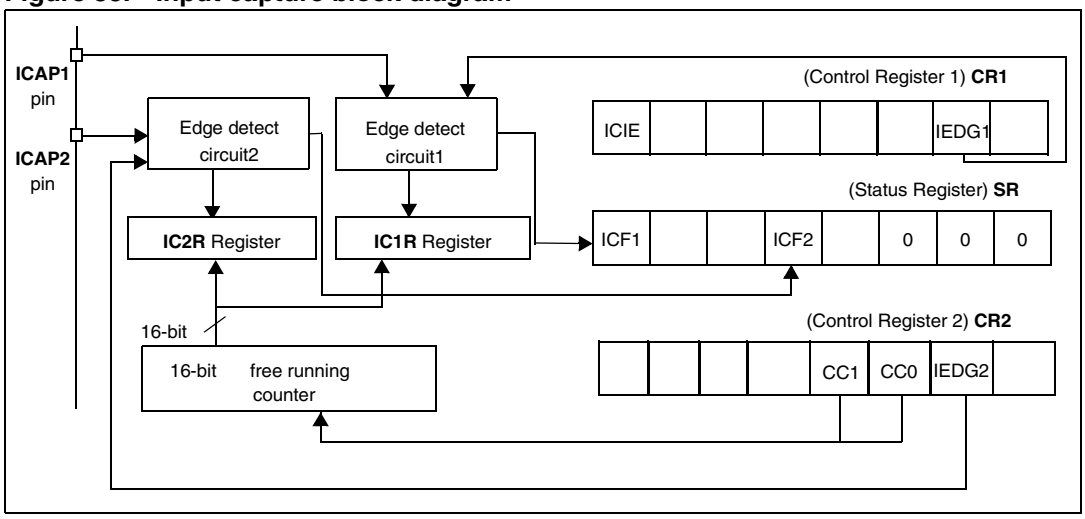
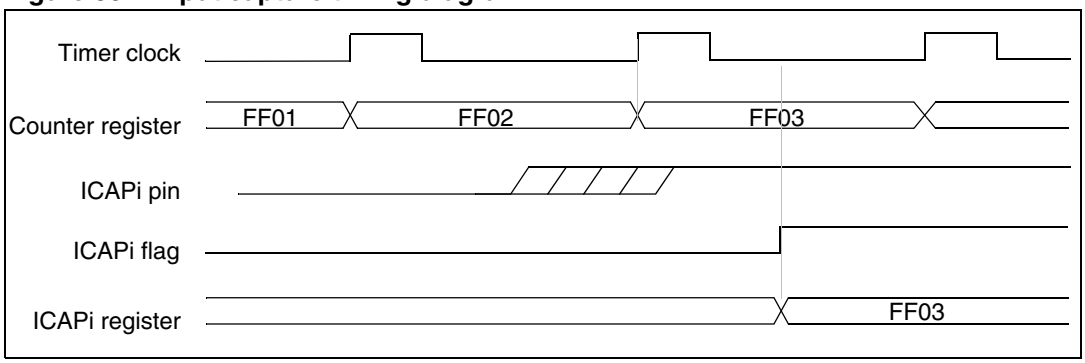


Figure 36. Input capture timing diagram



1. The rising edge is the active edge.

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*i*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.

Table 38. Output compare byte distribution

OC1R/OC2R	MS byte	LS byte
OC*i*R	OC*i*HR	OC*i*LR

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OC*i*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*i*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 42*).

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 OCRi 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*i*R register value required for a specific timing application can be calculated using the following formula:

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

Where:

- Δt = 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 42*)

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

$$\Delta OCiR = \Delta t * f_{EXT}$$

Where:

- Δt = 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*i*LR 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*i*R register:

- Write to the OC*i*HR 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*i*LR register (enables the output compare function and clears the OCF*i* bit).

Note: 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 In both internal and external clock modes, OCFi and OCMPi are set while the counter value equals the OCiR register value (see Figure 38 on page 75 for an example with $f_{CPU}/2$ and Figure 39 on page 75 for an example with $f_{CPU}/4$). This behavior is the same in OPM or PWM mode.

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 FOLV*i* bit is set by software, the OLVL*i* bit is copied to the OCMP*i* pin. The OLV*i* bit has to be toggled in order to toggle the OCMP*i* pin when it is enabled (OC*i*E bit=1). The OCF*i* bit is then not set by hardware, and thus no interrupt request is generated.

The FOLVL*i* bits have no effect in both one pulse mode and PWM mode.

Figure 37. Output compare block diagram

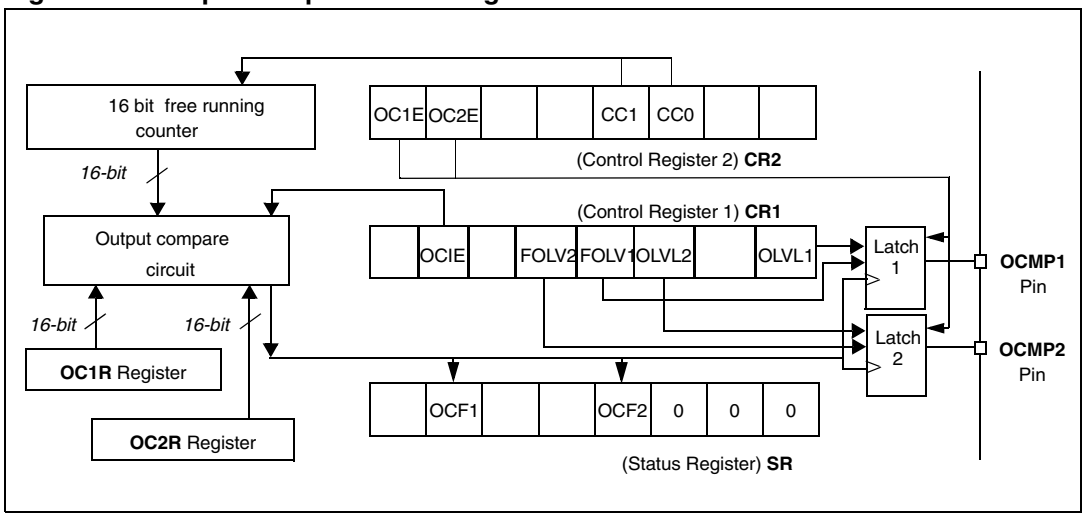


Figure 38. Output compare timing diagram, $f_{TIMER} = f_{CPU}/2$

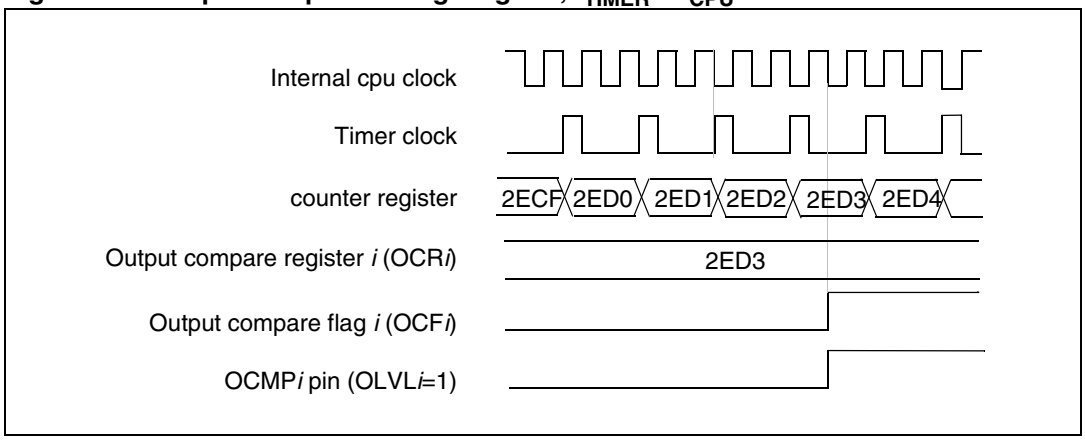
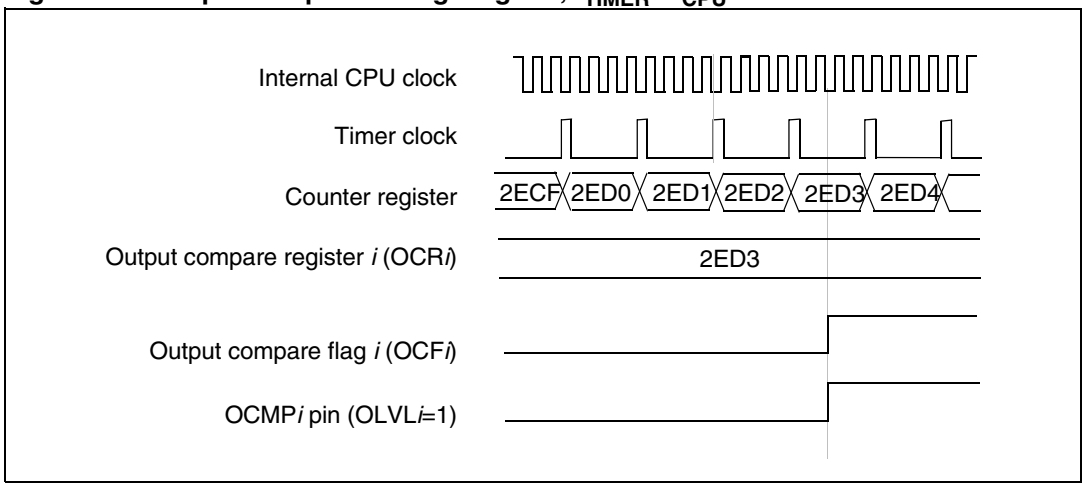


Figure 39. Output compare timing diagram, $f_{TIMER} = f_{CPU}/4$



One-pulse mode

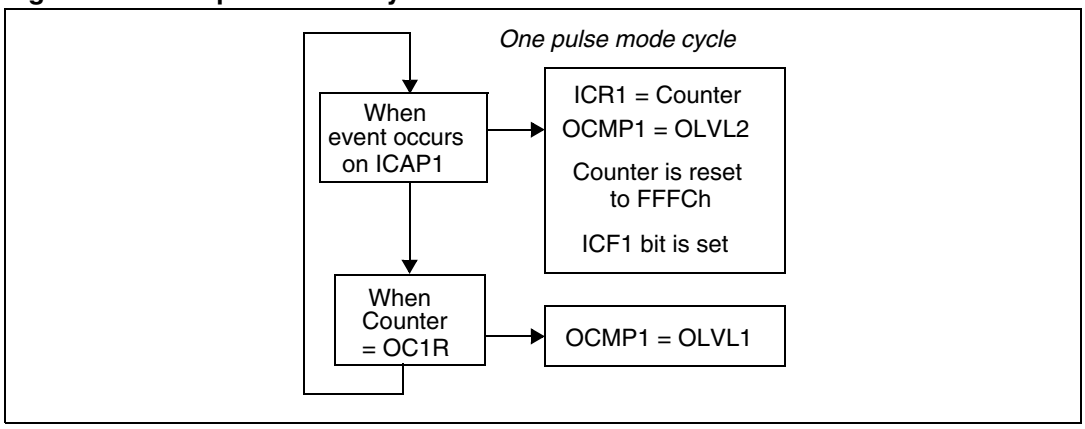
One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

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 42*).

Figure 40. One-pulse mode cycle



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 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.

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

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

Where:

- t = Pulse period (in seconds)
- f_{CPU} = CPU clock frequency (in hertz)
- $_{PRESC}$ = Timer prescaler factor (2, 4 or 8 depending on the CC[1:0] bits, see *Table 42*)

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

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

Where:

- t = Pulse period (in seconds)
- f_{EXT} = 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 41*).

Note: 1 The OCF1 bit cannot be set by hardware in one pulse mode but the OCF2 bit can generate an Output Compare interrupt.

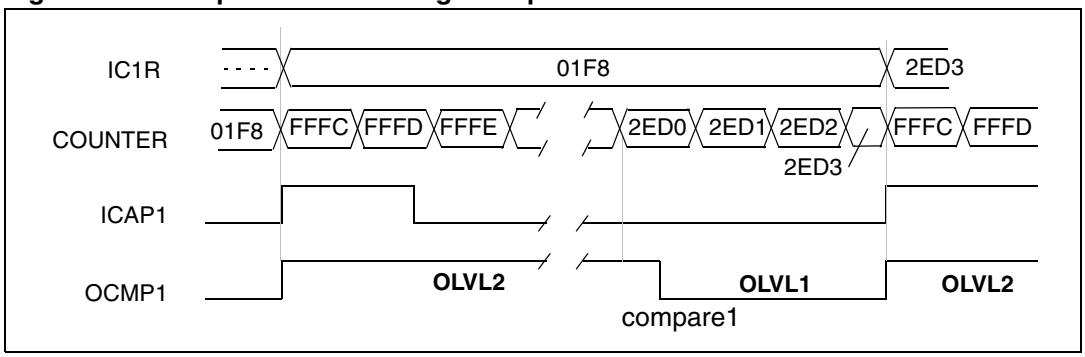
2 When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.

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.

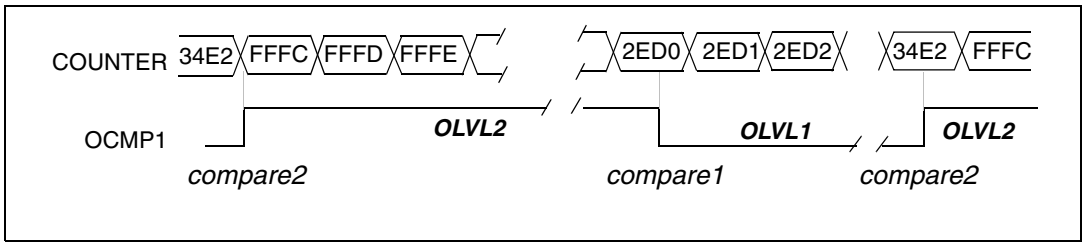
5 When one pulse mode is used OC1R is dedicated to this mode. Nevertheless OC2R and OCF2 can be used to indicate a period of time has been elapsed but cannot generate an output waveform because the level OLVL2 is dedicated to the one pulse mode.

Figure 41. One-pulse mode timing example



1. IEDG1 = 1, OC1R = 2ED0h, OLVL1 = 0, OLVL2 = 1

Figure 42. Pulse width modulation mode timing example with 2 output compare Functions



1. OC1R = 2ED0h, OC2R = 34E2, OLVL1 = 0, OLVL2 = 1

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.

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.

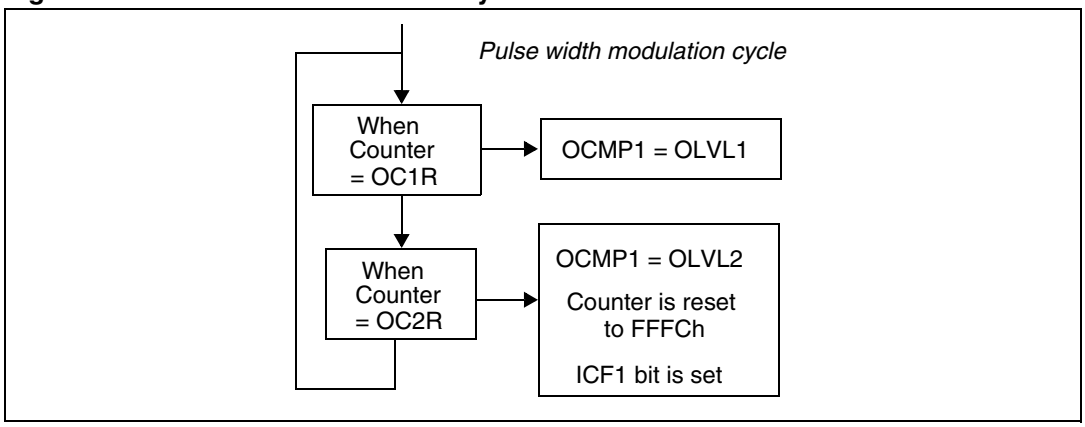
Pulse Width Modulation mode uses the complete Output Compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

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 42*).

Figure 43. Pulse width modulation cycle



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*i*R register value required for a specific timing application can be calculated using the following formula:

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

Where:

- t = Signal or pulse 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 42*)

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

$$\text{OC}i\text{R} = t * f_{EXT} - 5$$

Where:

- t = Signal or pulse period (in seconds)
- f_{EXT} = External timer clock frequency (in hertz)

The Output Compare 2 event causes the counter to be initialized to FFFCh (See *Figure 42*)

Note: 1 After a write instruction to the OCiHR register, the output compare function is inhibited until the OCiLR 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.

9.3.4 Low-power modes

Table 39. 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* 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* bit is set, and the counter value present when exiting from Halt mode is captured into the IC*i*R register.

9.3.5 Interrupts

Table 40. 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 Section 6: Interrupts). 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).

9.3.6 Summary of timer modes

Table 41. 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(1)	No	Partially (2)
PWM mode	No	Not recommended(3)	No	No

1. See note 4 in *Section : One-pulse mode*.
2. See note 5 in *Section : One-pulse mode*.
3. See note 4 in *Section : Pulse width modulation mode*.

9.3.7 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)

Reset value: 0000 0000 (00h)

7							0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
Read/write							

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.

Control register 2 (CR2)

Reset Value: 0000 0000 (00h)

7							0
OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG
Read/Write							

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.

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

The timer clock mode depends on these bits as shown in *Table 42*:

Table 42. Clock control bits

Timer clock	CC1	CC0
f_{CPU} / 4	0	0
f_{CPU} / 2	0	1
f_{CPU} / 8	1	0
External clock (when available)(1)	1	1

1. 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.

Control/status register (CSR)

Reset Value: 0000 0000 (00h)

7							0
ICF1	OCF1	TOF	ICF2	OCF2	TIMD	0	0
Read only					Read/Write	reserved	

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.

Input capture 1 higher register (IC1HR)

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

Reset Value: undefined

7							0
MSB							LSB
Read only							

Input capture 1 lower register (IC1LR)

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

Reset Value: undefined

7							0
MSB							LSB
Read only							

Output compare 1 higher register (OC1HR)

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

Reset Value: 1000 0000 (80h)

7							0
MSB							LSB
Read/write							

Output compare 1 low register (OC1LR)

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

Reset Value: 0000 0000 (00h)

7							0
MSB							LSB
Read/write							

Output compare 2 higher register (OC2HR)

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

Reset value: 1000 0000 (80h)

7							0
MSB							LSB
Read/write							

Output compare 2 lower register (OC2LR)

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

Reset value: 0000 0000 (00h)

7							0
MSB							LSB
Read/write							

Counter higher register (CHR)

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

Reset value: 1111 1111 (FFh)

7							0
MSB							LSB
Read only							

Counter lower register (CLR)

This is an 8-bit register that contains the lower 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.

Reset value: 1111 1100 (FCh)

7							0
MSB							LSB
Read only							

Alternate counter higher register (ACHR)

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

Reset value: 1111 1111 (FFh)

7							0
MSB							LSB
Read only							

Alternate counter lower register (ACLR)

This is an 8-bit register that contains the lower 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.

Reset value: 1111 1100 (FCh)

7							0
MSB							LSB
Read only							

Input capture 2 higher register (IC2HR)

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

Reset value: undefined

7							0
MSB							LSB
Read only							

Input capture 2 lower register (IC2LR)

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

Reset value: undefined

7							0
MSB							LSB
Read only							

Table 43. 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	CR1 reset value	ICIE 0	OCIE 0	TOIE 0	FOLV2 0	FOLV1 0	OLVL2 0	IEDG1 0	OLVL1 0
Timer A: 31 Timer B: 41	CR2 reset value	OC1E 0	OC2E 0	OPM 0	PWM 0	CC1 0	CC0 0	IEDG2 0	EXEDG 0
Timer A: 33 Timer B: 43	CSR reset value	ICF1 x	OCF1 x	TOF x	ICF2 x	OCF2 x	TIMD 0	- x	- x
Timer A: 34 Timer B: 44	IC1HR reset value	MSB x	x	x	x	x	x	x	LSB x
Timer A: 35 Timer B: 45	IC1LR reset value	MSB x	x	x	x	x	x	x	LSB x
Timer A: 36 Timer B: 46	OC1HR reset value	MSB 1	0	0	0	0	0	0	LSB 0
Timer A: 37 Timer B: 47	OC1LR reset value	MSB 0	0	0	0	0	0	0	LSB 0
Timer A: 3E Timer B: 4E	OC2HR reset value	MSB 1	0	0	0	0	0	0	LSB 0
Timer A: 3F Timer B: 4F	OC2LR reset value	MSB 0	0	0	0	0	0	0	LSB 0
Timer A: 38 Timer B: 48	CHR reset value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 39 Timer B: 49	CLR reset value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3A Timer B: 4A	ACHR reset value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 3B Timer B: 4B	ACLR reset value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3C Timer B: 4C	IC2HR reset value	MSB x	x	x	x	x	x	x	LSB x
Timer A: 3D Timer B: 4D	IC2LR reset value	MSB x	x	x	x	x	x	x	LSB x

9.4 Serial peripheral interface (SPI)

9.4.1 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 however the SPI interface can not be a master in a multimaster system.

9.4.2 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 (see note)
- $\overline{SS}$ Management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, Master Mode Fault and Overrun flags

Note: In slave mode, continuous transmission is not possible at maximum frequency due to the software overhead for clearing status flags and to initiate the next transmission sequence.

9.4.3 General description

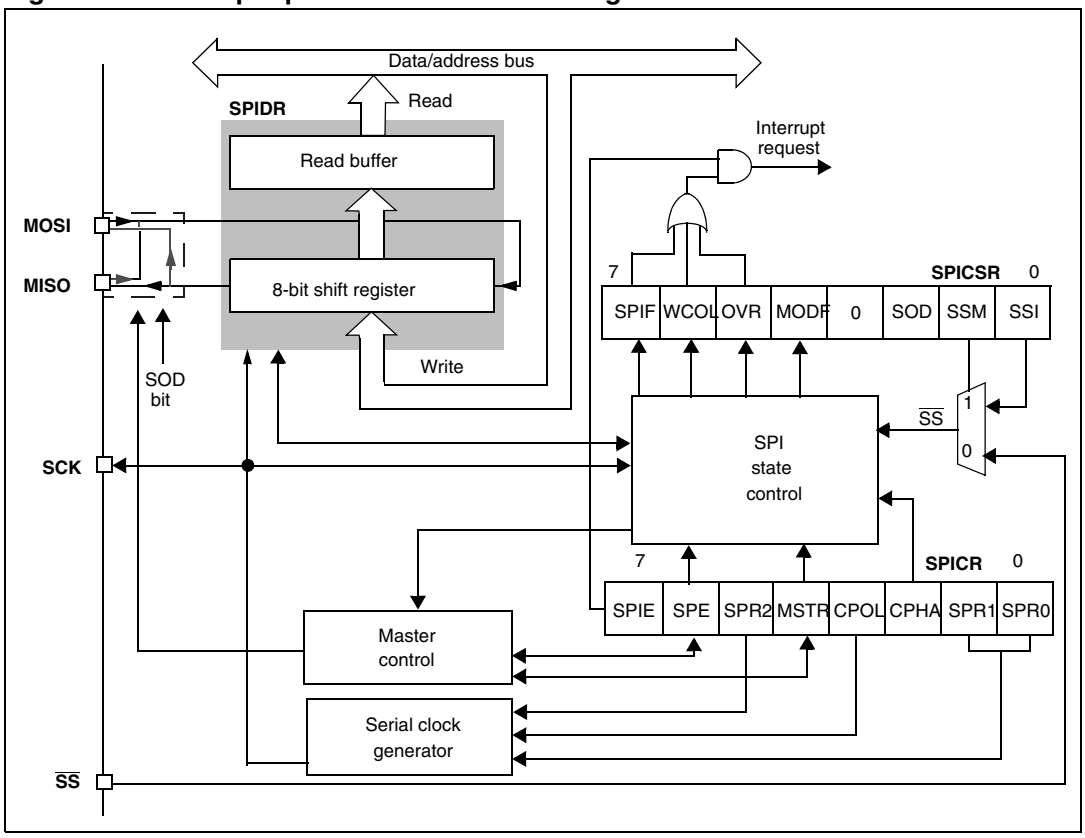
Figure 44 shows the serial peripheral interface (SPI) block diagram. There are 3 registers:

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

The SPI is connected to external devices through 4 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

Figure 44. Serial peripheral interface block diagram



– 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 SS inputs can be driven by standard I/O ports on the master MCU.

Functional description

A basic example of interconnections between a single master and a single slave is illustrated in *Figure 45*.

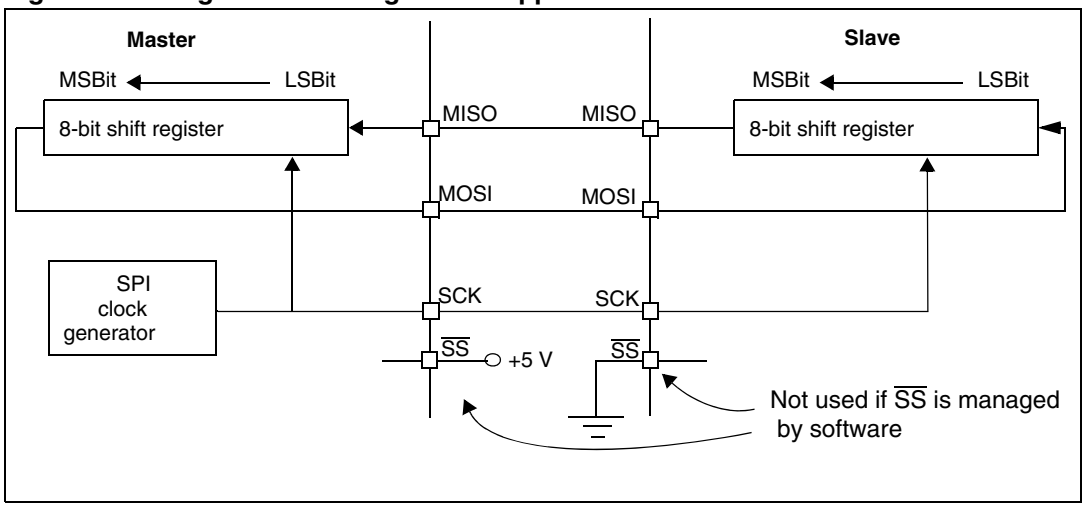
The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

The communication is always initiated by the master. When the master device transmits data to a slave device via MOSI pin, the slave device responds 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 48*) but master and slave must be programmed with the same timing mode.

Figure 45. Single master/ single slave application



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 47*)

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 46*):
  - 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 $\overline{SS}$ 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*).

Figure 46. Generic $\overline{SS}$ timing diagram

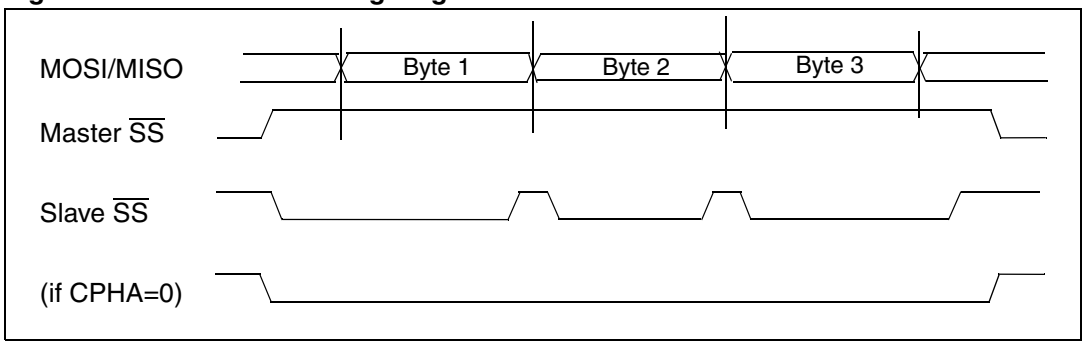
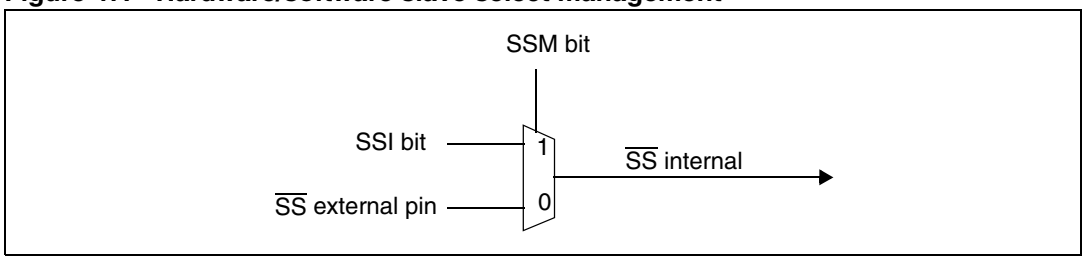


Figure 47. Hardware/software slave select management



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).

How to operate the SPI in master mode

To operate the SPI in master mode, perform the following steps in order:

1. Write to the SPICR 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 48* shows the four possible configurations. Note that the slave must have the same CPOL and CPHA settings as the master.
2. Write to the SPICSR register:
   - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the $\overline{SS}$ pin high for the complete byte transmit sequence.
3. Write to the SPICR register:
   - Set the MSTR and SPE bits. Note that the MSTR and SPE bits remain set only if $\overline{SS}$ is high).

Caution: if the SPICSR register is not written first, the SPICR register setting (MSTR bit) may be not taken into account.

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

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:

1. An access to the SPICSR register while the SPIF bit is set
2. 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.

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:

1. 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 48*).
     Note that the slave must have the same CPOL and CPHA settings as the master.
   - Manage the $\overline{SS}$ pin as described in *Section* and *Figure 46*. 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.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

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:

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

Note: 1 While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

2 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).

9.4.4 Clock phase and clock polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See *Figure 48*).

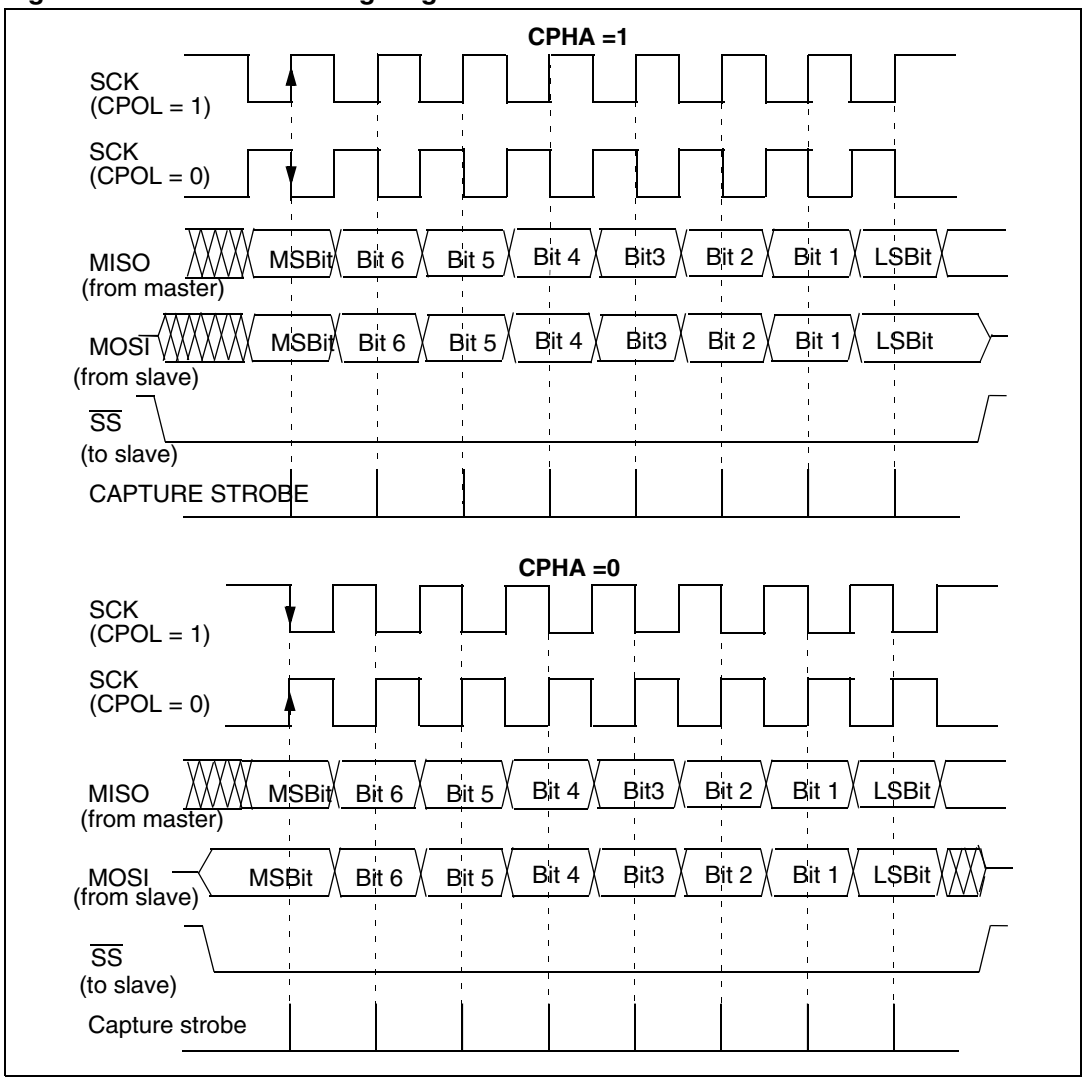
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

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

Figure 48, 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 resetting the SPE bit.

Figure 48. Data clock timing diagram



1. This figure should not be used as a replacement for parametric information. Refer to the *Section 11: Electrical characteristics*.

9.4.5 Error flags

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.

Note: 1 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.

2 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.

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.

Write collision error (WCOL)

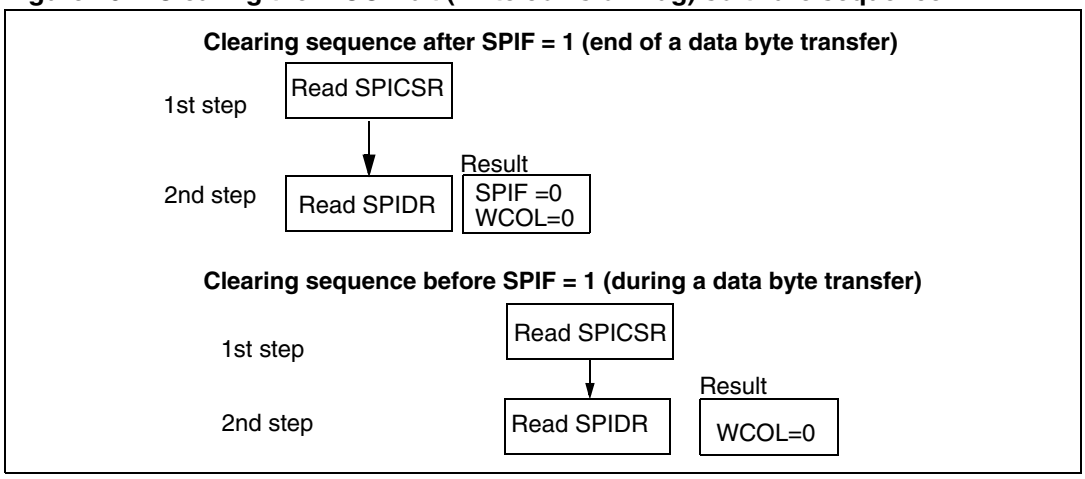
A write collision occurs when the software tries to write to the SPIDR register while a data transfer is taking place with an external device. When this happens, the transfer continues uninterrupted; and the software write will be unsuccessful.

Write collisions can occur both in master and slave mode. See also *Section : Slave select management*.

*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 MCU 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 49*).

Figure 49. Clearing the WCOL bit (write collision flag) software sequence



1. Writing to the SPIDR register instead of reading it does not reset the WCOL bit.

Single master systems

A typical single master system may be configured, using an MCU as the master and four MCUs as slaves (see *Figure 50*).

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.

Figure 50. Single master / multiple slave configuration

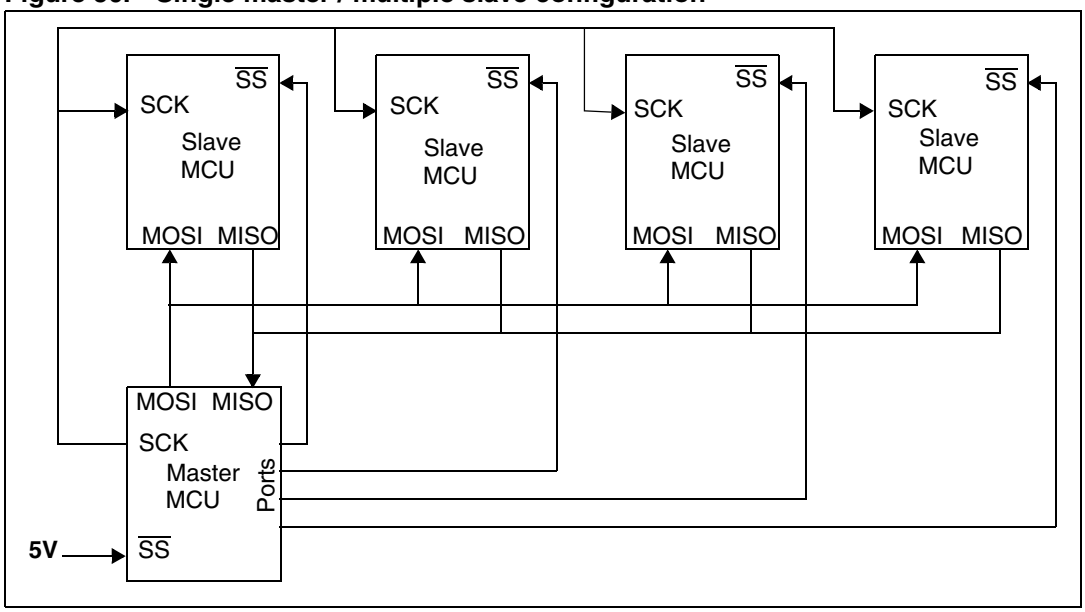


Table 44. Low-power modes

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 MCU 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.

Using the SPI to wake up the MCU from Halt mode

In slave configuration, the SPI is able to wakeup the ST7 device from Halt mode through an 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 ST7 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 ST7 enters Halt mode. So if Slave selection is configured as external (see *Section*), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters Halt mode.

Table 45. 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 Section 6: Interrupts). They generate an interrupt if the corresponding Enable Control bit is set and the interrupt mask in the CC register is reset (RIM instruction).

9.4.6 Register description

Control register (SPICR)

Reset value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0
Read/Write							

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 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 : Master mode fault (MODF)*). 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 46*.

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 : Master mode fault (MODF)*).

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 resetting 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 46. 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

Control/status register (SPICSR)

Reset value: 0000 0000 (00h)

7							0
SPIF	WCOL	OVR	MODF	-	SOD	SSM	SSI
Read only				Reserved	Read/write		

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 49*).

0: No write collision occurred

1: A write collision has been detected

Bit 5 = **OVR** S*PI 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*). An interrupt is generated if SPIE = 1 in SPICR 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 : Master mode fault (MODF)*). 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 SPICR 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** *SS Management.*

This bit is set and cleared by software. When set, it disables the alternate function of the SPI SS pin and uses the SSI bit value instead. See *Section : Slave select management*.

0: Hardware management (SS managed by external pin)

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

Bit 0 = **SSI** *SS Internal Mode.*

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

0: Slave selected

1: Slave deselected

Data I/O register (SPIDR)

Reset value: undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0
Read/write							

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.

Note: 1 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.

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

Caution: 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 44*).

Table 47. SPI register map and reset values

Address (hex.)	Register label	7	6	5	4	3	2	1	0
0021h	SPIDR reset value	MSB x	x	x	x	x	x	x	LSB x
0022h	SPICR reset value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
0023h	SPICSR reset value	SPIF 0	WCOL 0	OVR 0	MODF 0	0	SOD 0	SSM 0	SSI 0

9.5 10-bit A/D converter (ADC)

9.5.1 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 up to 16 multiplexed analog input channels (refer to device pinout description) that allow the peripheral to convert the analog voltage levels from up to 16 different sources.

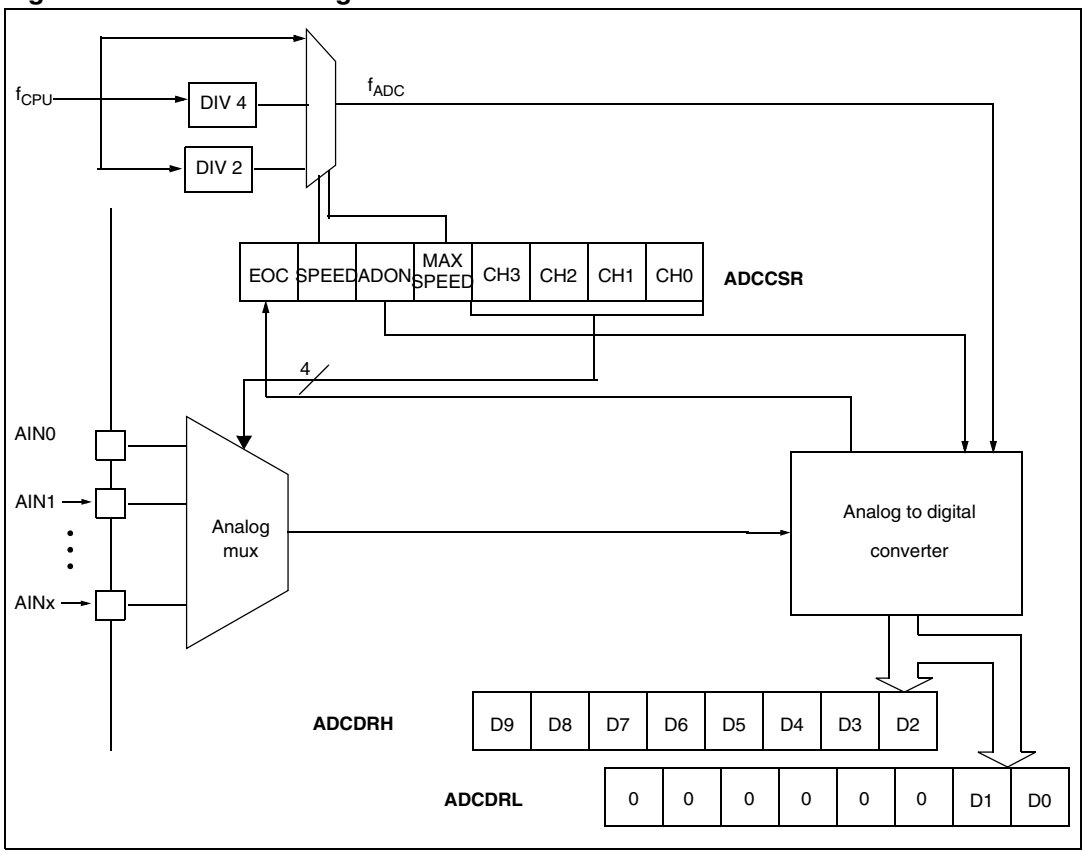
The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

9.5.2 Main features

- 10-bit conversion
- Up to 16 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 51*.

Figure 51. ADC block diagram



9.5.3 Functional description

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_{AREF} (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 allocated time.

A/D converter configuration

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to *Section 8: I/O ports*. 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 CS[3:0] bits to assign the analog channel to convert.

Starting the conversion

In the ADCCSR register:

- 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 resets the EOC bit.

To read the 10 bits, perform the following steps:

1. Poll the EOC bit
2. Read the ADCDRL register
3. Read the ADCDRH register. This clears EOC automatically.

Note: The data is not latched, so both the low and the high data register must be read before the next conversion is complete, so it is recommended to disable interrupts while reading the conversion result.

To read only 8 bits, perform the following steps:

1. Poll the EOC bit
2. Read the ADCDRH register. This clears EOC automatically.

Changing the conversion channel

The application can change channels during conversion. When software modifies the CH[3:0] bits in the ADCCSR register, the current conversion is stopped, the EOC bit is cleared, and the A/D converter starts converting the newly selected channel.

9.5.4 Low-power modes

Note that the A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed.

Table 48. Low-power modes

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 stabilization time t_{STAB} (see Electrical Characteristics) before accurate conversions can be performed.

9.5.5 Interrupts

None.

9.5.6 Register description

Control/status register (ADCCSR)

Reset value: 0000 0000 (00h)

7							0
EOC	SPEED	ADON	MAX_SPEED	CH3	CH2	CH1	CH0
Read only	Read/write						

Bit 7 = **EOC** *End of Conversion*

This bit is set by hardware. It is cleared by hardware when software reads the ADCDRH register or writes to any bit of the ADCCSR register.

0: Conversion is not complete

1: Conversion complete

Bit 6 = **SPEED** *ADC clock selection*

This bit is set and cleared by software. It is used together with the MAX_SPEED bit to select the ADC frequency. Refer to *Table 49*

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 = **MAX_SPEED** *Max. Speed Selection*

This bit is set and cleared by software. It is used together with the SPEED bit to select the ADC frequency. Refer to *Table 49*.

Table 49. ADC speed selection

MAX_SPEED bit	SPEED bit	f_{ADC}
0	0	$f_{CPU}/4$
0	1	$f_{CPU}/2$
1	0	$f_{CPU}/2$
1	1	f_{CPU}(1)

1. The maximum frequency of the ADC is 4 MHz. The MAX_SPEED bit must be kept at 0 (reset state) is f_{CPU} is greater than 4 MHz.

Bit 3:0 = **CH[3:0]** *Channel Selection*

These bits are set and cleared by software. They select the analog input to convert as shown in *Table 50*.

Table 50. ADC channel selection

Channel pin(1)	CH3	CH2	CH1	CH0
AIN0	0	0	0	0
AIN1	0	0	0	1
AIN2	0	0	1	0
AIN3	0	0	1	1
AIN4	0	1	0	0
AIN5	0	1	0	1
AIN6	0	1	1	0
AIN7	0	1	1	1
AIN8	1	0	0	0
AIN9	1	0	0	1
AIN10	1	0	1	0
AIN11	1	0	1	1
AIN12	1	1	0	0
AIN13	1	1	0	1
AIN14	1	1	1	0
AIN15	1	1	1	1

1. The number of channels is device dependent. Refer to the device pinout description.

Data register (ADCDRH)

Reset value: 0000 0000 (00h)

7							0
D9	D8	D7	D6	D5	D4	D3	D2
Read only							

Bit 7:0 = **D[9:2]** *MSB of Converted Analog Value*

Data register (ADCDRL)

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	D1	D0
Reserved						Read only	

Bit 7:2 = Reserved. Forced by hardware to 0.

Bit 1:0 = **D[1:0]** *LSB of Converted Analog Value*

Table 51. ADC register map and reset values

Address (hex.)	Register label	7	6	5	4	3	2	1	0
0070h	**ADCCSR** reset value	EOC 0	SPEED 0	ADON 0	MAX_ SPEED 0	CH3 0	CH2 0	CH1 0	CH0 0
0071h	**ADCDRH** reset value	D9 0	D8 0	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0
0072h	**ADCDRL** reset value	0	0	0	0	0	0	D1 0	D0 0

10 Instruction set

10.1 CPU addressing modes

The CPU features 17 different addressing modes which can be classified into 7 main groups as shown in *Table 52*.

Table 52. Addressing mode 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 submodes 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 53. CPU addressing mode overview

Mode			Syntax	Destination	Pointer address	Pointer size	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

Table 53. CPU addressing mode overview (continued)

Mode			Syntax	Destination	Pointer address	Pointer size	Length (bytes)
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

10.1.1 Inherent

All Inherent instructions consist of a single byte. The opcode fully specifies all the required information for the CPU to process the operation.

Table 54. Inherent instructions

Inherent instruction	Function
NOP	No operation
TRAP	S/W Interrupt
WFI	Wait for interrupt (Low Power mode)
Halt	Halt oscillator (Lowest Power mode)
RET	Subroutine return
IRET	Interrupt subroutine 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

10.1.2 Immediate

Immediate instructions have two bytes, the first byte contains the opcode, the second byte contains the operand value.

Table 55. Immediate instructions

Immediate instruction	Function
LD	Load
CP	Compare
BCP	Bit compare
AND, OR, XOR	Logical operations
ADC, ADD, SUB, SBC	Arithmetic operations

10.1.3 Direct

In Direct instructions, the operands are referenced by their memory address. The direct addressing mode consists of two submodes:

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.

10.1.4 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 submodes:

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.

10.1.5 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 submodes:

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.

10.1.6 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 submodes:

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 56. Instructions supporting direct, indexed, indirect and indirect indexed addressing modes

instructions		Function
Long and short	LD	Load
	CP	Compare
	AND, OR, XOR	Logical operations
	ADC, ADD, SUB, SBC	Arithmetic additions/subtractions operations
	BCP	Bit compare

Table 56. Instructions supporting direct, indexed, indirect and indirect indexed addressing modes (continued)

instructions		Function
Short only	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

10.1.7 Relative mode (Direct, Indirect)

This addressing mode is used to modify the PC register value, by adding an 8-bit signed offset to it.

Table 57. List of Relative Direct/Indirect instructions

Available Relative Direct/indirect instructions	Function
JRxx	Conditional jump
CALLR	Call relative

The relative addressing mode consists of two submodes:

Relative (Direct)

The offset is following the opcode.

Relative (Indirect)

The offset is defined in memory, which address follows the opcode.

10.2 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 *Table 58*:

Table 58. Instruction groups

Group	Instruction							
Load and Transfer	LD	CLR						
Stack operation	PUSH	POP	RSP					
Increment/Decrement	INC	DEC						
Compare and Tests	CP	TNZ	BCP					

Table 58. Instruction groups (continued)

Group	Instruction							
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 prebyte

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.

Table 59. Instruction set overview

Mnemo	Description	Function/example	Dst	Src	I1	H	I0	N	Z	C
ADC	Add with Carry	A = A + M + C	A	M		H		N	Z	C
ADD	Addition	A = A + M	A	M		H		N	Z	C
AND	Logical And	A = A . M	A	M				N	Z	
BCP	Bit compare A, Memory	tst (A . M)	A	M				N	Z	

Table 59. Instruction set overview (continued)

Mnemo	Description	Function/example	Dst	Src	I1	H	I0	N	Z	C
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							C
BTJT	Jump if bit is true (1)	btjt Byte, #3, Jmp1	M							C
CALL	Call subroutine									
CALLR	Call subroutine relative									
CLR	Clear		reg, M					0	1	
CP	Arithmetic Compare	tst(Reg - M)	reg	M				N	Z	C
CPL	One Complement	A = FFH-A	reg, M					N	Z	1
DEC	Decrement	dec Y	reg, M					N	Z	
Halt	Halt				1		0			
IRET	Interrupt routine return	Pop CC, A, X, PC			I1	H	I0	N	Z	C
INC	Increment	inc X	reg, M					N	Z	
JP	Absolute Jump	jp [TBL.w]								
JRA	Jump relative always									
JRT	Jump relative									
JRF	Never jump	jrf *								
JRIH	Jump if ext. INT pin = 1	(ext. INT pin high)								
JRIL	Jump if ext. INT pin = 0	(ext. INT pin low)								
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 >								
JRULE	Jump if (C + Z = 1)	Unsigned <=								
LD	Load	dst <= src	reg, M	M, reg				N	Z	

Table 59. Instruction set overview (continued)

Mnemo	Description	Function/example	Dst	Src	I1	H	I0	N	Z	C
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	Subtraction	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 lbl1						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	

11 Electrical characteristics

11.1 Parameter conditions

Unless otherwise specified, all voltages are referred to V_{SS}.

11.1.1 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 °C and T_A = $T_A$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 (mean±3Σ).

11.1.2 Typical values

Unless otherwise specified, typical data are based on T_A = 25 °C, V_{DD} = 5 V. They are given only as design guidelines and are not tested.

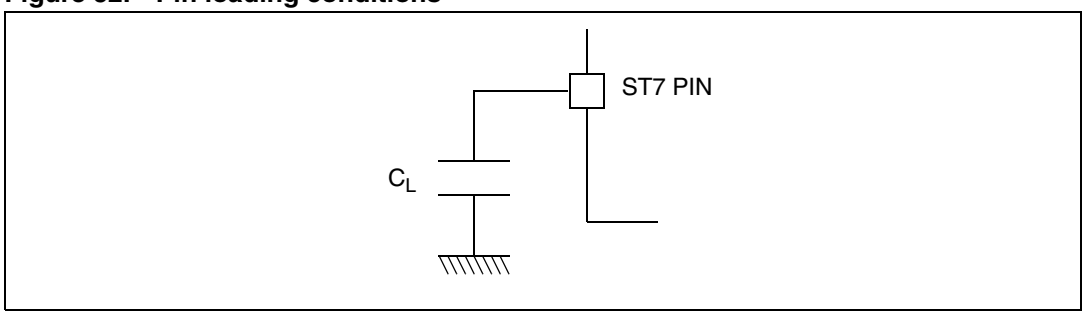
11.1.3 Typical curves

Unless otherwise specified, all typical curves are given only as design guidelines and are not tested.

11.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in *Figure 52*.

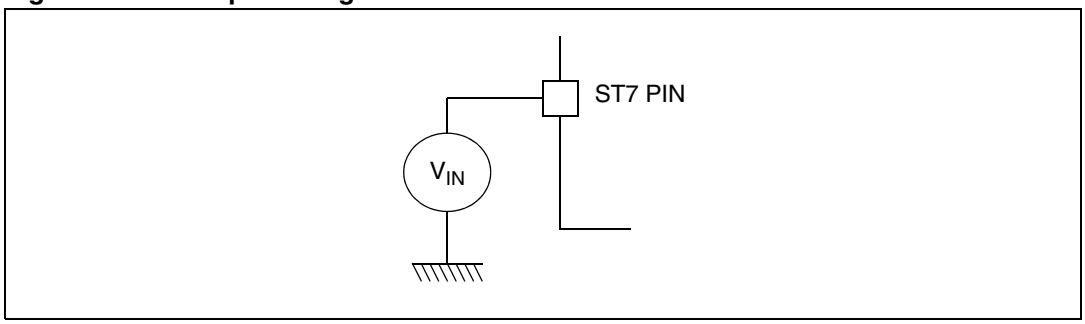
Figure 52. Pin loading conditions



11.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in *Figure 53*.

Figure 53. Pin input voltage



11.2 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.

Table 60. Voltage characteristics

Symbol	Ratings	Maximum value	Unit
V_{DD} - V_{SS}	Supply voltage	6.5	V
V_{IN} 1) & 2)	Input Voltage on true open drain pin	V_{SS} – 0.3 to 6.5	
	Input voltage on any other pin	V_{SS} – 0.3 to V_{DD} + 0.3	
$\lvert\Delta V_{DDx}\rvert$ and $\lvert\Delta V_{SSx}\rvert$	Variations between different digital power pins	50	mV
$\lvert V_{SSA} - V_{SSx}\rvert$	Variations between digital and analog ground pins	50	
$V_{ESD(HBM)}$	Electrostatic discharge voltage (human body model)	see *Section 11.7.3 on page 127*	
$V_{ESD(MM)}$	Electrostatic discharge voltage (machine model)		

Table 61. Current characteristics

Symbol	Ratings		Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source) (1)	32-pin devices	75	mA
		44-pin devices	150	
I_{VSS}	Total current out of V_{SS} ground lines (sink)(1)	32-pin devices	75	mA
		44-pin devices	150	
I_{IO}(2)	Output current sunk by any standard I/O and control pin		25	mA
	Output current sunk by any high sink I/O pin		50	
	Output current source by any I/Os and control pin		–25	
$I_{INJ(PIN)}$(3)(4)	Injected current on $\overline{RESET}$ pin		± 5	
	Injected current on OSC1 and OSC2 pins		± 5	
	Injected current on any other pin (5) & (6)		± 5	
$\Sigma I_{INJ(PIN)}$ (3)	Total injected current (sum of all I/O and control pins) (5)		± 25	

1. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
2. 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: 4.7 kΩ for $\overline{RESET}$, 10 kΩ for I/Os). For the same reason, unused I/O pins must not be directly tied to V_{DD} or V_{SS}.
3. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$. For true open-drain pads, there is no positive injection current, and the corresponding V_{IN} maximum must always be respected.
4. Negative injection disturbs the analog performance of the device. See note in *ADC accuracy on page 145*. For best reliability, it is recommended to avoid negative injection of more than 1.6 mA.
5. 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 characterization with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.
6. True open drain I/O port pins do not accept positive injection.

Table 62. Thermal characteristics

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	–65 to +150	°C
T_J	Maximum junction temperature (see *Section 12.2: Thermal characteristics*)		

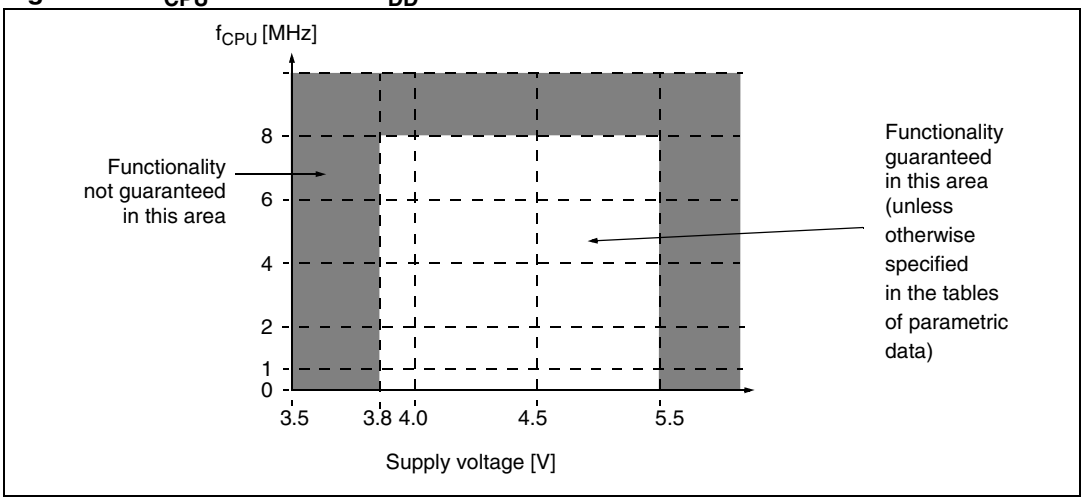
11.3 Operating conditions

11.3.1 Operating conditions (ST72323 5 V devices)

Table 63. Operating conditions (ST72323 5 V devices)

Symbol	Parameter	Conditions	Min	Max	Unit
f_{CPU}	Internal clock frequency		0	8	MHz
V_{DD}	Operating voltage		3.8	5.5	V
T_A	Ambient temperature range	1 suffix version	0	70	°C
		5 suffix version	–10	85	
		6 suffix versions	–40	85	
		7 suffix versions	–40	105	
		3 suffix version	–40	125	

Figure 54. f_{CPU} max versus V_{DD}

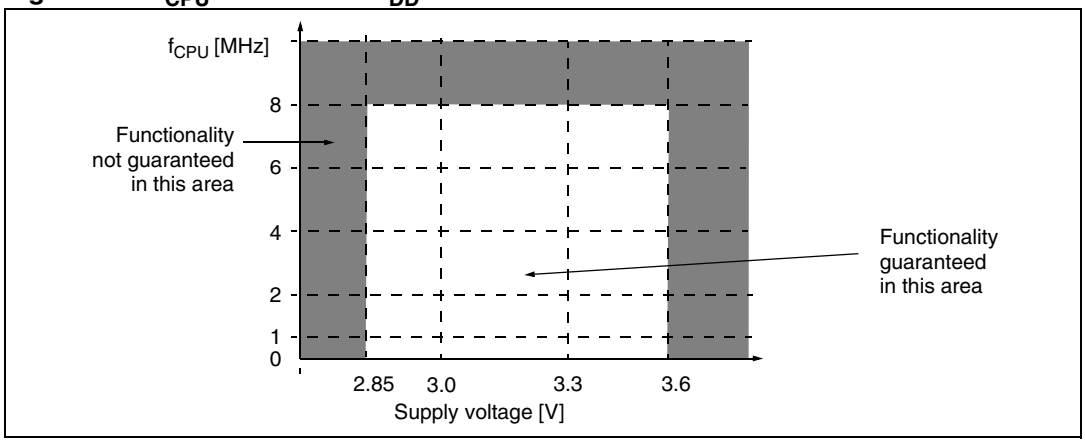


Note: Some temperature ranges are only available with a specific package and memory size. Refer to Ordering Information.

Table 64. Operating conditions (ST72323L 3 V devices)

Symbol	Parameter	Conditions	Min	Max	Unit
f_{CPU}	Internal clock frequency		0	8	MHz
V_{DD}	Operating Voltage		2.85	3.6	V
T_A	Ambient temperature range	1 suffix version	0	70	°C
		5 suffix version	–10	85	
		6 suffix version	–40	85	

Figure 55. f_{CPU} max versus V_{DD}



11.4 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).

Table 65. Current consumption

Symbol	Parameter	Conditions	V_{DD} = 3 V		V_{DD} = 5 V		Unit
			Typ	Max(1)	Typ	Max(1)	
I_{DD}	Supply current in Run mode(2)	f_{OSC} = 2 MHz, f_{CPU} = 1 MHz f_{OSC} = 4 MHz, f_{CPU} = 2 MHz f_{OSC} = 8 MHz, f_{CPU} = 4 MHz f_{OSC} = 16 MHz, f_{CPU} = 8 MHz	0.23 0.45 0.88 1.8	0.5 1.0 2.0 4.0	0.46 0.93 1.9 3.7	0.69 1.4 2.7 5.5	mA
	Supply current in Slow mode(2)	f_{OSC}=2MHz, f_{CPU} = 62.5 kHz f_{OSC} = 4 MHz, f_{CPU} = 125 kHz f_{OSC} = 8 MHz, f_{CPU} = 250 kHz f_{OSC} = 16 MHz, f_{CPU} = 500 kHz	15 40 80 170	45 90 180 350	30 70 150 310	60 120 250 500	µA
	Supply current in Wait mode(2)	f_{OSC} = 2 MHz, f_{CPU} = 1 MHz f_{OSC} = 4 MHz, f_{CPU} = 2 MHz f_{OSC} = 8 MHz, f_{CPU} = 4 MHz f_{OSC} = 16 MHz, f_{CPU} = 8 MHz	0.12 0.22 0.43 0.83	0.25 0.5 1 2	0.22 0.45 0.91 1.82	0.37 0.75 1.5 3	mA
	Supply current in Slow Wait mode(2)	f_{OSC}=2MHz, f_{CPU} = 62.5 kHz f_{OSC} = 4 MHz, f_{CPU} = 125 kHz f_{OSC} = 8 MHz, f_{CPU} = 250 kHz f_{OSC} = 16 MHz, f_{CPU} = 500 kHz	10 20 50 100	31 63 125 250	20 40 90 190	40 90 180 350	µA
	Supply current in Halt mode (3)	–40 °C ≤ T_A ≤ +85 °C	<1	10	<1	10	µA
		–40 °C ≤ T_A ≤ +125 °C			<1	50	
I_{DD}	Supply current in Active-Halt mode (4)	f_{OSC} = 2 MHz f_{OSC} = 4 MHz f_{OSC} = 8 MHz f_{OSC} = 16 MHz	45	100	11 22 43 85	15 30 60 150	µA

1. Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.
2. Measurements are done in the following conditions:
   - Program executed from RAM, CPU running with RAM 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.
   - In Slow and Slow Wait mode, f_{CPU} is based on f_{OSC} divided by 32.

   To obtain the total current consumption of the device, add the clock source (*Section 11.5.1*) and the peripheral power consumption (*Section 11.4.2*).
3. All I/O pins in push-pull 0 mode (when applicable) with a static value at V_{DD} or V_{SS} (no load). Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.
4. *Data based on characterization results, not tested in production. All I/O pins in push-pull 0 mode (when applicable) with a static value at V_{DD} or V_{SS} (no load); clock input (OSC1) driven by external square wave. To obtain the total current consumption of the device, add the clock source consumption (Section 11.5.1).*

Power consumption vs. f_{CPU}: 3 V ROM devices

Figure 56. Typical I_{DD} in Run mode

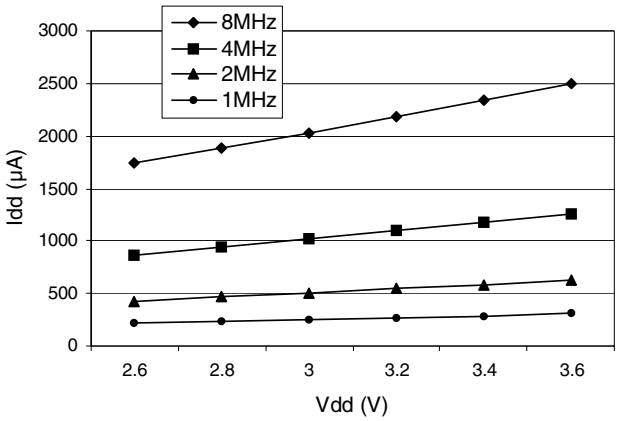


Figure 57. Typical I_{DD} Slow mode

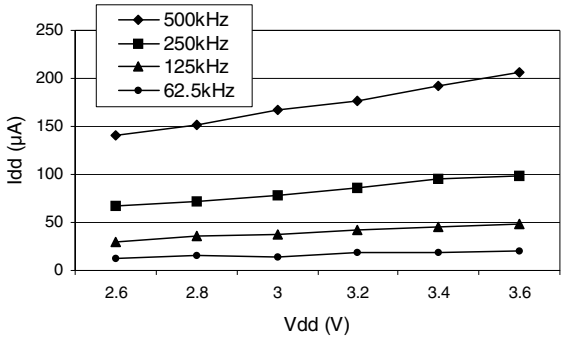


Figure 58. Typical I_{DD} Wait mode

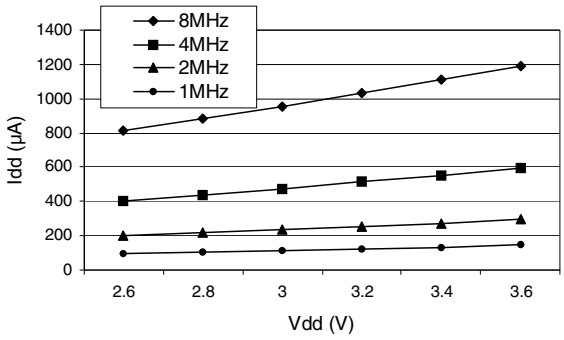
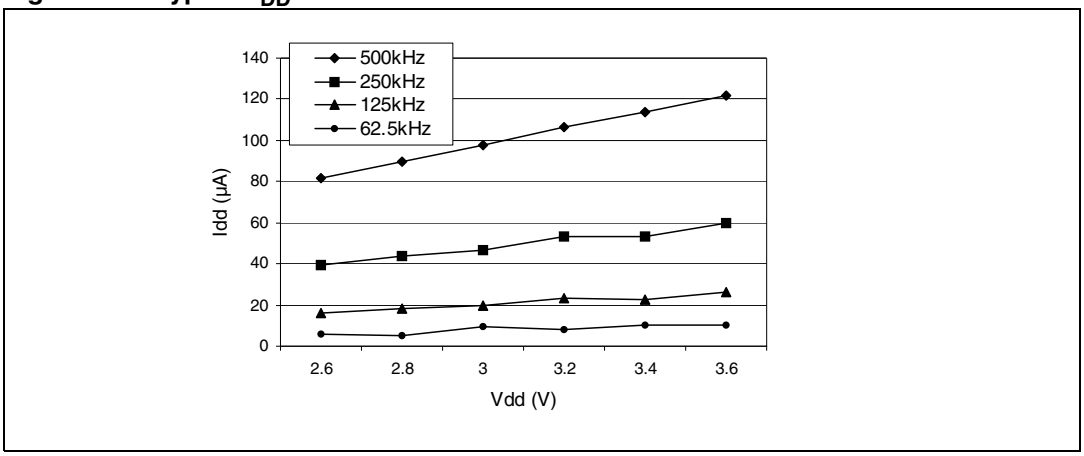


Figure 59. Typical I_{DD} Slow-Wait mode



Power consumption vs. f_{CPU}: 5 V ROM devices

Figure 60. Typical I_{DD} in Run mode

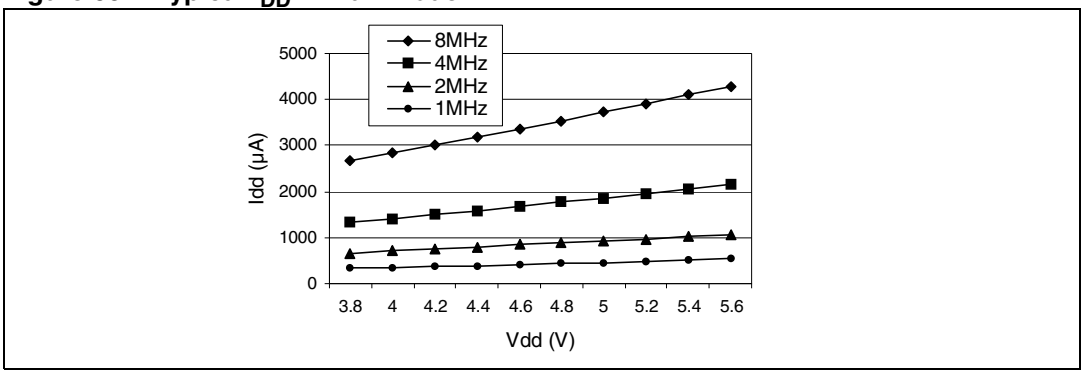


Figure 61. Typical I_{DD} in Slow mode

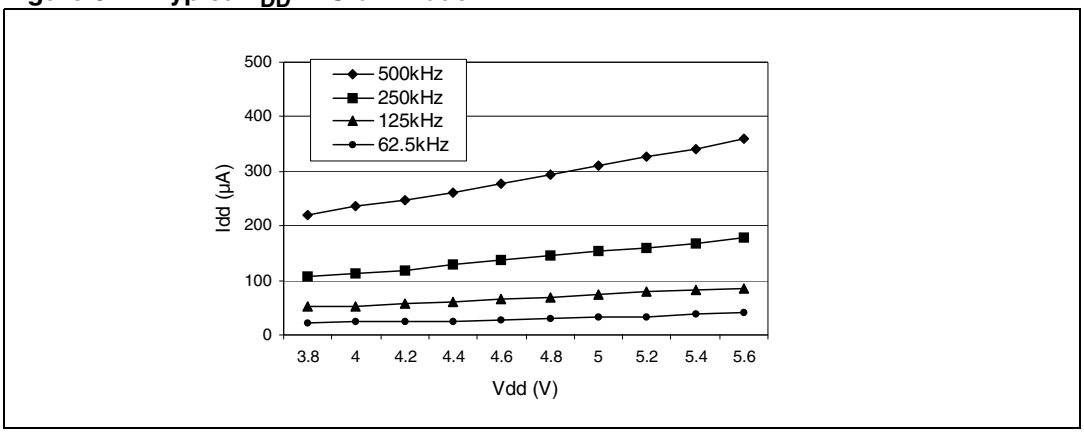


Figure 62. Typical I_{DD} in Wait mode

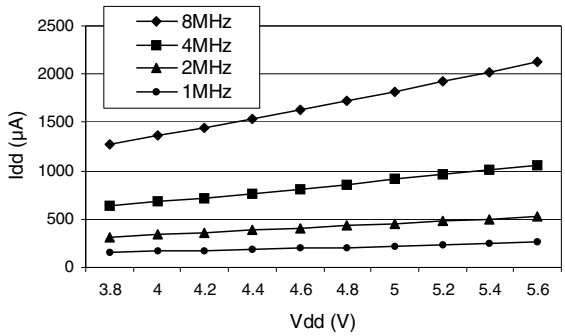
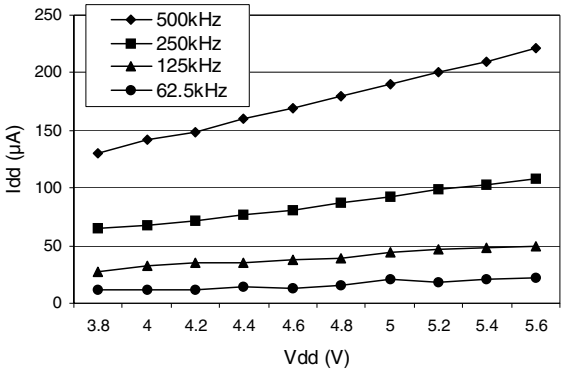


Figure 63. Typ. I_{DD} in Slow-Wait mode



11.4.1 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).

Table 66. Oscillator current consumption

Symbol	Parameter	Conditions	Typ	Max	Unit
$I_{DD(RCINT)}$	Supply current of internal RC oscillator		625		µA
$I_{DD(RES)}$	Supply current of resonator oscillator(1) (2)		see *Section 11.5.1 on page 124*		

1. Data based on characterization results done with the external components specified in *Section 11.5.1*, not tested in production.
2. As the oscillator is based on a current source, the consumption does not depend on the voltage.

11.4.2 On-chip peripherals

Table 67. On-chip peripheral current consumption(1)

Symbol	Parameter	Conditions	Typ	Unit
$I_{DD(TIM)}$	16-bit Timer supply current(2)	V_{DD} = 5.0 V	50	µA
		V_{DD} = 3.3 V	20	
$I_{DD(SPI)}$	SPI supply current(3)	V_{DD} = 5.0 V	400	
		V_{DD} = 3.3 V	250	
$I_{DD(ADC)}$	ADC supply current when converting (4)	V_{DD} = 5.0 V	300	
		V_{DD} = 3.3 V	400	

1. T_A = 25 °C f_{CPU} = 4 MHz.
2. Data based on a differential I_{DD} measurement between reset configuration (timer counter running at $f_{CPU}/4$) and timer counter stopped (only TIMD bit set). Data valid for one timer.
3. Data based on a differential I_{DD} measurement between reset configuration (SPI disabled) and a permanent SPI master communication at maximum speed (data sent equal to 55h). This measurement includes the pad toggling consumption.
4. Data based on a differential I_{DD} measurement between reset configuration and continuous ADC conversion.

11.5 Clock and timing characteristics

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A.

Table 68. General timings

Symbol	Parameter	Conditions	Min	Typ (1)	Max	Unit
$t_{c(INST)}$	Instruction cycle time		2	3	12	t_{CPU}
		f_{CPU} = 8 MHz	250	375	1500	ns
$t_{v(IT)}$	Interrupt reaction time (2) $t_{v(IT)} = \Delta t_{c(INST)} + 10$		10		22	t_{CPU}
		f_{CPU} = 8 MHz	1.25		2.75	µs

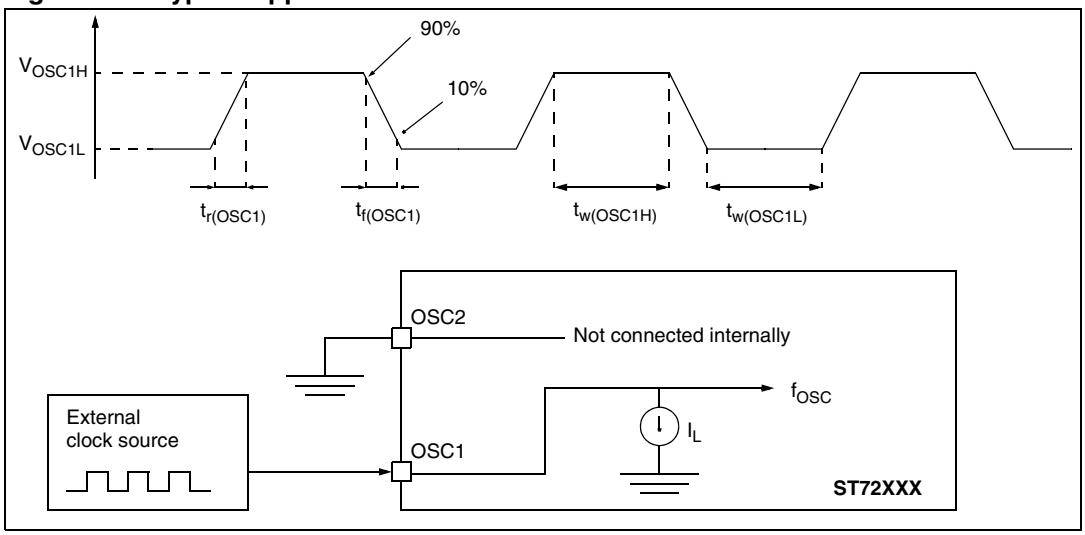
1. Data based on typical application software.
2. Time measured between interrupt event and interrupt vector fetch. $\Delta t_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.

Table 69. External clock source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OSC1H}	OSC1 input pin high level voltage	see *Figure 64*	V_{DD} – 1		V_{DD}	V
V_{OSC1L}	OSC1 input pin low level voltage		V_{SS}		V_{SS} + 1	
$t_{w(OSC1H)}$ $t_{w(OSC1L)}$	OSC1 high or low time(1)		5			ns
$t_{r(OSC1)}$ $t_{f(OSC1)}$	OSC1 rise or fall time (1)				15	
I_L	OSC1 Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			±1	µA

1. Data based on design simulation and/or technology characteristics, not tested in production.

Figure 64. Typical application with an external clock source



11.5.1 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...).

Table 70. Crystal and ceramic resonator oscillators

Symbol	Parameter	Conditions	Min	Max	Unit
f_{OSC}	Oscillator frequency(1)		1	16	MHz
R_F	Feedback resistor(2)		20	40	kΩ
C_{L1} C_{L2}	Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator (R_S)(3)	f_{OSC}= 1 to 2 MHz f_{OSC}= 2 to 4 MHz f_{OSC}= 4 to 8 MHz f_{OSC}= 8 to 16 MHz	20 20 15 15	60 50 35 35	pF
i_2	OSC2 driving current	$V_{IN} = V_{SS}$ f_{OSC} = 1 to 2 MHz f_{OSC} = 2 to 4 MHz f_{OSC} = 4 to 8 MHz f_{OSC} = 8 to 16 MHz	 80 160 310 610	 150 250 460 910	µA

1. The oscillator selection can be optimized in terms of supply current using an high quality resonator with small RS value. Refer to crystal/ceramic resonator manufacturer for more details.
2. Data based on characterization results, not tested in production. The relatively low value of the RF resistor, offers a good protection against issues resulting from use in a humid environment, due to the induced leakage and the bias condition change. However, it is recommended to take this point into account if the µC is used in tough humidity conditions.
3. For C_{L1} and C_{L2} it is recommended to use high-quality ceramic capacitors in the 5-pF to 25-pF range (typ.) designed for high-frequency applications and selected to match the requirements of the crystal or resonator. C_{L1} and C_{L2}, are usually the same size. The crystal manufacturer typically specifies a load capacitance which is the series combination of C_{L1} and C_{L2}. PCB and MCU pin capacitance must be included when sizing C_{L1} and C_{L2} (10 pF can be used as a rough estimate of the combined pin and board capacitance).

Figure 65. Typical application with a crystal or ceramic resonator

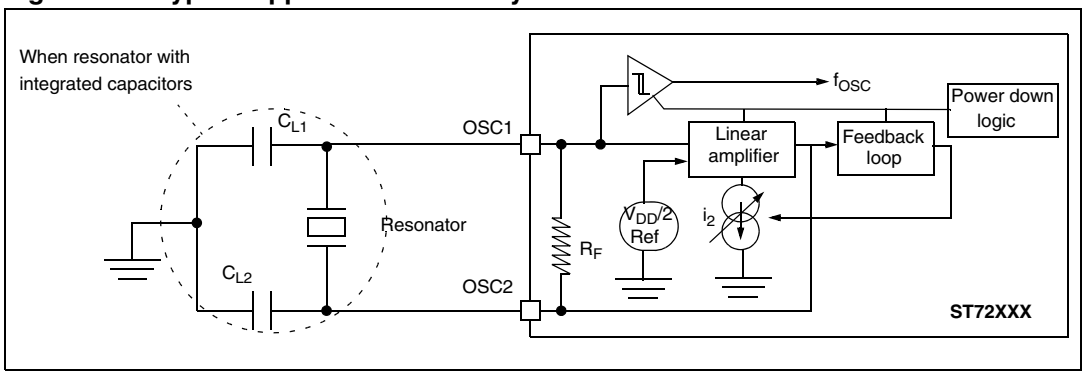


Table 71. Typical resonator selection

Supplier	f_{OSC} (MHz)	Typical ceramic resonators(1)
		Reference(2)
Murata	2	CSTCC2M00G56A-R0
	4	CSTCR4M00G55B-R0
	8	CSTCE8M00G52A-R0
	16	CSTCE16M0V51A-R0

1. Resonator characteristics given by the ceramic resonator manufacturer.
2. SMD = [-R0: Plastic tape package (∅ =180 mm), -B0: Bulk]
LEAD = [-A0: Flat pack package (Radial taping Ho= 18 mm), -B0: Bulk]

11.5.2 RC oscillators

Table 72. RC oscillators(1)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{CPU} (RCINT)	CPU frequency with internal RC oscillator	T_A = 0 to 70 °C, V_{DD} = 3 V to 3.6 V	6	7	8	MHz
		T_A = –10 to 85 °C, V_{DD} = 4.5 V to 5.5 V	6	7	8	MHz
		T_A = –40 to 125 °C, V_{DD} = 4.5 V to 5.5 V	TBD	7	TBD	

1. To reduce disturbance to the RC oscillator, it is recommended to place decoupling capacitors between V_{DD} and V_{SS}.

11.6 Memory characteristics

11.6.1 RAM and hardware registers

Table 73. RAM and hardware registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode (1)	Halt mode (or Reset)	1.6			V

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). Not tested in production.

11.7 EMC characteristics

Susceptibility tests are performed on a sample basis during product characterization.

11.7.1 Functional EMS (electromagnetic susceptibility)

Based on a simple running application on the product (toggling 2 LEDs through I/O ports), the product is stressed by two electromagnetic events until a failure occurs (indicated by the LEDs).

- **ESD**: Electrostatic 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. The test results are given in the table below based on the EMS levels and classes defined in application note AN1709.

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 behavior is detected, the software can be hardened to prevent unrecoverable errors from occurring (see application note AN1015)

Table 74. EMS test results

Symbol	Parameter	Conditions	Level/Class(1)
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	V_{DD} = 5 V, T_A = +25 °C, f_{OSC} = 8 MHz conforms to IEC 1000-4-2	4A
V_{FFTB}	Fast transient voltage burst limits to be applied through 100 pF on V_{DD} and V_{DD} pins to induce a functional disturbance	V_{DD} = 5 V, T_A = +25 °C, f_{OSC} = 8 MHz conforms to IEC 1000-4-4	4A

1. Design target value only.

11.7.2 Electromagnetic 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.

Table 75. EMI emissions(1)

Symbol	Parameter	Conditions	Device/ package(2)	Monitored frequency band	Max vs. [f_{OSC}/f_{CPU}]		Unit
					8/4 MHz	16/8 MHz	
S_{EMI}	Peak level	V_{DD} = 5 V, T_A = +25 °C conforming to SAE J 1752/3	LQFP44	0.1 MHz to 30 MHz	16	21	dBµV
				30 MHz to 130 MHz	24	29	
				130 MHz to 1 GHz	14	21	
				SAE EMI level	3.0	3.5	-
			LQFP32	0.1 MHz to 30 MHz	12	15	dBµV
				30 MHz to 130 MHz	23	26	
				130 MHz to 1 GHz	15	20	
				SAE EMI level	3.0	3.5	-

1. Data based on characterization results, not tested in production.
2. Refer to Application Note AN1709 for data on other package types.

11.7.3 Absolute maximum ratings (electrical sensitivity)

Based on two different tests (ESD and LU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity. .

Electrostatic discharge (ESD)

Electrostatic 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 on the number of supply pins in the device (3 parts*(n+1) supply pin). Two models can be simulated: Human Body Model and Machine Model. This test conforms to the JESD22-A114A/A115A standard. For more details, refer to the application note AN1181.

Table 76. ESD absolute maximum ratings

Symbol	Ratings	Conditions	Maximum value(1)	Unit
$V_{ESD(HBM)}$	Electrostatic discharge voltage (human body model)	T_A = +25 °C	2000	V
$V_{ESD(MM)}$	Electrostatic discharge voltage (machine model)	T_A = +25 °C	200	
$V_{ESD(CD)}$	Electrostatic discharge voltage (charged device model)	T_A = +25 °C	500	

1. Data based on characterization results, not tested in production.

Static latch-up

Two complementary static tests are required on ten parts to assess the latch-up performance:

- A supply overvoltage is applied to each power supply pin.
- A current injection is applied to each input, output and configurable I/O pin.

These tests are compliant with the EIA/JESD 78 IC latch-up standard.

Table 77. Electrical sensitivities

Symbol	Parameter	Conditions	Class (1)
LU	Static latch-up class	T_A = +25 °C T_A = +85 °C T_A = +125 °C	A A A

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).

11.8 I/O port pin characteristics

11.8.1 General characteristics

Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Table 78. General characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage (standard voltage devices)(1)					$0.3V_{DD}$	V
V_{IH}	Input high level voltage(1)			$0.7V_{DD}$			V
V_{hys}	Schmitt trigger voltage hysteresis(2)				0.7		
$I_{INJ(PIN)}$(3)	Injected current on other I/O pins	V_{DD} = 5 V				±4	mA
$\Sigma I_{INJ(PIN)}$(3)	Total injected current (sum of all I/O and control pins)					±25	
I_{lkg}	Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$				±1	µA
I_S	Static current consumption induced by each floating input pin	Floating input mode(4)			200		
R_{PU}	Weak pull-up equivalent resistor(5)	$V_{IN} = V_{SS}$	V_{DD} = 5 V	50	120	250	kΩ
C_{IO}	I/O pin capacitance				5		pF
$t_{f(IO)out}$	Output high to low level fall time(1)	C_L = 50 pF Between 10% and 90%			25		ns
$t_{r(IO)out}$	Output low to high level rise time(1)				25		
$t_{w(IT)in}$	External interrupt pulse time(6)			1			t_{CPU}

1. Data based on characterization results, not tested in production.
2. Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.
3. 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 Table 61. on page 116* for more details.
4. Static peak current value taken at a fixed V_{IN} value, based on design simulation and technology characteristics, not tested in production. This value depends on V_{DD} and temperature values)?
5. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in *Figure 67*).
6. 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 66. Unused I/O pins configured as input

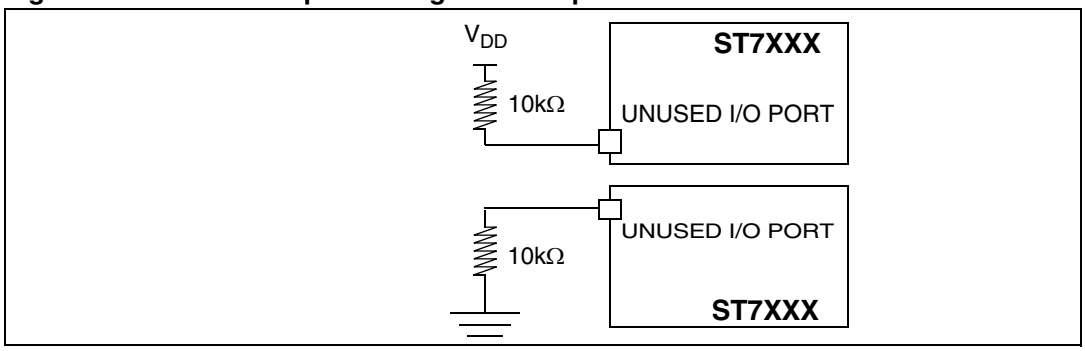
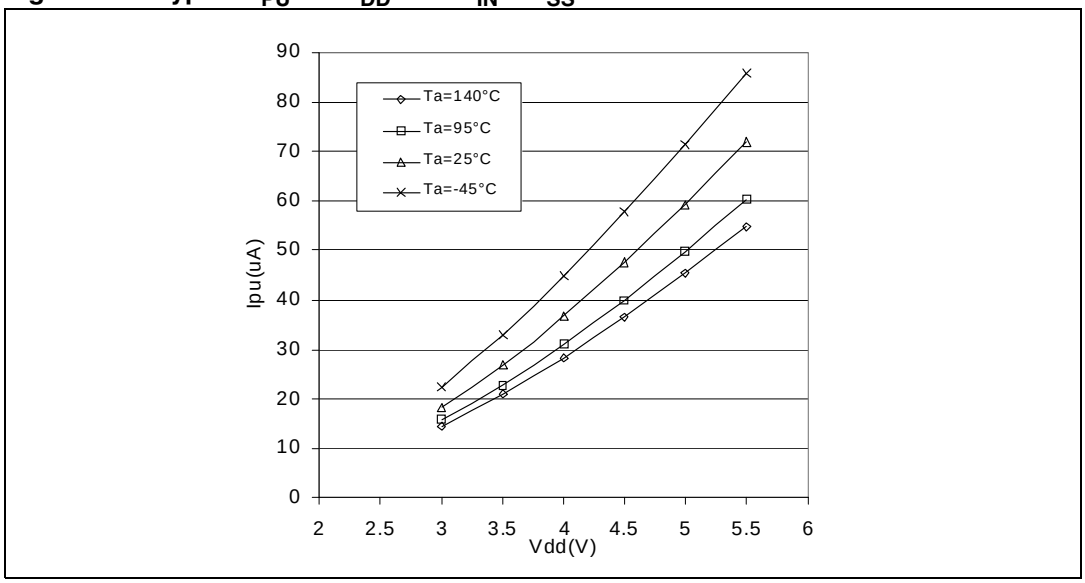


Figure 67. Typical I_{PU} vs. V_{DD} with $V_{IN} = V_{SS}$



11.8.2 Output driving current

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Table 79. Output driving current

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 (see *Figure 68*)	V_{DD} = 5 V, I_{IO} = +5 mA			1.2	V
		V_{DD} = 5 V, I_{IO} = +2 mA			0.5	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see *Figure 69* and *Figure 71*)	V_{DD} = 5 V, I_{IO} = +20 mA	$T_A \le 85$ °C $T_A > 85$°C		1.3 1.5	
		V_{DD} = 5 V, I_{IO}=+8mA			0.6	
V_{OH} (2)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see *Figure 70* and *Figure 73*)	V_{DD} = 5 V, I_{IO} = –5 mA,	$T_A \le 85$ °C $T_A > 85$ °C	V_{DD}–1.4 V_{DD}–1.6		
		V_{DD} = 5 V, I_{IO} = –2 mA		V_{DD}–0.7		
V_{OL} (1)	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time (see *Figure 74* and *Figure 77*)	V_{DD} = 3 V, I_{IO} = +2 mA			0.7	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see *Figure 75* and *Figure 78*)	V_{DD} = 3 V, I_{IO} = +10 mA			0.7	
		V_{DD} = 3 V, I_{IO} = +14 mA, 0 °C ≤ T_A ≤ 70 °C			0.9	
V_{OH} (2)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see *Figure 76* and *Figure 79*)	V_{DD} = 3 V, I_{IO} = –2 mA		V_{DD}–0.9		

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in *Section Table 61.* 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 Table 61.* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}. True open drain I/O pins do not have V_{OH}.

ST72323 5 V devices

Figure 68. Typical V_{OL} at V_{DD} = 5 V (std. ports)

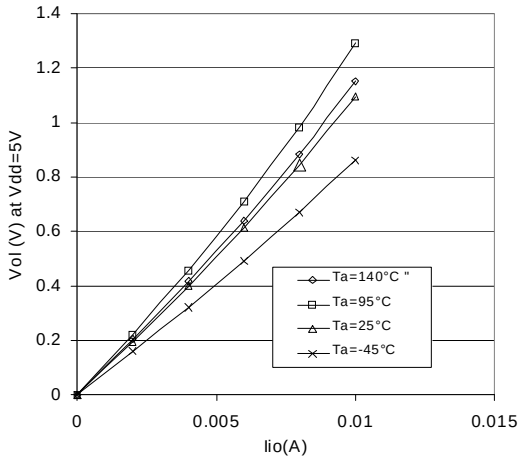


Figure 69. Typ. V_{OL} at V_{DD} = 5 V (high-sink ports)

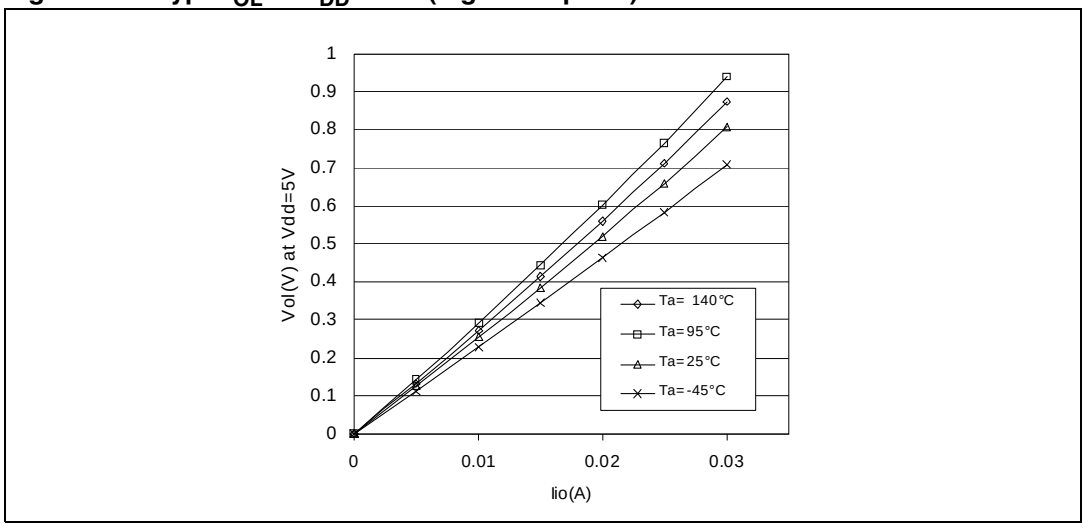


Figure 70. Typical V_{OH} at V_{DD} = 5 V

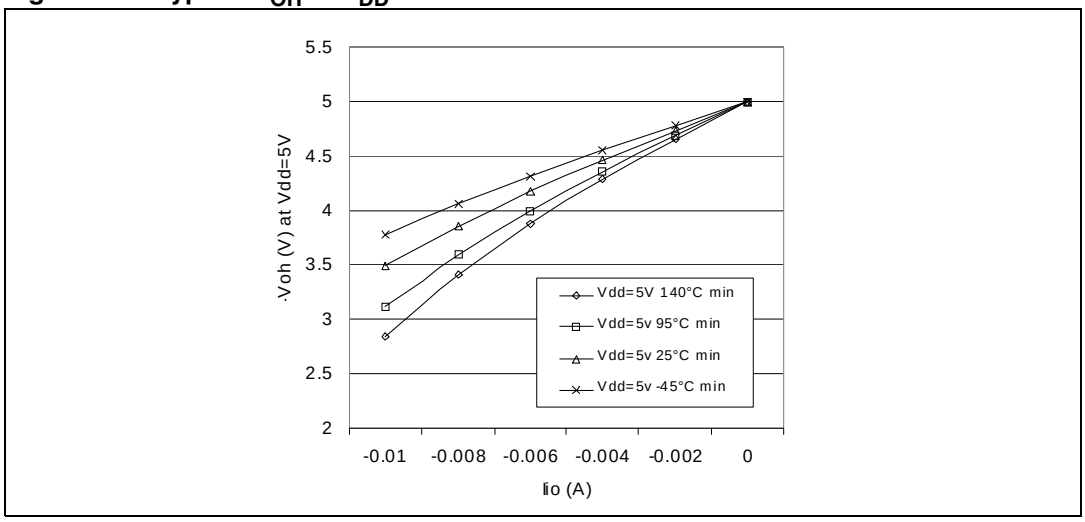


Figure 71. Typical V_{OL} vs. V_{DD} (std. ports)

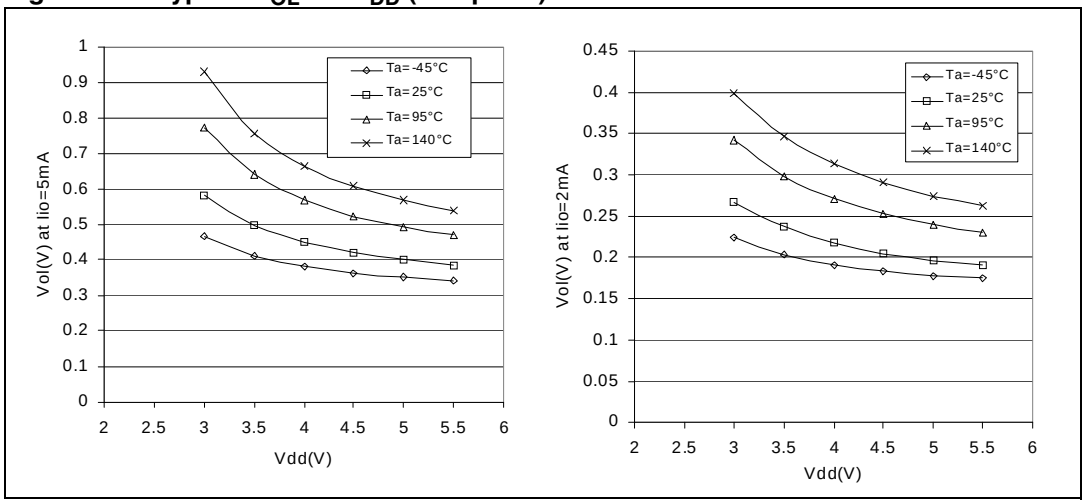


Figure 72. Typical V_{OL} vs. V_{DD} (high-sink ports)

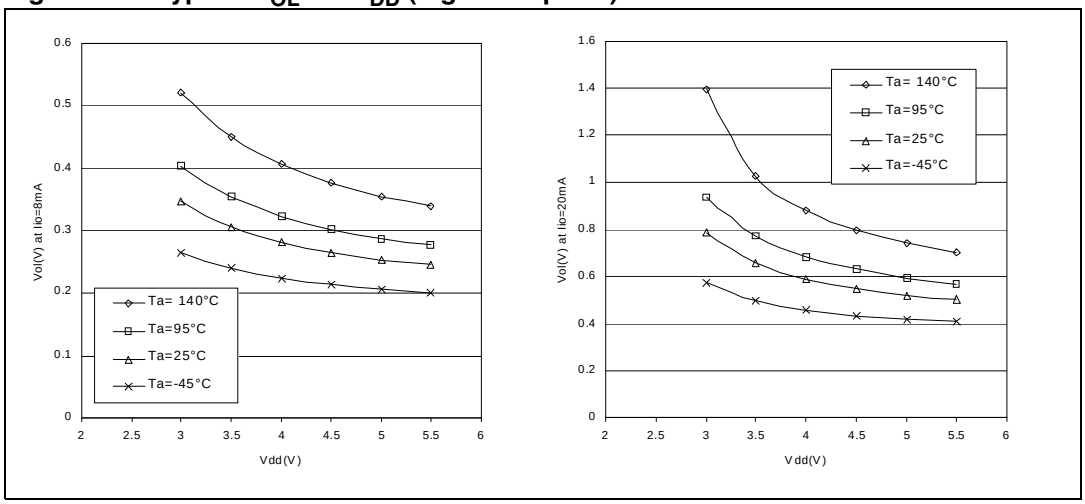
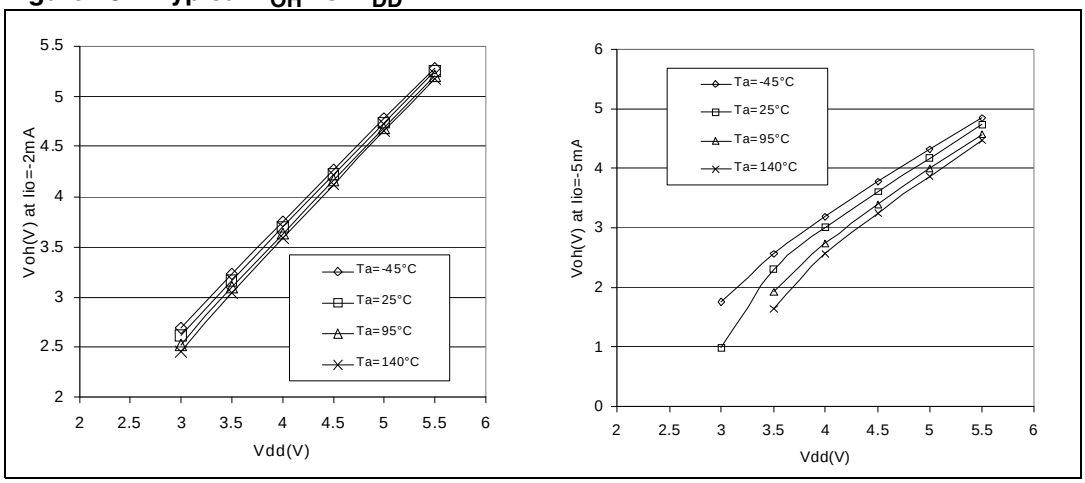


Figure 73. Typical V_{OH} vs. V_{DD}



ST72323L 3 V devices

Figure 74. Typical V_{OL} at V_{DD} = 3 V (std. ports)

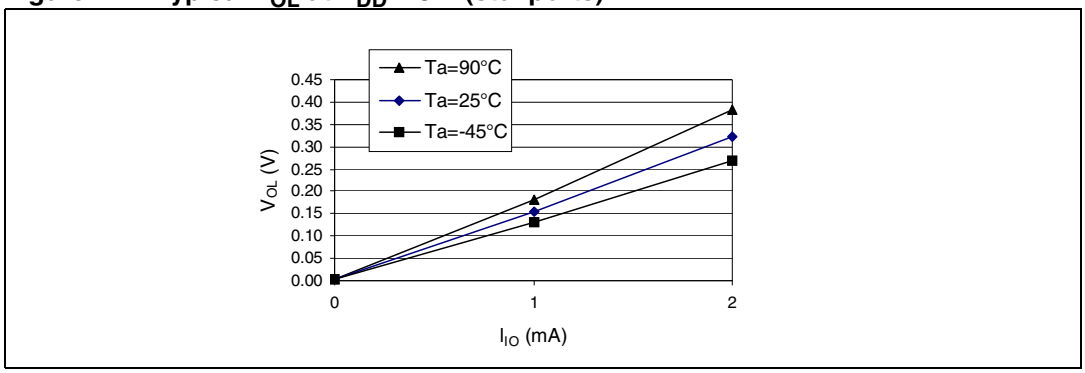


Figure 75. Typ. V_{OL} at V_{DD} = 3 V (high-sink ports)

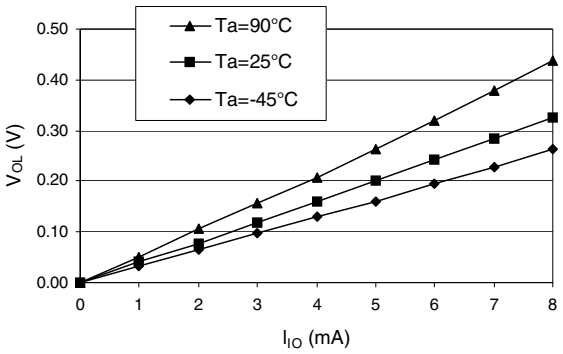


Figure 76. Typical V_{OH} at V_{DD} = 3 V

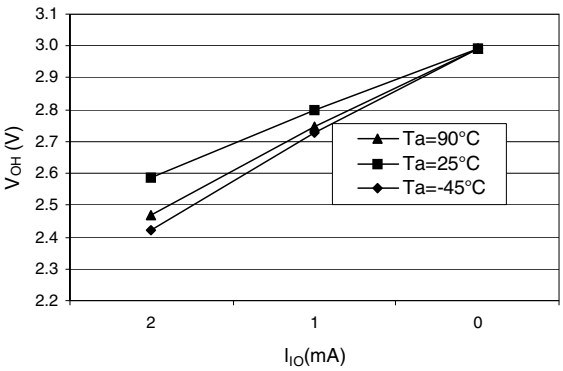


Figure 77. Typical V_{OL} vs. V_{DD} (std. ports)

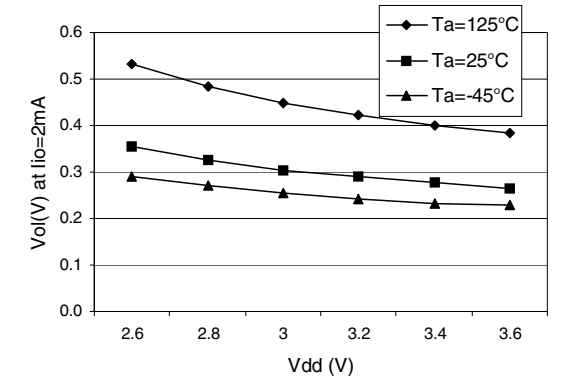


Figure 78. Typ. V_{OL} vs. V_{DD} (high-sink ports)

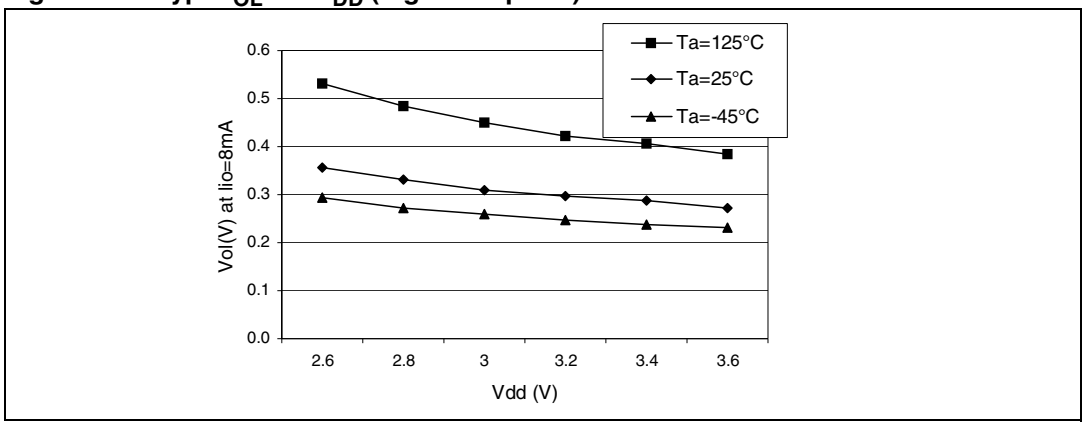
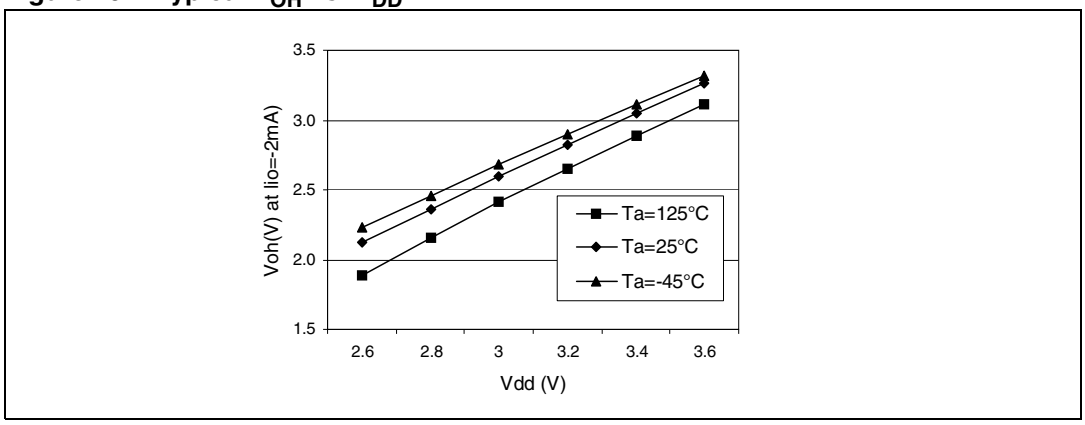


Figure 79. Typical V_{OH} vs. V_{DD}



11.9 Control pin characteristics

11.9.1 Asynchronous $\overline{\text{RESET}}$ pin

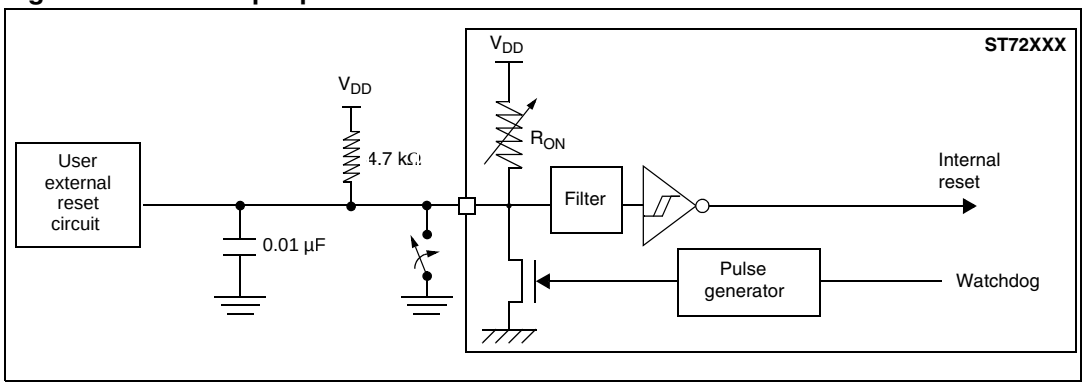
Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Table 80. Asynchronous $\overline{\text{RESET}}$ pin characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage (1)					$0.3V_{DD}$	V
V_{IH}	Input high level voltage (1)			$0.7V_{DD}$			
V_{hys}	Schmitt trigger voltage hysteresis(2)				2.5		
V_{OL}	Output low level voltage (3)	V_{DD} = 5 V	I_{IO} = +2 mA		0.2	0.5	
I_{IO}	Driving current on $\overline{\text{RESET}}$ pin				2		mA
R_{ON}	Weak pull-up equivalent resistor	V_{DD} = 5 V		20	30	120	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	Watchdog reset		20(4)			µs
$t_{h(RSTL)in}$	External reset pulse hold time(5)			2.5			µs
$t_{g(RSTL)in}$	Filtered glitch duration(6)				200		ns

1. Data based on characterization results, not tested in production.
2. Hysteresis voltage between Schmitt trigger switching levels.
3. The I_{IO} current sunk must always respect the absolute maximum rating specified in *Section Table 61.* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
4. Data guaranteed by design, not tested in production.
5. 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 the RESET pin with a duration below $t_{h(RSTL)in}$ can be ignored.
6. The reset network (the resistor and two capacitors) protects the device against parasitic resets, especially in noisy environments.

Figure 80. $\overline{\text{RESET}}$ pin protection(1),(2),(3),(4)



1. The reset network protects the device against parasitic resets.
2. 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 (watchdog).
3. 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 11.9.1*. Otherwise the reset will not be taken into account internally.
4. Because the reset circuit is designed to allow the internal 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 or external reset circuit for example) is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in *Section Table 61. on page 116*.

11.9.2 ICCSEL pin

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Table 81. CCSEL pin characteristics

Symbol	Parameter	Conditions	Min	Max	Unit
V_{IL}	Input low level voltage(1)		V_{SS}	$0.3V_{DD}$	V
V_{IH}	Input high level voltage(1)		$0.7V_{DD}$	V_{DD}	
I_L	Input leakage current	$V_{IN} = V_{SS}$		±1	µA

1. Data based on design simulation and/or technology characteristics, not tested in production.

11.10 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...).

Data based on design simulation and/or characterization results, not tested in production.

Table 82. 16-bit timer

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 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

11.11 Communication interface characteristics

11.11.1 SPI - serial peripheral interface

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified. Data based on design simulation and/or characterization results, not tested in production.

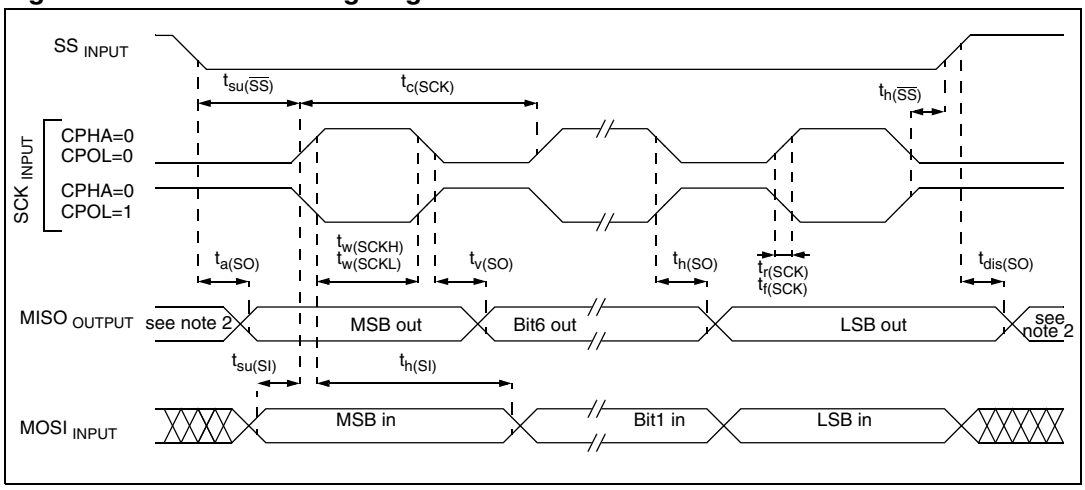
When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration. Refer to I/O port characteristics for more details on the input/output alternate function characteristics ($\overline{SS}$, SCK, MOSI, MISO).

Table 83. SPI characteristics

Symbol	Parameter	Conditions	Min	Max	Unit
f_{SCK} $1/t_{c(SCK)}$	SPI clock frequency	Master f_{CPU} = 8 MHz	$f_{CPU}/128$ 0.0625	$f_{CPU}/4$ 2	MHz
		Slave f_{CPU} = 8 MHz	0	$f_{CPU}/2$ 4	
$t_{r(SCK)}$ $t_{f(SCK)}$	SPI clock rise and fall time		see I/O port pin description		
$t_{su(\overline{SS})}$	$\overline{SS}$ setup time	Slave	t_{CPU} + 50(1)		ns
$t_{h(\overline{SS})}$	$\overline{SS}$ hold time	Slave	120		
$t_{w(SCKH)}$ $t_{w(SCKL)}$	SCK high and low time	Master Slave	100 90		
$t_{su(MI)}$ $t_{su(SI)}$	Data input setup time	Master Slave	100 100		
$t_{h(MI)}$ $t_{h(SI)}$	Data input hold time	Master Slave	100 100		
$t_{a(SO)}$	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)		90	
$t_{h(SO)}$	Data output hold time		0		
$t_{v(MO)}$	Data output valid time	Master (before capture edge)		120	
$t_{h(MO)}$	Data output hold time		0		

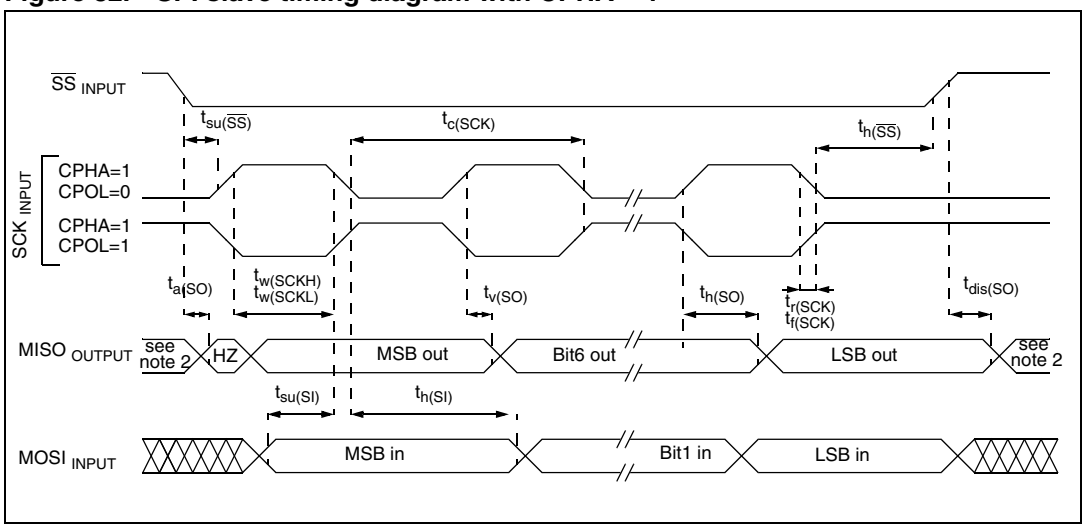
1. Depends on f_{CPU}. For example, if f_{CPU} = 8 MHz, then t_{CPU} = 1 / f_{CPU} = 125 ns and $t_{su(\overline{SS})}$ = 175 ns.

Figure 81. SPI slave timing diagram with CPHA = 0(1)



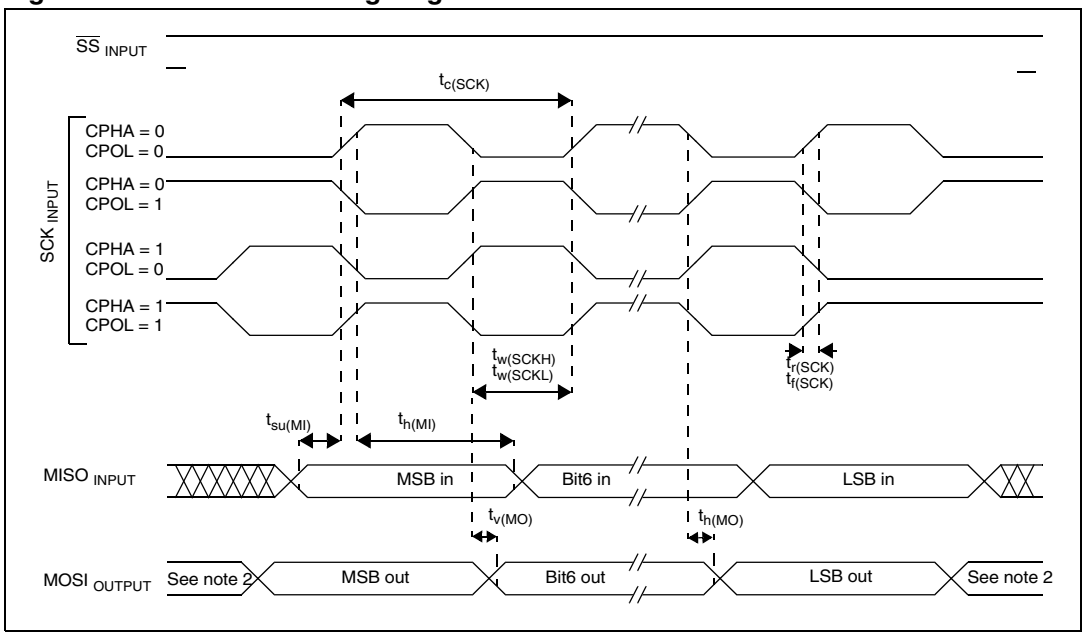
1. Measurement points are done at CMOS levels: $0.3V_{DD}$ and $0.7V_{DD}$.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

Figure 82. SPI slave timing diagram with CPHA = 1(1)



1. Measurement points are done at CMOS levels: $0.3V_{DD}$ and $0.7V_{DD}$.
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.

Figure 83. SPI master timing diagram(1)



1. Measurement points are done at CMOS levels: $0.3V_{DD}$ and $0.7V_{DD}$.
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.

11.12 10-bit ADC characteristics

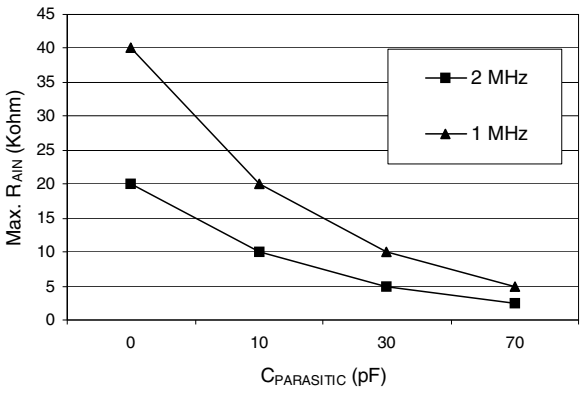
Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Table 84. 10-bit ADC characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{ADC}	ADC clock frequency	$f_{CPU} \leq 4$ MHz	0.4		**4**	MHz
V_{AREF}	Analog reference voltage	$0.7V_{DD} \leq V_{AREF} \leq V_{DD}$	3.8		V_{DD}	V
V_{AIN}	Conversion voltage range (1)		V_{SSA}		V_{AREF}	
I_{lkg}	Positive input leakage current for analog input	$-40\ °C \leq T_A \leq +85\ °C$			±250	nA
		$+85\ °C \leq T_A \leq +125\ °C$			±1	µA
	Negative input leakage current on robust analog pins	$V_{IN} < V_{SS}$, $\lvert I_{IN} \rvert < 400$ µA on adjacent robust analog pin		5	6	µA
R_{AIN}	External input impedance				see *Figure 84* and *Figure 85*	kΩ
C_{AIN}	External capacitor on analog input					pF
f_{AIN}	Variation frequency of analog input signal					Hz
C_{ADC}	Internal sample and hold capacitor			12		pF
t_{ADC}	Conversion time (Sample+Hold) f_{CPU} = 4 MHz, MAX_SPEED = 1, SPEED = 1 f_{ADC} = 4 MHz		3.75			µs
t_{ADC}	– No of sample capacitor loading cycles – No. of Hold conversion cycles		4 11			$1/f_{ADC}$

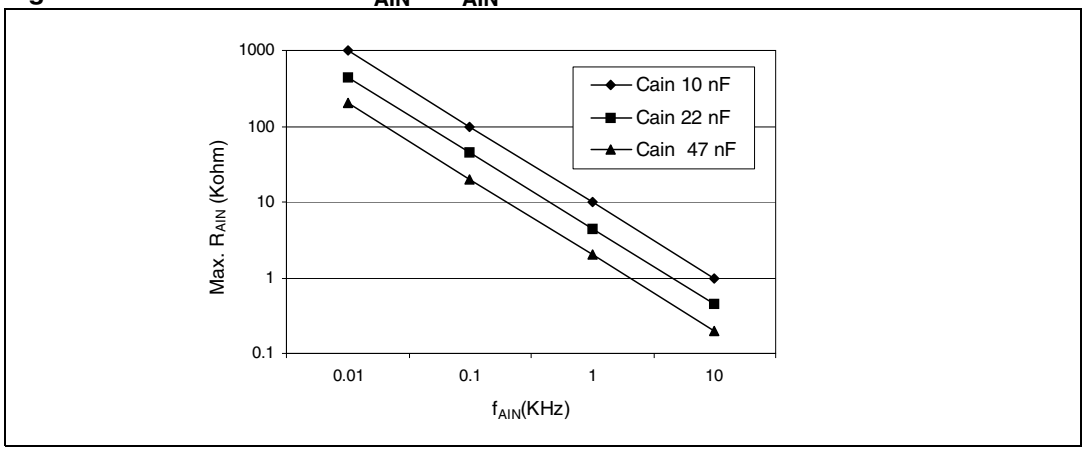
1. *Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than 10 kΩ). Data based on characterization results, not tested in production.*

Figure 84. R_{AIN} max. vs. f_{ADC} with C_{AIN} = 0 pF(1)



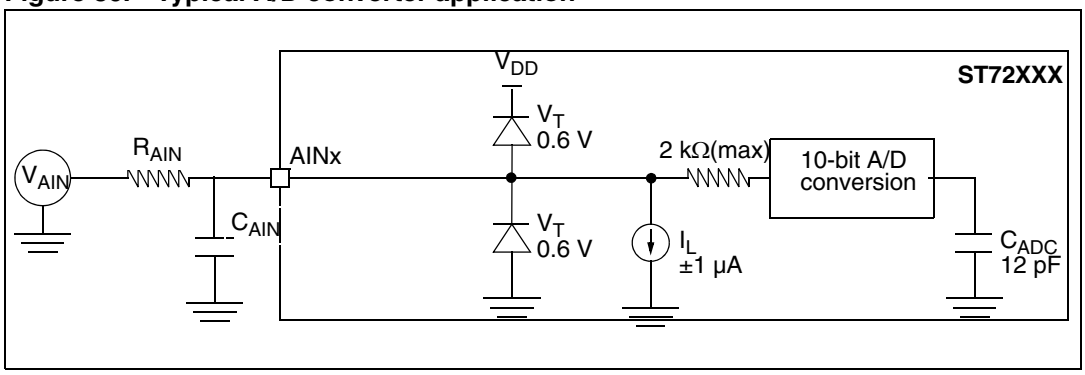
1. $C_{PARASITIC}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high $C_{PARASITIC}$ value will downgrade conversion accuracy. To remedy this, f_{ADC} should be reduced.

Figure 85. Recommended C_{AIN} & R_{AIN} values(1)



1. This graph shows that depending on the input signal variation (f_{AIN}), C_{AIN} can be increased for stabilization time and decreased to allow the use of a larger serial resistor (R_{AIN}).

Figure 86. Typical A/D converter application



11.12.1 Analog power supply and reference pins

Depending on the MCU pin count, the package may feature separate V_{AREF} and V_{SSA} analog power supply pins. These pins supply power to the A/D converter cell and function as the high and low reference voltages for the conversion. In some packages, V_{AREF} and V_{SSA} pins are not available (refer to *Section 2 on page 12*). In this case the analog supply and reference pads are internally bonded to the V_{DD} and V_{SS} pins.

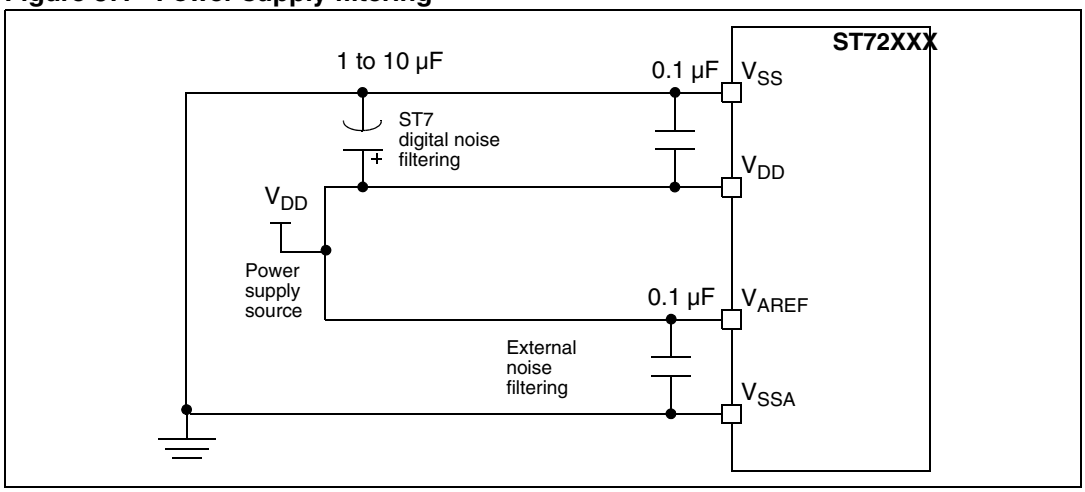
Separation of the digital and analog power pins allow board designers to improve A/D performance. Conversion accuracy can be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines (see *Section 11.12.2: General PCB design guidelines*).

11.12.2 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.

- Use separate digital and analog planes. The analog ground plane should be connected to the digital ground plane via a single point on the PCB.
- Filter power to the analog power planes. It is recommended to connect capacitors, with good high frequency characteristics, between the power and ground lines, placing 0.1 µF and optionally, if needed 10 pF capacitors as close as possible to the ST7 power supply pins and a 1 to 10 µF capacitor close to the power source (see *Figure 87*).
- The analog and digital power supplies should be connected in a star network. Do not use a resistor, as V_{AREF} is used as a reference voltage by the A/D converter and any resistance would cause a voltage drop and a loss of accuracy.
- 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.

Figure 87. Power supply filtering



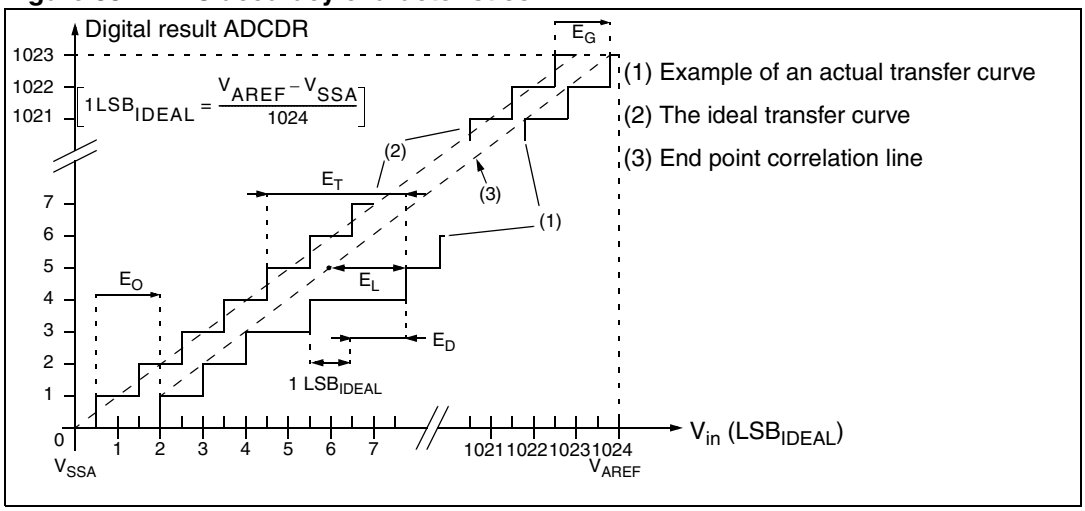
11.12.3 ADC accuracy

Table 85. ADC accuracy characteristics

Symbol	Parameter	Conditions	V_{DD} = 3 V		V_{DD} = 5 V		Unit
			Typ	Max	Typ	Max	
$\|E_T\|$	Total unadjusted error(1)		3	4	3	4	LSB
$\|E_O\|$	Offset error(1)		2	3	2	3	
$\|E_G\|$	Gain Error(1)		0.5	3	0.5	3	
$\|E_D\|$	Differential linearity error (1)	CPU in run mode @ f_{ADC} 4 MHz.	1.5	3	1	2	
$\|E_L\|$	Integral linearity error (1)	CPU in run mode @ f_{ADC} 4 MHz.	1.5	3	1	2	

1. Design target values.

Figure 88. ADC accuracy characteristics(1)

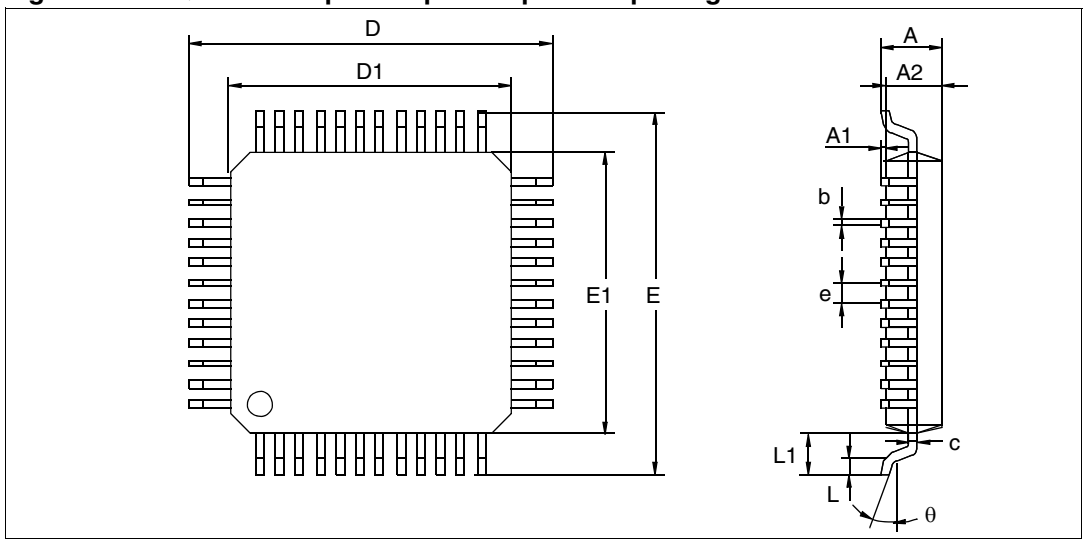


1. E_T = total unadjusted error: maximum deviation between the actual and the ideal transfer curves.
E_O = offset error: deviation between the first actual transition and the first ideal one.
E_G = gain error: deviation between the last ideal transition and the last actual one.
E_D = differential linearity error: maximum deviation between actual steps and the ideal one.
E_L = integral linearity error: maximum deviation between any actual transition and the end point correlation line.

12 Package characteristics

12.1 Package mechanical data

Figure 89. LQFP48 – 48-pin low profile quad flat package outline



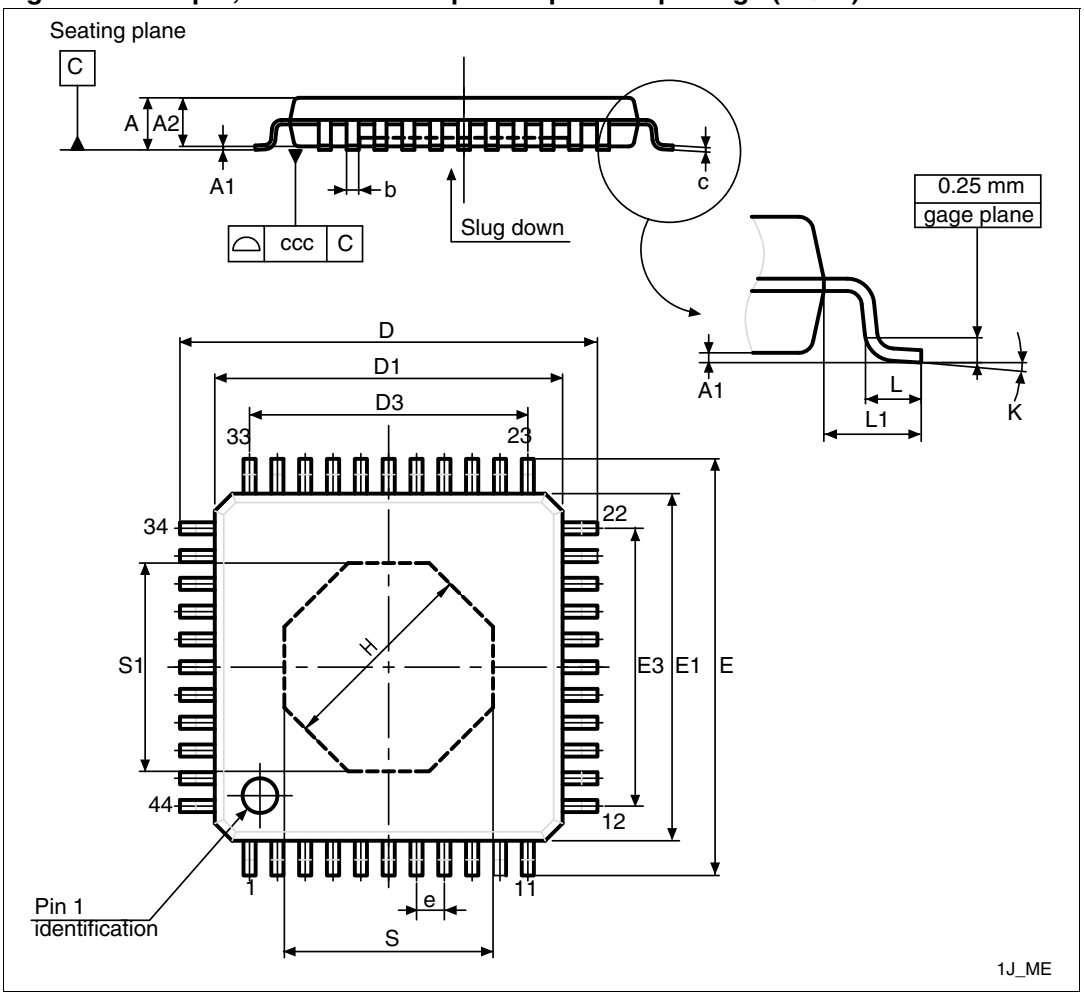
1. Drawing is not to scale.

Table 86. LQFP48 – 48-pin low profile quad flat package mechanical data

Symbol	millimeters			inches(1)		
	Typ	Min	Max	Typ	Min	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.17	0.22	0.27	0.0067	0.0087	0.0106
C	0.09		0.20	0.0035		0.0079
D		9.00			0.3543	
D1		7.00			0.2756	
E		9.00			0.3543	
E1		7.00			0.2756	
e		0.50			0.0197	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
N	Number of pins					
	48					

1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 90. 44-pin, 10 x 10 mm low-profile quad flat package (LQFP) outline



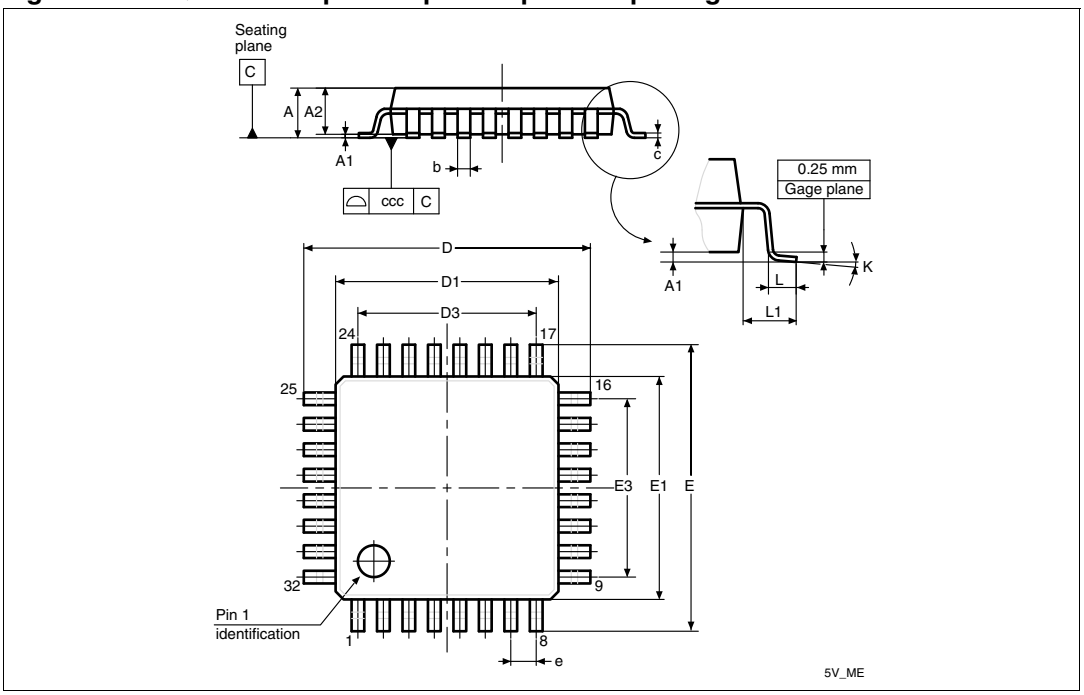
1. Drawing is not to scale.

Table 87. 44-pin, low-profile quad flat package (LQFP) mechanical data

Symbol	millimeters			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.6			0.063
A1	0.05		0.15	0.002		0.0059
A2	1.35	1.4	1.45	0.0531	0.0551	0.0571
b	0.3	0.37	0.45	0.0118	0.0146	0.0177
c	0.09		0.2	0.0035		0.0079
D	11.8	12	12.2	0.4646	0.4724	0.4803
D1	9.8	10	10.2	0.3858	0.3937	0.4016
D3		8			0.315	
E	11.8	12	12.2	0.4646	0.4724	0.4803
E1	9.8	10	10.2	0.3858	0.3937	0.4016
E3		8			0.315	
e		0.8			0.0315	
H		5.89			0.2319	
L	0.45	0.6	0.75	0.0177	0.0236	0.0295
L1		1			0.0394	
S	6			0.2362		
S1	6			0.2362		
K	0.0°	3.5°	7.0°	0.0°	3.5°	7.0°
ccc		0.1			0.0039	

1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 91. LQFP32 – 32-pin low-profile quad flat package outline



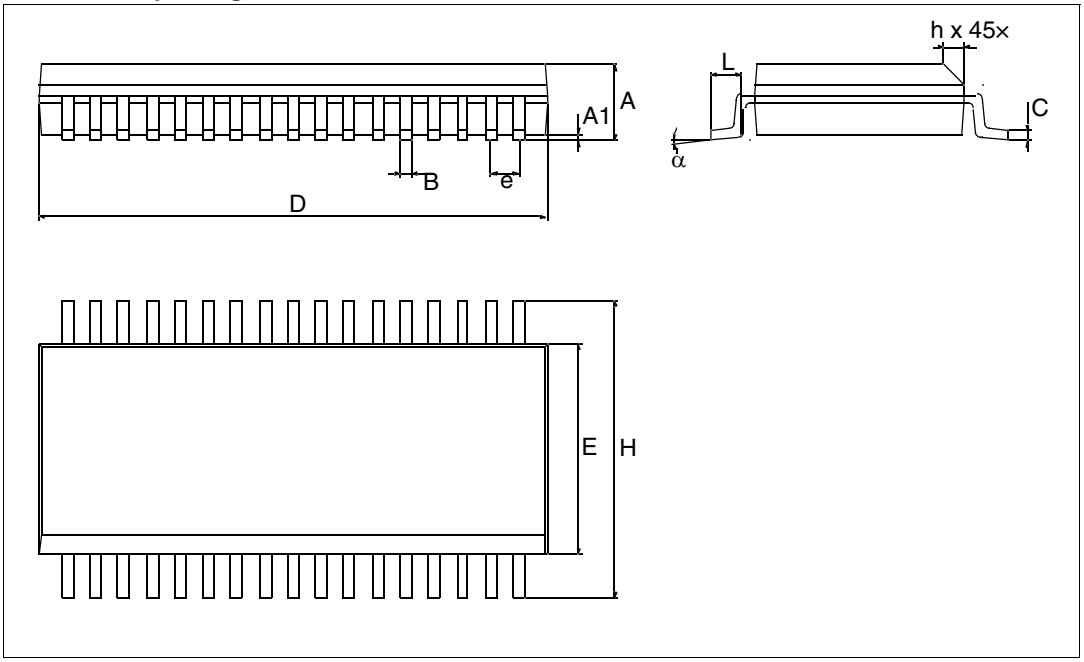
1. Drawing is not to scale.

Table 88. LQFP32 – 32-pin low-profile quad flat package mechanical data

Symbol	millimeters			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.6			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.4	1.45	0.0531	0.0551	0.0571
b	0.3	0.37	0.45	0.0118	0.0146	0.0177
c	0.09		0.2	0.0035		0.0079
D	8.8	9	9.2	0.3465	0.3543	0.3622
D1	6.8	7	7.2	0.2677	0.2756	0.2835
D3		5.6			0.2205	
E	8.8	9	9.2	0.3465	0.3543	0.3622
E1	6.8	7	7.2	0.2677	0.2756	0.2835
E3		5.6			0.2205	
e		0.8			0.0315	
L	0.45	0.6	0.75	0.0177	0.0236	0.0295
L1		1			0.0394	
K	0°	3.5°	7°	0°	3.5°	7°
ccc		0.1			0.0039	

1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 92. SO34 – 34-pin plastic small outline package, shrink 300-mil width, package outline



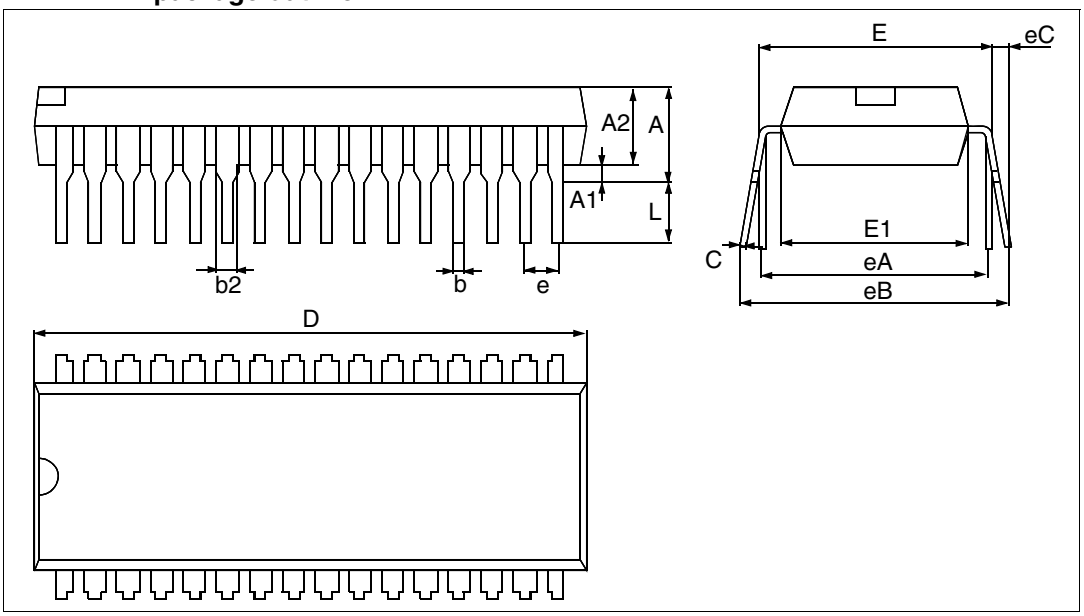
1. Drawing is not to scale.

Table 89. SO34 – 34-pin plastic small outline package, shrink 300-mil width, mechanical data

Symbol	millimeters			inches[1]		
	Typ	Min	Max	Typ	Min	Max
A	2.464		2.642	0.0970		0.1040
A1	0.127		0.292	0.0050		0.0115
B	0.356		0.483	0.0140		0.0190
C	0.231		0.318	0.0091		0.0125
D	17.729		18.059	0.6980		0.7110
E	7.417		7.595	0.2920		0.2990
e		1.016			0.040	
H	10.160		10.414	0.4000		0.4100
h	0.635		0.737	0.0250		0.0290
α	0°		8°	0°		8°
L	0.610		1.016	0.0240		0.0400
N	Number of pins					
	34					

1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 93. SDIP32 – 32-pin plastic dual in-line package, shrink 400-mil width, package outline



1. Drawing is not to scale.

Table 90. SDIP32 – 32-pin plastic dual in-line package, shrink 400-mil width, mechanical data

Symbol	millimeters			inches[1]		
	Typ	Min	Max	Typ	Min	Max
A	3.56	3.76	5.08	0.1402	0.1480	0.2000
A1	0.51			0.0201		
A2	3.05	3.56	4.57	0.1201	0.1402	0.1799
b	0.36	0.46	0.58	0.0142	0.0181	0.0228
b1	0.76	1.02	1.40	0.0299	0.0402	0.0551
C	0.20	0.25	0.36	0.0079	0.0098	0.0142
D	27.43		28.45	1.0799		1.1201
E	9.91	10.41	11.05	0.3902	0.4098	0.4350
E1	7.62	8.89	9.40	0.3000	0.3500	0.3701
e		1.78			0.0701	
eA		10.16			0.4000	
eB			12.70			0.500
eC			1.40			0.055
L	2.54	3.05	3.81	0.100	0.120	0.150
N	Number of pins					
	32					

1. Values in inches are converted from mm and rounded to 4 decimal digits.

12.2 Thermal characteristics

Table 91. Package thermal characteristics

Symbol	Ratings		Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient)	LQFP48 7x7 LQFP44 10x10 LQFP32 7x7 SDIP32 200 mil	80 52 70 50	°C/W
P_D	Power dissipation (1)		500	mW
T_{Jmax}	Maximum junction temperature(2)		150	°C

1. The maximum power dissipation is obtained from the formula $P_D = (T_J - T_A) / R_{thJA}$. The power dissipation of an application can be defined by the user with 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 depending on the ports used in the application.
2. The maximum chip-junction temperature is based on technology characteristics.

12.3 Soldering information

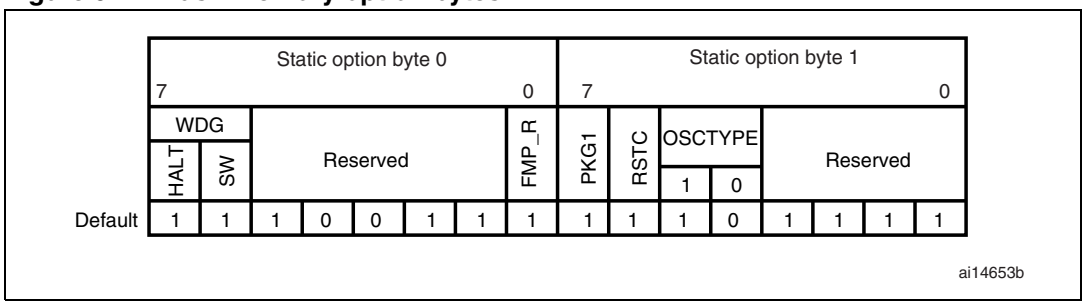
In order to meet environmental requirements, ST offers these devices in ECOPACK® packages. These packages have a lead-free second level interconnect. The category of second-level interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label.

13 ST72323 device configuration and ordering information

The device is available for development in a user programmable compatible version (ST72F324B Flash). Flash devices are shipped to customers with a default content (FFh), while ROM 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 devices are factory-configured.

13.1 Flash memory option bytes (ST72F324B compatible superset)

Figure 94. Flash memory option bytes



The option bytes allows the hardware configuration of the microcontroller to be selected. They 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. To program directly the Flash devices using ICP, Flash devices are shipped to customers with the internal RC clock source. In masked ROM devices, the option bytes are fixed in hardware by the ROM code (see option list).

Option byte 0

OPT7 = **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

OPT6 = **WDG SW** *Hardware or software watchdog*

This option bit selects the watchdog type.

0: Hardware (watchdog always enabled)

1: Software (watchdog to be enabled by software)

OPT5:1 = Reserved, must be kept at default value.

OPT0 = **FMP_R** *Flash memory read-out protection*

Read-out protection, when selected, provides a protection against Program Memory content extraction and against write access to Flash memory.

Erasing the option bytes when the FMP_R option is selected causes the whole user memory to be erased first, and the device can be reprogrammed. Refer to the ST72F324B datasheet and the ST7 Flash Programming Reference Manual for more details.

0: Read-out protection enabled

1: Read-out protection disabled

Option byte 1

OPT7 = **PKG1** *Pin package selection bit*

This option bit selects the package as shown in *Table 92*.

Table 92. Package selection (OPT7)

Version	Selected package	PKG1
J	LQFP48/LQFP44	1
K	LQFP32 / SDIP32 / SO34	0

Note: On the chip, each I/O port has 8 pads. Pads that are not bonded to external pins are in input pull-up configuration after reset. The configuration of these pads must be kept at reset state to avoid added current consumption.

OPT6 = RSTC *RESET clock cycle selection*

This option bit selects the number of CPU cycles applied during the RESET phase and when exiting Halt mode. For resonator oscillators, it is advised to select 4096 due to the long crystal stabilization time.

0: Reset phase with 4096 CPU cycles

1: Reset phase with 256 CPU cycles

OPT5:4 = **OSCTYPE[1:0]** *Oscillator Type*

These option bits select the ST7 main clock source type as shown in *Table 93*.

Table 93. Clock source selection

Clock source	OSCTYPE	
	1	0
Resonator oscillator	0	0
Reserved	0	1
Internal RC oscillator(1)	1	0
External source	1	1

1. The RCosc frequency in ST72F324B Flash devices is centered on 3.5 MHz. (f_{CPU}=1.75MHz). For ROM ST72323 devices refer to *Section 11.5.2*.

OPT3:0 = Reserved, must be kept at default value..

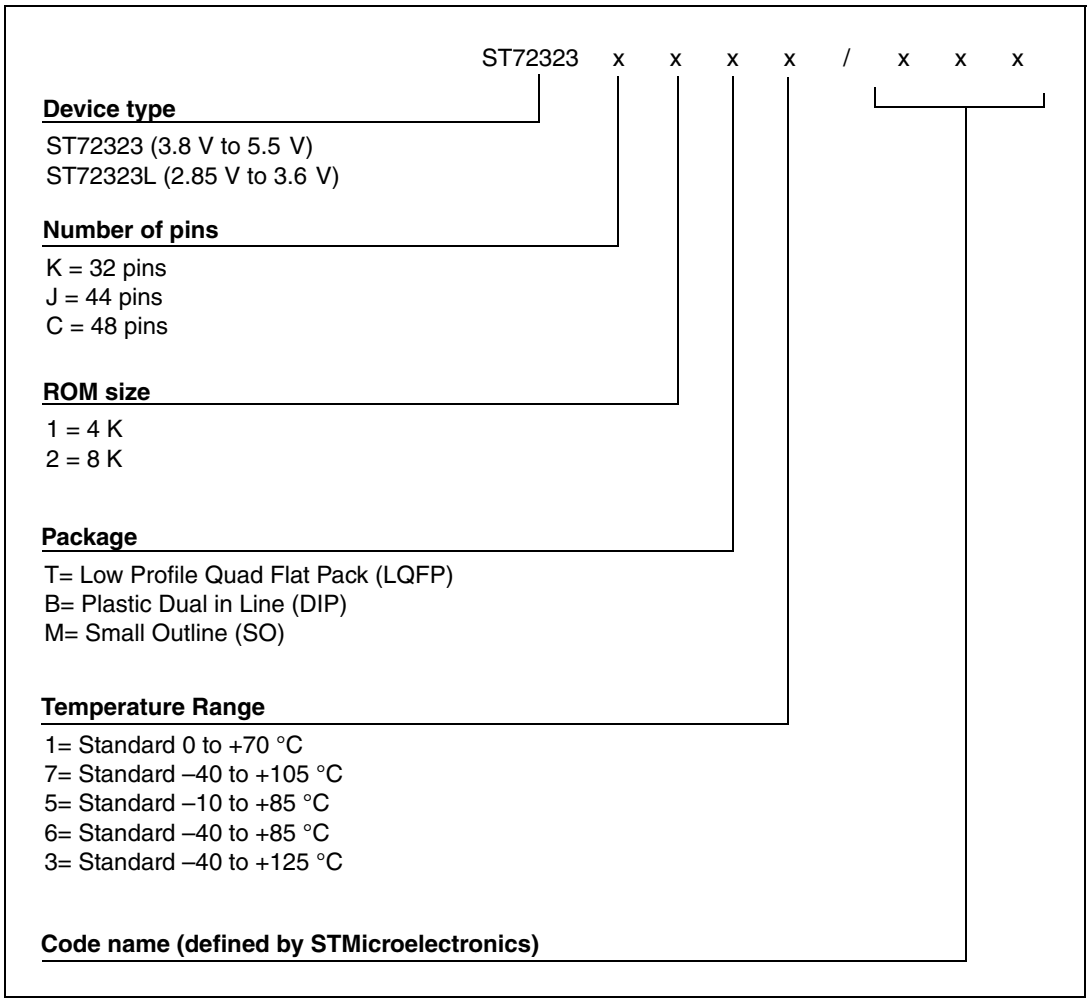
13.2 ROM device ordering information and transfer of customer code

ROM devices can be ordered in any combination of memory size and temperature range with the types given in *Figure 95* and by completing the option list on the next page. ROM customer code is made up of the ROM contents and the list of the selected options (if any). The ROM contents are to be sent with the S19 hexadecimal file generated by the development tool. All unused bytes must be set to FFh.

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.

Figure 95. Ordering information scheme



Note: All packages may not be available in some devices. Contact your ST sales office for information.

Figure 96. ST72323 5 V microcontroller option list

ST72323 (5V) MICROCONTROLLER OPTION LIST
(Last updated October 2007)

Customer: . . . . . .
Address: . . . . . .
. . . . . .
Contact: . . . . . .
Phone No: . . . . . .
Reference/ROM Code* : . . . . . .
*The ROM code name is assigned by STMicroelectronics.
ROM code must be sent in .S19 format. .Hex extension cannot be processed.

Device Type/Memory Size/Package (check only one option):

ROM DEVICE:	4K	8K
LQFP32:	[ ] ST72323K1T	[ ] ST72323K2T
LQFP44:	[ ] ST72323J1T	[ ] ST72323J2T
LQFP48 :	[ ] ST72323C1T	[ ] ST72323C2T
DIP32:	[ ] ST72323K1B	[ ] ST72323J2B
SO34:	[ ] ST72323J1M	[ ] ST72323J2M
DIE FORM:	**4K**	**8K**
32-pin:	[ ]	[ ]
44-pin:	[ ]	[ ]

Conditioning (check only one option):

Packaged Product	Die Product (dice tested at 25°C only)
LQFP: [ ] Tape & Reel [ ] Tray	[ ] Tape & Reel
DIP: [ ] Tube	[ ] Inked wafer
	[ ] Sawn wafer on sticky foil
	[ ] Waffle pack

Power Supply Range: [ ] 3.8 to 5.5V
Temp. Range (do not check for die product).

[ ]	0°C to +70°C
[ ]	-10°C to +85°C
[ ]	-40°C to +85°C
[ ]	-40°C to +105°C
[ ]	-40°C to +125°C

Special Marking: [ ] No [ ] Yes "_ _ _ _ _ _ _ _ _ _ " (LQFP32 & LQFP48 7 char., other pkg. 10 char. max)
Authorized characters are letters, digits, '.', '-', '/' and spaces only.

Clock Source Selection:
[ ] Resonator:
[ ] Internal RC:
[ ] External Clock

Reset Delay [ ] 256 Cycles [ ] 4096 Cycles

Watchdog Selection: [ ] Software Activation [ ] Hardware Activation
Watchdog Reset on Halt: [ ] Reset [ ] No Reset

Readout Protection: [ ] Disabled [ ] Enabled

Date . . . . . .
Signature . . . . . .

Figure 97. ST72323 3 V microcontroller option list

ST72323 (3V) MICROCONTROLLER OPTION LIST
(Last updated October 2007)

Customer: . . . . . .
Address: . . . . . .
. . . . . .
Contact: . . . . . .
Phone No: . . . . . .
Reference/ROM Code* : . . . . . .
*The ROM code name is assigned by STMicroelectronics.
ROM code must be sent in .S19 format. .Hex extension cannot be processed.

Device Type/Memory Size/Package (check only one option):

ROM DEVICE:	4K	8K
LQFP32:	[ ] ST72323LK1T	[ ] ST72323LK2T
LQFP44:	[ ] ST72323LJ1T	[ ] ST72323LJ2T
LQFP48 :	[ ] ST72323LC1T	[ ] ST72323LC2T
DIP32:	[ ] ST72323LK1B	[ ] ST72323LJ2B
SO34:	[ ] ST72323LJ1M	[ ] ST72323LJ2M
DIE FORM:	4K	8K
32-pin:	[ ]	[ ]
44-pin:	[ ]	[ ]

Conditioning (check only one option):

Packaged Product	Die Product (dice tested at 25°C only)
LQFP: [ ] Tape & Reel [ ] Tray	[ ] Tape & Reel
DIP: [ ] Tube	[ ] Inked wafer
	[ ] Sawn wafer on sticky foil
	[ ] Waffle pack

Power Supply Range: [ ] 2.85 to 3.6V
Temp. Range (do not check for die product).

[ ] | 0°C to +70°C
[ ] | -10°C to +85°C
[ ] | -40°C to +85°C

Special Marking: [ ] No [ ] Yes "_ _ _ _ _ _ _ _ _ _ " (LQFP32 & LQFP48 7 char., other pkg. 10 char. max)
Authorized characters are letters, digits, '.', '-', '/' and spaces only.

Clock Source Selection:
[ ] Resonator:
[ ] Internal RC:
[ ] External Clock

Reset Delay [ ] 256 Cycles [ ] 4096 Cycles

Watchdog Selection: [ ] Software Activation [ ] Hardware Activation
Watchdog Reset on Halt: [ ] Reset [ ] No Reset

Readout Protection: [ ] Disabled [ ] Enabled

Date . . . . . .
Signature . . . . . .

13.3 Ordering information for compatible Flash devices

Table 94. ST72F324B compatible Flash memory order codes

Part number	Package	Flash memory (Kbytes)	Temp. range
ST72F324BK2T6	LQFP32	8	–40 °C +85 °C
ST72F324BK2T5		8	–10 °C +85 °C
ST72F324BK2T3		8	–40 °C +125 °C
ST72F324BK2B6	SDIP32	8	–40 °C +85 °C
ST72F324BK2B5		8	–10 °C +85 °C
ST72F324BK2M6	SO34	8	–40 °C +85 °C
ST72F324BK2M5		8	–10 °C +85 °C
ST72F324BJ2T6	LQFP44	8	–40 °C +85 °C
ST72F324BJ2T5		8	–10 °C +85 °C
ST72F324BJ2T3		8	–40 °C +125 °C
ST72F324BC2T6	LQFP48	8	–40 °C +85°C
ST72F324BC2T5		8	–10°C +85 °C
ST72F324BC2T3		8	–40 °C +125 °C

13.4 Development tools

Development tools for the ST7 microcontrollers include a complete range of hardware systems and software tools from STMicroelectronics and third-party tool suppliers. The range of tools includes solutions to help you evaluate microcontroller peripherals, develop and debug your application, and program your microcontrollers.

13.4.1 Evaluation tools and starter kits

ST offers complete, affordable **starter kits** and full-featured **evaluation boards** that allow you to evaluate microcontroller features and quickly start developing ST7 applications. Starter kits are complete, affordable hardware/software tool packages that include features and samples to help you quickly start developing your application. ST evaluation boards are open-design, embedded systems, which are developed and documented to serve as references for your application design. They include sample application software to help you demonstrate, learn about and implement your ST7's features.

13.4.2 Development and debugging tools

Application development for ST7 is supported by fully optimizing **C Compilers** and the **ST7 Assembler-Linker** toolchain, which are all seamlessly integrated in the ST7 integrated development environments in order to facilitate the debugging and fine-tuning of your application. The Cosmic C Compiler is available in a free version that outputs up to 16K of code.

The range of hardware tools includes full-featured **ST7-EMU3 series emulators** and the low-cost **RLink** in-circuit debugger/programmer. These tools are supported by the **ST7 Toolset** from STMicroelectronics, which includes the STVD7 integrated development environment (IDE) with high-level language debugger, editor, project manager and integrated programming interface.

13.4.3 Programming tools

During the development cycle, the **ST7-EMU3 series emulators** and the **RLink** provide in-circuit programming capability for programming the Flash microcontroller on your application board.

In addition ST provides dedicated programming tools including the **ST7-EPB programming boards**, which include all the sockets required to program any of the devices in a specific ST7 subfamily.

For production programming of ST7 devices, ST's third-party tool partners also provide a complete range of gang and automated programming solutions, which are ready to integrate into your production environment.

13.4.4 Order codes for ST72324B development tools

Table 95. STMicroelectronics development tools

Supported products	Emulation				Programming
	ST7 DVP3 series		ST7 EMU3 series		ICC socket board
	Emulator	Connection kit	Emulator	Active probe & T.E.B.	
ST72324BJ, ST72F324BJ	ST7MDT20-DVP3	ST7MDT20-T44/DVP	ST7MDT20J-EMU3	ST7MDT20J-TEB	ST7SB20J/xx(1)
ST72324BK, ST72F324BK	ST7MDT20-DVP3	ST7MDT20-T32/DVP			

1. Add suffix /EU, /UK, /US for the power supply of your region.

14 Known limitations

14.1 Unexpected reset fetch

If an interrupt request occurs while a "POP CC" instruction is executed, the interrupt controller does not recognize the source of the interrupt and, by default, passes the RESET vector address to the CPU.

Workaround

To solve this issue, a "POP CC" instruction must always be preceded by a "SIM" instruction.

14.2 Clearing active interrupts outside interrupt routine

When an active interrupt request occurs at the same time as the related flag is being cleared, an unwanted reset may occur.

Note: Clearing the related interrupt mask will not generate an unwanted reset

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, i.e. when:

- The interrupt flag is cleared within its own interrupt routine
- The interrupt flag is cleared within any interrupt routine
- The interrupt flag is cleared in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

Perform SIM and RIM operation before and after resetting an active interrupt request.

Example:

```
SIM
reset interrupt flag
RIM
```

Nested interrupt context

The symptom does not occur when the interrupts are handled normally, i.e. when:

- The interrupt flag is cleared within its own interrupt routine
- The interrupt flag is cleared within any interrupt routine with higher or identical priority level
- The interrupt flag is cleared in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

```
PUSH CC
SIM
reset interrupt flag
POP CC
```

14.3 External interrupt missed

To avoid any risk if generating a parasitic interrupt, the edge detector is automatically disabled for one clock cycle during an access to either DDR and OR. Any input signal edge during this period will not be detected and will not generate an interrupt.

This case can typically occur if the application refreshes the port configuration registers at intervals during runtime.

Workaround

The workaround is based on software checking the level on the interrupt pin before and after writing to the PxOR or PxDDR registers. If there is a level change (depending on the sensitivity programmed for this pin) the interrupt routine is invoked using the call instruction with three extra PUSH instructions before executing the interrupt routine (this is to make the call compatible with the IRET instruction at the end of the interrupt service routine).

But detection of the level change does not make sure that edge occurs during the critical 1 cycle duration and the interrupt has been missed. This may lead to occurrence of same interrupt twice (one hardware and another with software call).

To avoid this, a semaphore is set to '1' before checking the level change. The semaphore is changed to level '0' inside the interrupt routine. When a level change is detected, the semaphore status is checked and if it is '1' this means that the last interrupt has been missed. In this case, the interrupt routine is invoked with the call instruction.

There is another possible case i.e. if writing to PxOR or PxDDR is done with global interrupts disabled (interrupt mask bit set). In this case, the semaphore is changed to '1' when the level change is detected. Detecting a missed interrupt is done after the global interrupts are enabled (interrupt mask bit reset) and by checking the status of the semaphore. If it is '1' this means that the last interrupt was missed and the interrupt routine is invoked with the call instruction.

To implement the workaround, the following software sequence is to be followed for writing into the PxOR/PxDDR registers. The example is for Port PF1 with falling edge interrupt sensitivity. The software sequence is given for both cases (global interrupt disabled/enabled).

Case 1: Writing to PxOR or PxDDR with global interrupts enabled

```
LD A,#01
LD sema,A          ; set the semaphore to '1'
LD A,PFDR
AND A,#02
LD X,A             ; store the level before writing to PxOR/PxDDR
LD A,#$90
LD PFDDR,A         ; Write to PFDDR
LD A,#$ff
LD PFOR,A          ; Write to PFOR
LD A,PFDR
```

```
AND A,#02
LD Y,A              ; store the level after writing to PxOR/PxDDR
LD A,X              ; check for falling edge
cp A,#02
jrne OUT
TNZ Y
jrne OUT
LD A,sema           ; check the semaphore status if edge is detected
CP A,#01
jrne OUT
call call_routine   ; call the interrupt routine
OUT:LD A,#00
LD sema,A
.call_routine       ; entry to call_routine
PUSH A
PUSH X
PUSH CC
.ext1_rt            ; entry to interrupt routine
LD A,#00
LD sema,A
IRET
```

Case 2: Writing to PxOR or PxDDR with global interrupts disabled

```
SIM                 ; set the interrupt mask
LD A,PFDR
AND A,#$02
LD X,A              ; store the level before writing to PxOR/PxDDR
LD A,#$90
LD PFDDR,A          ; Write into PFDDR
LD A,#$ff
LD PFOR,A           ; Write to PFOR
LD A,PFDR
AND A,#$02
LD Y,A              ; store the level after writing to PxOR/PxDDR
LD A,X              ; check for falling edge
cp A,#$02
jrne OUT
TNZ Y
jrne OUT
LD A,#$01
LD sema,A           ; set the semaphore to '1' if edge is detected
RIM                 ; reset the interrupt mask
LD A,sema           ; check the semaphore status
CP A,#$01
jrne OUT
call call_routine   ; call the interrupt routine
RIM
OUT:                RIM
JP while_loop
.call_routine       ; entry to call_routine
PUSH A
PUSH X
PUSH CC
.ext1_rt            ; entry to interrupt routine
LD A,#$00
LD sema,A
IRET
```

14.4 16-bit timer

14.4.1 PWM mode

In PWM mode, the first PWM pulse is missed after writing the value FFFCh in the OC1R register (OC1HR, OC1LR). It leads to either full or no PWM during a period, depending on the OLVL1 and OLVL2 settings.

14.4.2 TIMD set simultaneously with OC interrupt

If the 16-bit timer is disabled at the same time the output compare event occurs then output compare flag gets locked and cannot be cleared before the timer is enabled again.

Impact on the application: If output compare interrupt is enabled, then the output compare flag cannot be cleared in the timer interrupt routine. Consequently the interrupt service routine is called repeatedly and application get stuck which causes the watchdog reset if enabled by the application.

Workaround

Disable the timer interrupt before disabling the timer. Again while enabling, first enable the timer then the timer interrupts.

- Perform the following to disable the timer:
  - TACR1 or TBCR1 = 0x00h; // Disable the compare interrupt
  - TACSR | or TBCSR | = 0x40; // Disable the timer
- Perform the following to enable the timer again:
  - TACSR & or TBCSR &= ~0x40; // Enable the timer
  - TACR1 or TBCR1 = 0x40; // Enable the compare interrupt

14.5 ST72F324B compatible Flash devices

14.5.1 Internal RC operation

In ST72F324J and ST72F324K devices, the internal RC oscillator is centered on a different frequency from the ST72323L and ST72323 ROM devices.

15 Revision history

Table 96. Document revision history

Date	Revision	Changes
20-Apr-2005	1	Initial release
12-Nov-2007	2	Document reformatted. *Note 8 on page 17* added below *Table 1: Device pin description*. Caution paragraph added to *Section 5.2: Reset sequence manager (RSM)*, *Introduction*. *Note 3 on page 49* modified in *Section 8.2.1: Input modes*. *Figure 39: Output compare timing diagram, fTIMER =fCPU/4* modified and *Note 3 on page 74* modified. $t_{su(\overline{SS})}$, $t_{v(MO)}$ and $t_{h(MO)}$modified in *Table 83: SPI characteristics* and note added. *Figure 83: SPI master timing diagram(1)* modified. LQFP44 and LQFP32 package specifications updated (see *Section 12: Package characteristics*). *Section 12.3: Soldering information* modified. ECOPACK® text added. Option lists updated (see *Figure 96* and *Figure 97*). *Section 13.4: Development tools* modified. Added *Section 14.4: Watchdog Reset pulse on page 165* *Section 14.4.2: TIMD set simultaneously with OC interrupt* added.
05-June-2008	3	Modified *Section 11.7.3: Absolute maximum ratings (electrical sensitivity) on page 127*. Modified $t_{w(RSTL)out}$ in *Table 80: Asynchronous RESET pin characteristics on page 136*. Removed “Watchdog reset pulse” in *Section 14: Known limitations on page 161* (this limitation has been corrected from rev Z ST72323x devices).

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com