

Welcome to [E-XFL.COM](#)

### What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

### Applications of "[Embedded - Microcontrollers](#)"

#### Details

Product Status	Obsolete
Core Processor	ST7
Core Size	8-Bit
Speed	8MHz
Connectivity	SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	32
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	3.8V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	42-SDIP (0.600", 15.24mm)
Supplier Device Package	-
Purchase URL	<a href="https://www.e-xfl.com/product-detail/stmicroelectronics/st72f324j6b6">https://www.e-xfl.com/product-detail/stmicroelectronics/st72f324j6b6</a>

---

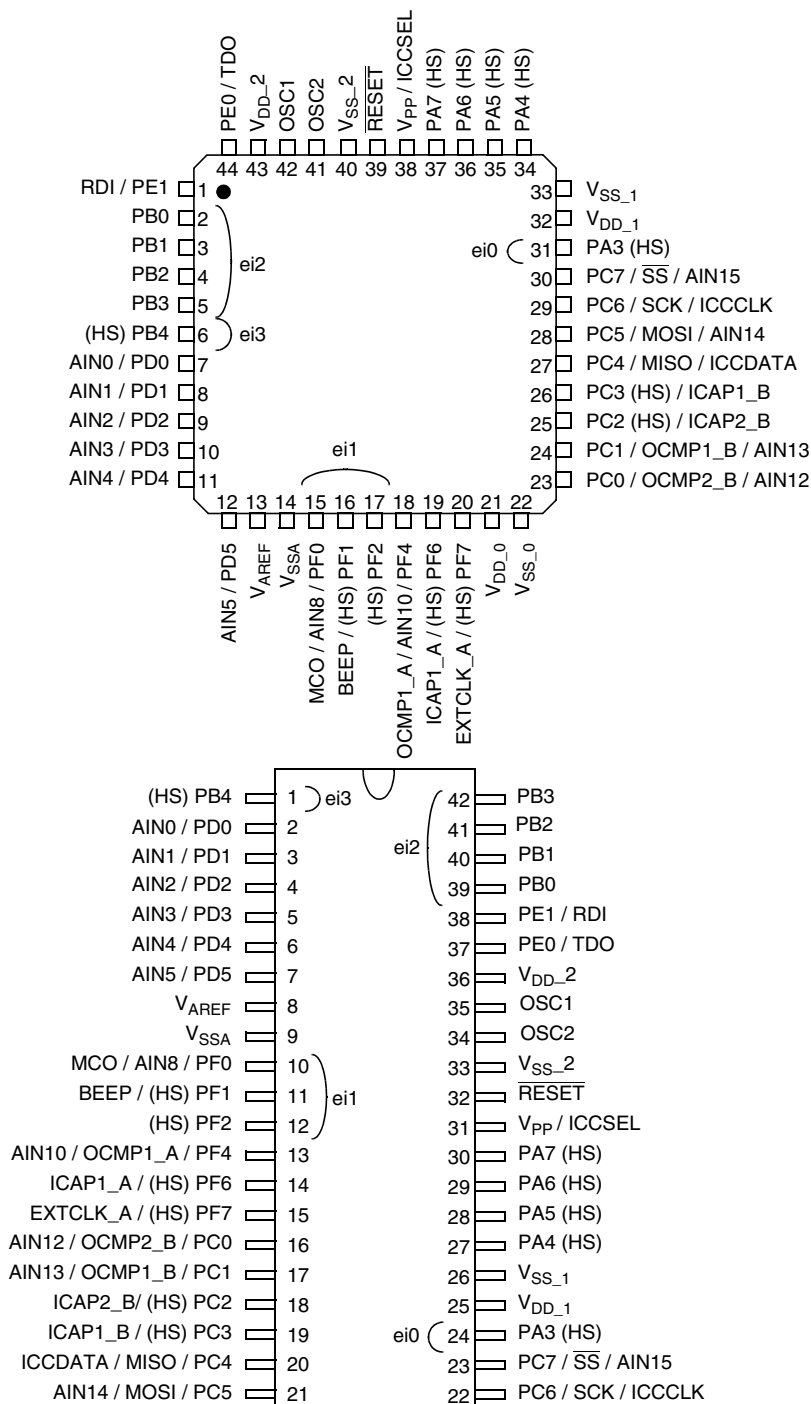
# Table of Contents

---

8.4	ACTIVE-HALT AND HALT MODES	42
8.4.1	ACTIVE-HALT MODE	42
8.4.2	HALT MODE	43
<b>9</b>	<b>I/O PORTS</b>	<b>45</b>
9.1	INTRODUCTION	45
9.2	FUNCTIONAL DESCRIPTION	45
9.2.1	Input Modes	45
9.2.2	Output Modes	45
9.2.3	Alternate Functions	45
9.3	I/O PORT IMPLEMENTATION	48
9.4	LOW POWER MODES	48
9.5	INTERRUPTS	48
9.5.1	I/O Port Implementation	49
<b>10</b>	<b>ON-CHIP PERIPHERALS</b>	<b>51</b>
10.1	WATCHDOG TIMER (WDG)	51
10.1.1	Introduction	51
10.1.2	Main Features	51
10.1.3	Functional Description	51
10.1.4	How to Program the Watchdog Timeout	52
10.1.5	Low Power Modes	54
10.1.6	Hardware Watchdog Option	54
10.1.7	Using Halt Mode with the WDG (WDGHALT option)	54
10.1.8	Interrupts	54
10.1.9	Register Description	54
10.2	MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK AND BEEPER (MCC/RTC)	56
10.2.1	Programmable CPU Clock Prescaler	56
10.2.2	Clock-out Capability	56
10.2.3	Real Time Clock Timer (RTC)	56
10.2.4	Beeper	56
10.2.5	Low Power Modes	57
10.2.6	Interrupts	57
10.2.7	Register Description	57
10.3	16-BIT TIMER	59
10.3.1	Introduction	59
10.3.2	Main Features	59
10.3.3	Functional Description	59
10.3.4	Low Power Modes	71
10.3.5	Interrupts	71
10.3.6	Summary of Timer modes	71
10.3.7	Register Description	72
10.4	SERIAL PERIPHERAL INTERFACE (SPI)	79
10.4.1	Introduction	79
10.4.2	Main Features	79
10.4.3	General Description	79
10.4.4	Clock Phase and Clock Polarity	83
10.4.5	Error Flags	84
10.4.6	Low Power Modes	86

## 2 PIN DESCRIPTION

Figure 2. 42-Pin SDIP and 44-Pin TQFP Package Pinouts



(HS) 20mA high sink capability  
eix associated external interrupt vector

Pin n°				Pin Name	Type	Level		Port						Main function (after reset)	Alternate Function	
TQFP44	SDIP42	TQFP32	SDIP32			Input	Output	Input				Output				
								float	wpu	int	ana	OD	PP			
24	17	9	12	PC1/OCMP1_B/AIN13	I/O	C <sub>T</sub>		X	X		X	X	X	Port C1	Timer B Output Compare 1	ADC Analog Input 13
25	18	10	13	PC2 (HS)/ICAP2_B	I/O	C <sub>T</sub>	HS	X	X			X	X	Port C2	Timer B Input Capture 2	
26	19	11	14	PC3 (HS)/ICAP1_B	I/O	C <sub>T</sub>	HS	X	X			X	X	Port C3	Timer B Input Capture 1	
27	20	12	15	PC4/MISO/ICCDA-TA	I/O	C <sub>T</sub>		X	X			X	X	Port C4	SPI Master In / Slave Out Data	ICC Data Input
28	21	13	16	PC5/MOSI/AIN14	I/O	C <sub>T</sub>		X	X		X	X	X	Port C5	SPI Master Out / Slave In Data	ADC Analog Input 14
29	22	14	17	PC6/SCK/ICCCLK	I/O	C <sub>T</sub>		X	X			X	X	Port C6	SPI Serial Clock	ICC Clock Output
30	23	15	18	PC7/ $\overline{SS}$ /AIN15	I/O	C <sub>T</sub>		X	X		X	X	X	Port C7	SPI Slave Select (active low)	ADC Analog Input 15
31	24	16	19	PA3 (HS)	I/O	C <sub>T</sub>	HS	X		ei0		X	X	Port A3		
32	25			V <sub>DD_1</sub>	S									Digital Main Supply Voltage		
33	26			V <sub>SS_1</sub>	S									Digital Ground Voltage		
34	27	17	20	PA4 (HS)	I/O	C <sub>T</sub>	HS	X	X			X	X	Port A4		
35	28			PA5 (HS)	I/O	C <sub>T</sub>	HS	X	X			X	X	Port A5		
36	29	18	21	PA6 (HS)	I/O	C <sub>T</sub>	HS	X				T		Port A6 <sup>1)</sup>		
37	30	19	22	PA7 (HS)	I/O	C <sub>T</sub>	HS	X				T		Port A7 <sup>1)</sup>		
38	31	20	23	V <sub>PP</sub> /ICCSEL	I									Must be tied low. In the flash programming mode, this pin acts as the programming voltage input V <sub>PP</sub> . See Section 12.10.2 for more details.		
39	32	21	24	$\overline{RESET}$	I/O	C <sub>T</sub>								Top priority non maskable interrupt.		
40	33	22	25	V <sub>SS_2</sub>	S									Digital Ground Voltage		
41	34	23	26	OSC2	O									Resonator oscillator inverter output		
42	35	24	27	OSC1	I									External clock input or Resonator oscillator inverter input		
43	36	25	28	V <sub>DD_2</sub>	S									Digital Main Supply Voltage		
44	37	26	29	PE0/TDO	I/O	C <sub>T</sub>		X	X			X	X	Port E0	SCI Transmit Data Out	
1	38	27	30	PE1/RDI	I/O	C <sub>T</sub>		X	X			X	X	Port E1	SCI Receive Data In	
2	39	28	31	PB0	I/O	C <sub>T</sub>		X		ei2		X	X	Port B0	<b>Caution:</b> Negative current injection not allowed on this pin <sup>5)</sup>	
3	40			PB1	I/O	C <sub>T</sub>		X		ei2		X	X	Port B1		
4	41			PB2	I/O	C <sub>T</sub>		X		ei2		X	X	Port B2		
5	42	29	32	PB3	I/O	C <sub>T</sub>		X		ei2		X	X	Port B3		

**Notes:**

1. In the interrupt input column, “eiX” defines the associated external interrupt vector. If the weak pull-up

**Legend:** x=undefined, R/W=read/write

**Notes:**

1. The contents of the I/O port DR registers are readable only in output configuration. In input configuration, the values of the I/O pins are returned instead of the DR register contents.
2. The bits associated with unavailable pins must always keep their reset value.
3. The Timer A Input Capture 2 pin is not available (not bonded).
  - In Flash devices:  
The TAIC2HR and TAIC2LR registers are not present. Bit 5 of the TACSR register (ICF2) is forced by hardware to 0. Consequently, the corresponding interrupt cannot be used.
4. The Timer A Output Compare 2 pin is not available (not bonded).
  - The TAOC2HR and TAOC2LR Registers are write only, reading them will return undefined values. Bit 4 of the TACSR register (OCF2) is forced by hardware to 0. Consequently, the corresponding interrupt cannot be used.

**Caution:** The TAIC2HR and TAIC2LR registers and the ICF2 and OCF2 flags are not present in Flash devices but are present in the emulator. For compatibility with the emulator, it is recommended to perform a dummy access (read or write) to the TAIC2LR and TAOC2LR registers to clear the interrupt flags.

## INTERRUPTS (Cont'd)

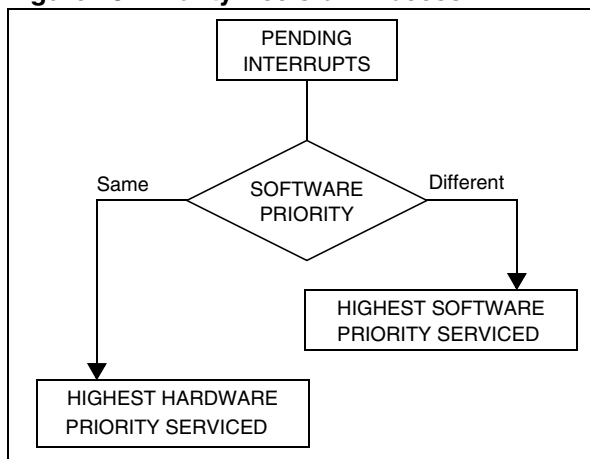
### Servicing Pending Interrupts

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

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

Figure 18 describes this decision process.

**Figure 18. Priority Decision Process**



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

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

**Note 2:** RESET and TRAP 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).

### Non-Maskable Sources

These sources are processed regardless of the state of the I1 and I0 bits of the CC register (see Figure 17). After stacking the PC, X, A and CC registers (except for RESET), the corresponding

vector is loaded in the PC register and the I1 and I0 bits of the CC are set to disable interrupts (level 3). These sources allow the processor to exit HALT mode.

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

This software interrupt is serviced when the TRAP instruction is executed. It will be serviced according to the flowchart in Figure 17.

#### ■ RESET

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

See the RESET chapter for more details.

### Maskable Sources

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

#### ■ External Interrupts

External interrupts allow the processor to exit from HALT low power mode. External interrupt sensitivity is software selectable through the External Interrupt Control register (EICR).

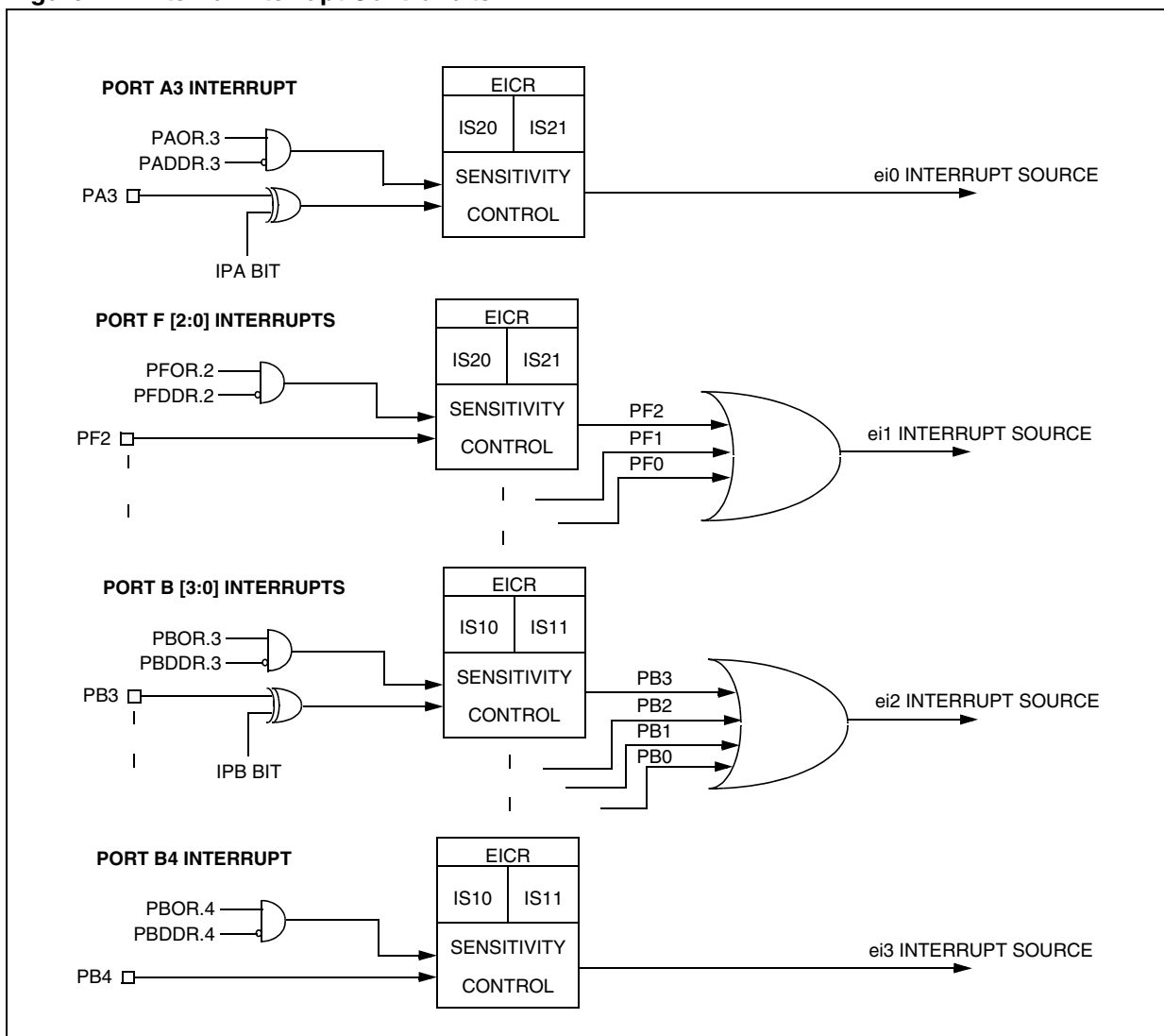
External interrupt triggered on edge will be latched and the interrupt request automatically cleared upon entering the interrupt service routine.

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.

**Figure 21. External Interrupt Control bits**

## 8 POWER SAVING MODES

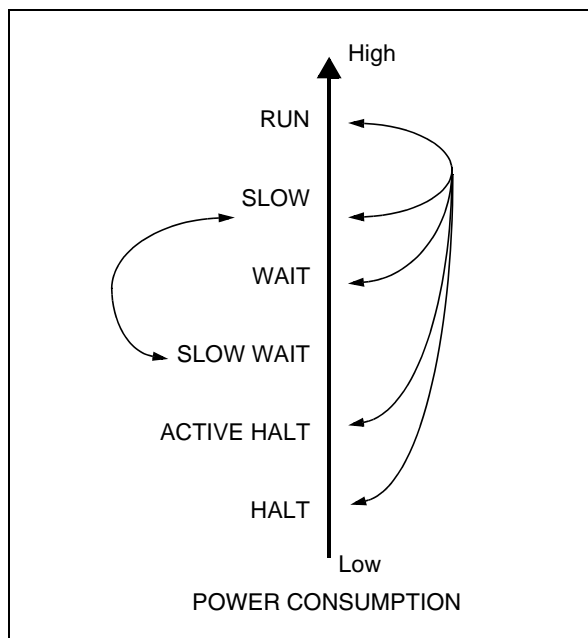
### 8.1 INTRODUCTION

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 22): 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 22. Power Saving Mode Transitions**



### 8.2 SLOW MODE

This mode has two targets:

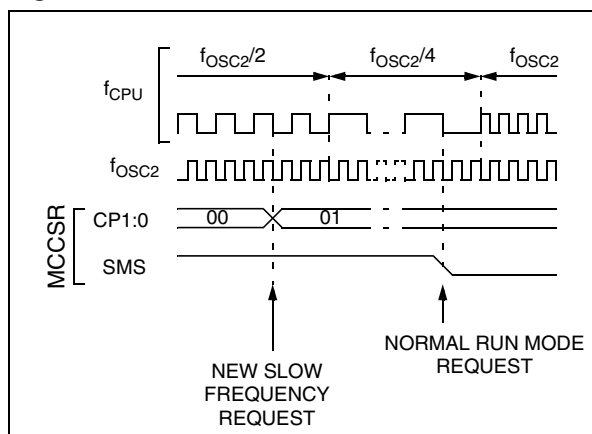
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency ( $f_{CPU}$ ) to the available supply voltage.

SLOW mode is controlled by three bits in the 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 23. SLOW Mode Clock Transitions**





WATCHDOG TIMER (Cont'd)

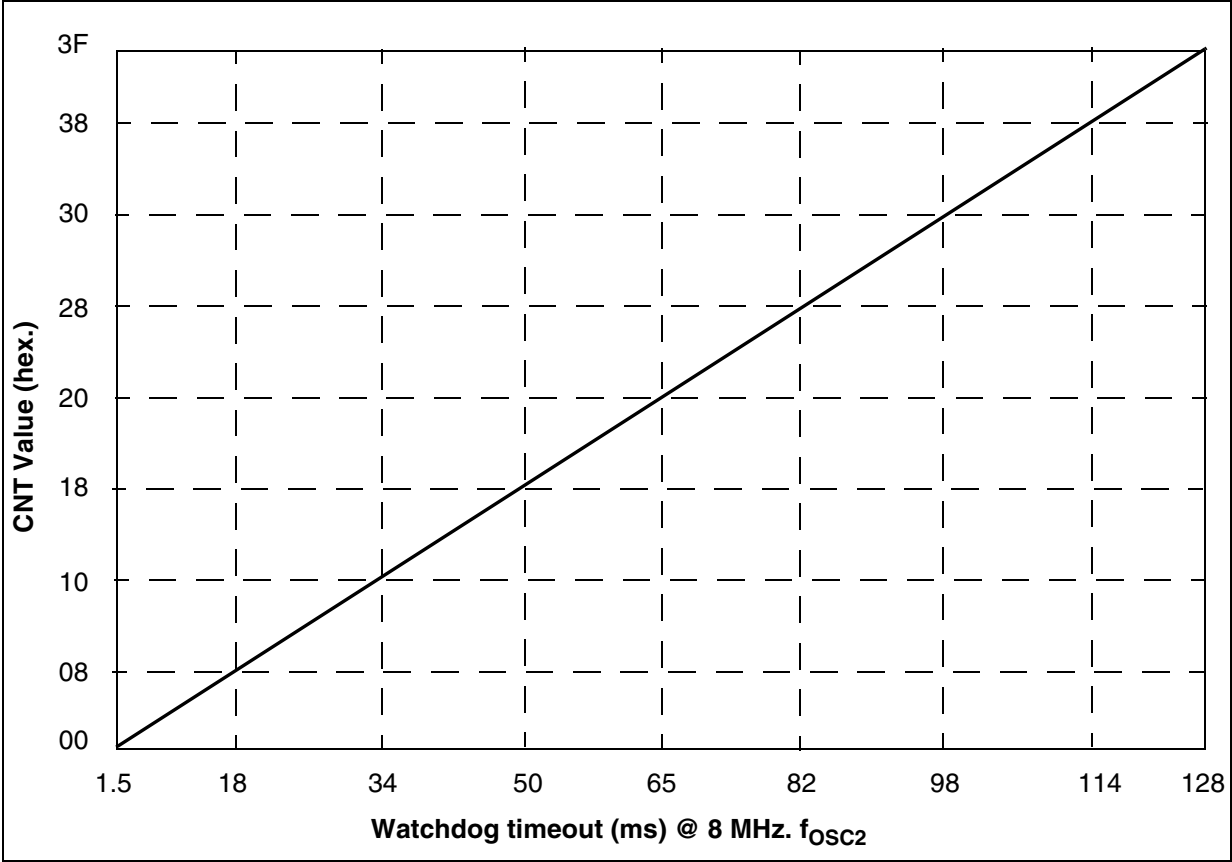
10.1.4 How to Program the Watchdog Timeout

Figure 32 shows the linear relationship between the 6-bit value to be loaded in the Watchdog Counter (CNT) and the resulting timeout duration in milliseconds. This can be used for a quick calculation without taking the timing variations into account. If

more precision is needed, use the formulae in Figure 33.

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

Figure 32. Approximate Timeout Duration



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

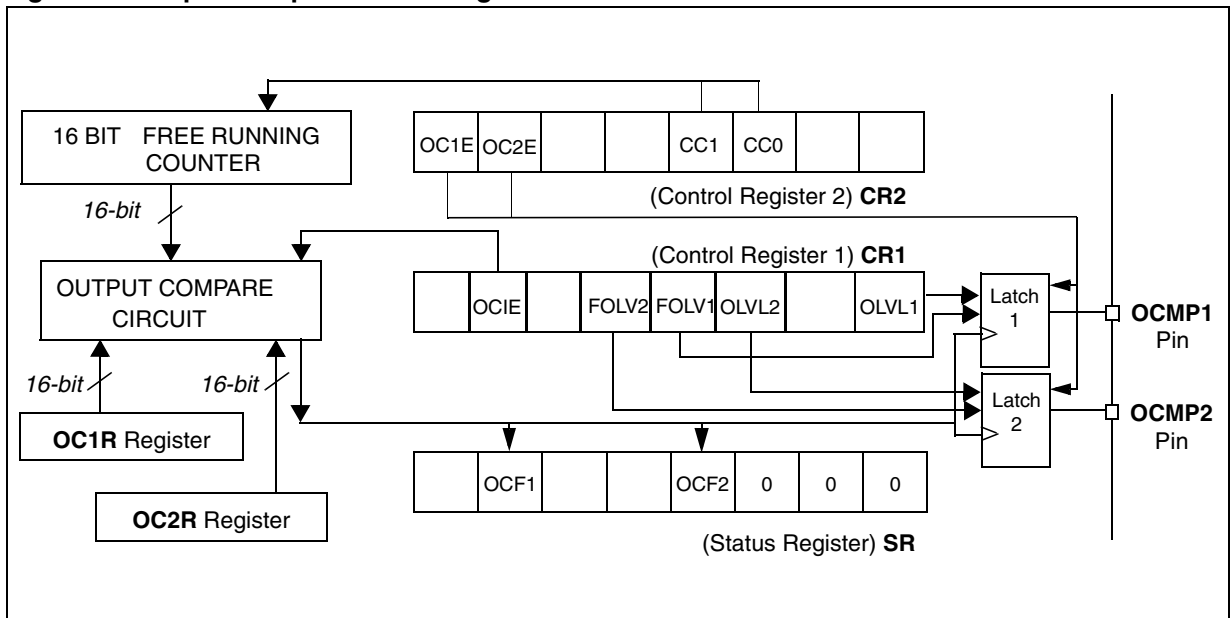
1. After a processor write cycle to the OC/HR register, the output compare function is inhibited until the OC/LR register is also written.
2. If the OC/E bit is not set, the OCMP*i* pin is a general I/O port and the OLV*L**i* bit will not appear when a match is found but an interrupt could be generated if the OCIE bit is set.
3. When the timer clock is  $f_{CPU}/2$ , OCF*i* and OCMP*i* are set while the counter value equals the OC/R register value (see Figure 42 on page 67). This behaviour is the same in OPM or PWM mode.  
When the timer clock is  $f_{CPU}/4$ ,  $f_{CPU}/8$  or in external clock mode, OCF*i* and OCMP*i* are set while the counter value equals the OC/R register value plus 1 (see Figure 43 on page 67).
4. The output compare functions can be used both for generating external events on the OCMP*i* pins even if the input capture mode is also used.
5. The value in the 16-bit OC/R register and the OLV*i* bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout.

6. In Flash devices, the TAOC2HR, TAOC2LR registers are “write only” in Timer A. The corresponding event cannot be generated (OCF2 is forced by hardware to 0).

**Forced Compare Output capability**

When the FOLV*i* bit is set by software, the OLV*L**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/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 41. Output Compare Block Diagram**

**16-BIT TIMER (Cont'd)****10.3.3.6 Pulse Width Modulation Mode**

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

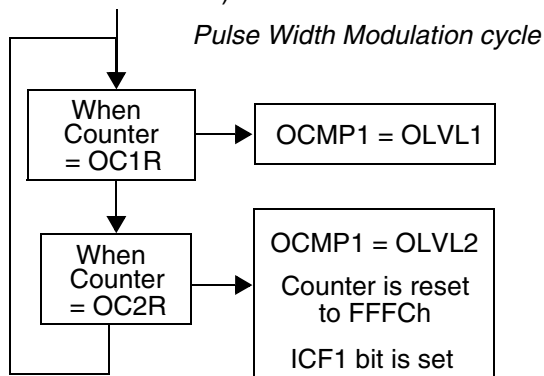
Pulse Width Modulation mode uses the complete Output Compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

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

**Procedure**

To use pulse width modulation mode:

1. Load the OC2R register with the value corresponding to the period of the signal using the formula in the opposite column.
2. Load the OC1R register with the value corresponding to the period of the pulse if (OLVL1=0 and OLVL2=1) using the formula in the opposite column.
3. Select the following in the CR1 register:
  - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with the OC1R register.
  - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with the OC2R register.
4. Select the following in the CR2 register:
  - Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.
  - Set the PWM bit.
  - Select the timer clock (CC[1:0]) (see Table 16 Clock Control Bits).



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

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

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

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

Where:

$t$  = Signal or pulse period (in seconds)

$f_{\text{CPU}}$  = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 16)

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

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

Where:

$t$  = Signal or pulse period (in seconds)

$f_{\text{EXT}}$  = External timer clock frequency (in hertz)

The Output Compare 2 event causes the counter to be initialized to FFFCh (See Figure 45)

**Notes:**

1. After a write instruction to the OC/HR register, the output compare function is inhibited until the OC/LR register is also written.
2. The OCF1 and OCF2 bits cannot be set by hardware in PWM mode therefore the Output Compare interrupt is inhibited.
3. The ICF1 bit is set by hardware when the counter reaches the OC2R value and can produce a timer interrupt if the ICIE bit is set and the I bit is cleared.
4. In PWM mode the ICAP1 pin can not be used to perform input capture because it is disconnected to the timer. The ICAP2 pin can be used to perform input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each period and ICF1 can also generates interrupt if ICIE is set.
5. When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.
6. In Flash devices, the TAOC2HR, TAOC2LR registers in Timer A are "write only". A read operation returns an undefined value.
7. In Flash devices, the ICAP2 registers (TAIC2HR, TAIC2LR) are not available in Timer A. The ICF2 bit is forced by hardware to 0.

## SERIAL PERIPHERAL INTERFACE (Cont'd)

## 10.4.6 Low Power Modes

Mode	Description
WAIT	No effect on SPI. SPI interrupt events cause the device to exit from WAIT mode.
HALT	SPI registers are frozen. In HALT mode, the SPI is inactive. SPI operation resumes when the 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.

## 10.4.6.1 Using the SPI to wakeup the MCU from Halt mode

In slave configuration, the SPI is able to wakeup the ST7 device from HALT mode through a SPIF interrupt. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetch). If multiple data transfers have been performed before software clears the SPIF bit, then the OVR bit is set by hardware.

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

**Caution:** The SPI can wake up the 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 10.4.3.2), make sure the master drives a low level on the  $\overline{SS}$  pin when the slave enters Halt mode.

## 10.4.7 Interrupts

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

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

**SERIAL PERIPHERAL INTERFACE (Cont'd)****10.4.8 Register Description****CONTROL REGISTER (SPICR)**

Read/Write

Reset Value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0

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

This bit is set and cleared by software.

0: Interrupt is inhibited

1: An SPI interrupt is generated whenever  
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 10.4.5.1 Master Mode Fault  
(MODF)). The SPE bit is cleared by reset, so the  
SPI peripheral is not initially connected to the ex-  
ternal 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 18 SPI Master  
mode SCK Frequency.

0: Divider by 2 enabled

1: Divider by 2 disabled

**Note:** This bit has no effect in slave mode.**Bit 4 = MSTR** *Master Mode.*This bit is set and cleared by software. It is also  
cleared by hardware when, in master mode,  $\overline{SS}=0$   
(see Section 10.4.5.1 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 func-  
tions of the MISO and MOSI pins are reversed.**Bit 3 = CPOL** *Clock Polarity.*This bit is set and cleared by software. This bit de-  
termines the idle state of the serial Clock. The  
CPOL bit affects both the master and slave  
modes.

0: SCK pin has a low level idle state

1: SCK pin has a high level idle state

**Note:** If CPOL is changed at the communication  
byte boundaries, the SPI must be disabled by re-  
setting the SPE bit.**Bit 2 = CPHA** *Clock Phase.*

This bit is set and cleared by software.

0: The first clock transition is the first data capture  
edge.1: The second clock transition is the first capture  
edge.**Note:** The slave must have the same CPOL and  
CPHA settings as the master.**Bits 1:0 = SPR[1:0]** *Serial Clock Frequency.*These bits are set and cleared by software. Used  
with the SPR2 bit, they select the baud rate of the  
SPI serial clock SCK output by the SPI in master  
mode.**Note:** These 2 bits have no effect in slave mode.**Table 18. SPI Master mode SCK Frequency**

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

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****10.5.7 Register Description****STATUS REGISTER (SCISR)**

Read Only

Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR	NF	FE	PE

**Bit 7 = TDRE** *Transmit data register empty.*

This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TIE bit = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).

0: Data is not transferred to the shift register

1: Data is transferred to the shift register

**Note:** Data is not transferred to the shift register unless the TDRE bit is cleared.**Bit 6 = TC** *Transmission complete.*

This bit is set by hardware when transmission of a frame containing Data is complete. An interrupt is generated if TCIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).

0: Transmission is not complete

1: Transmission is complete

**Note:** TC is not set after the transmission of a Preamble or a Break.**Bit 5 = RDRF** *Received data ready flag.*

This bit is set by hardware when the content of the RDR register has been transferred to the SCIDR register. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: Data is not received

1: Received data is ready to be read

**Bit 4 = IDLE** *Idle line detect.*

This bit is set by hardware when an Idle Line is detected. An interrupt is generated if the ILIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Idle Line is detected

1: Idle Line is detected

**Note:** The IDLE bit is not set again until the RDRF bit has been set itself (that is, a new idle line occurs).**Bit 3 = OR** *Overrun error.*

This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RDRF = 1. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Overrun error

1: Overrun error is detected

**Note:** When this bit is set RDR register content is not lost but the shift register is overwritten.**Bit 2 = NF** *Noise flag.*

This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No noise is detected

1: Noise is detected

**Note:** This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt.**Bit 1 = FE** *Framing error.*

This bit is set by hardware when a de-synchronization, excessive noise or a break character is detected. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Framing error is detected

1: Framing error or break character is detected

**Note:** This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt. If the word currently being transferred causes both frame error and overrun error, it will be transferred and only the OR bit will be set.**Bit 0 = PE** *Parity error.*

This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE = 1 in the SCICR1 register.

0: No parity error

1: Parity error

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****DATA REGISTER (SCIDR)**

Read/Write

Reset Value: Undefined

Contains the Received or Transmitted data character, depending on whether it is read from or written to.

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).

The TDR register provides the parallel interface between the internal bus and the output shift register (see Figure 1.).

The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 1.).

**BAUD RATE REGISTER (SCIBRR)**

Read/Write

Reset Value: 0000 0000 (00h)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0

Bits 7:6 = **SCP[1:0]** *First SCI Prescaler*

These 2 prescaling bits allow several standard clock division ranges:

PR Prescaling factor	SCP1	SCP0
1	0	0
3	0	1
4	1	0
13	1	1

Bits 5:3 = **SCT[2:0]** *SCI Transmitter rate divisor*

These 3 bits, in conjunction with the SCP1 & SCP0 bits define the total division applied to the bus clock to yield the transmit rate clock in conventional Baud Rate Generator mode.

TR dividing factor	SCT2	SCT1	SCT0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Bits 2:0 = **SCR[2:0]** *SCI Receiver rate divisor.*

These 3 bits, in conjunction with the SCP[1:0] bits define the total division applied to the bus clock to yield the receive rate clock in conventional Baud Rate Generator mode.

RR Dividing factor	SCR2	SCR1	SCR0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

## 10.6 10-BIT A/D CONVERTER (ADC)

### 10.6.1 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 pin out description) that allow the peripheral to convert the analog voltage levels from up to 16 different sources.

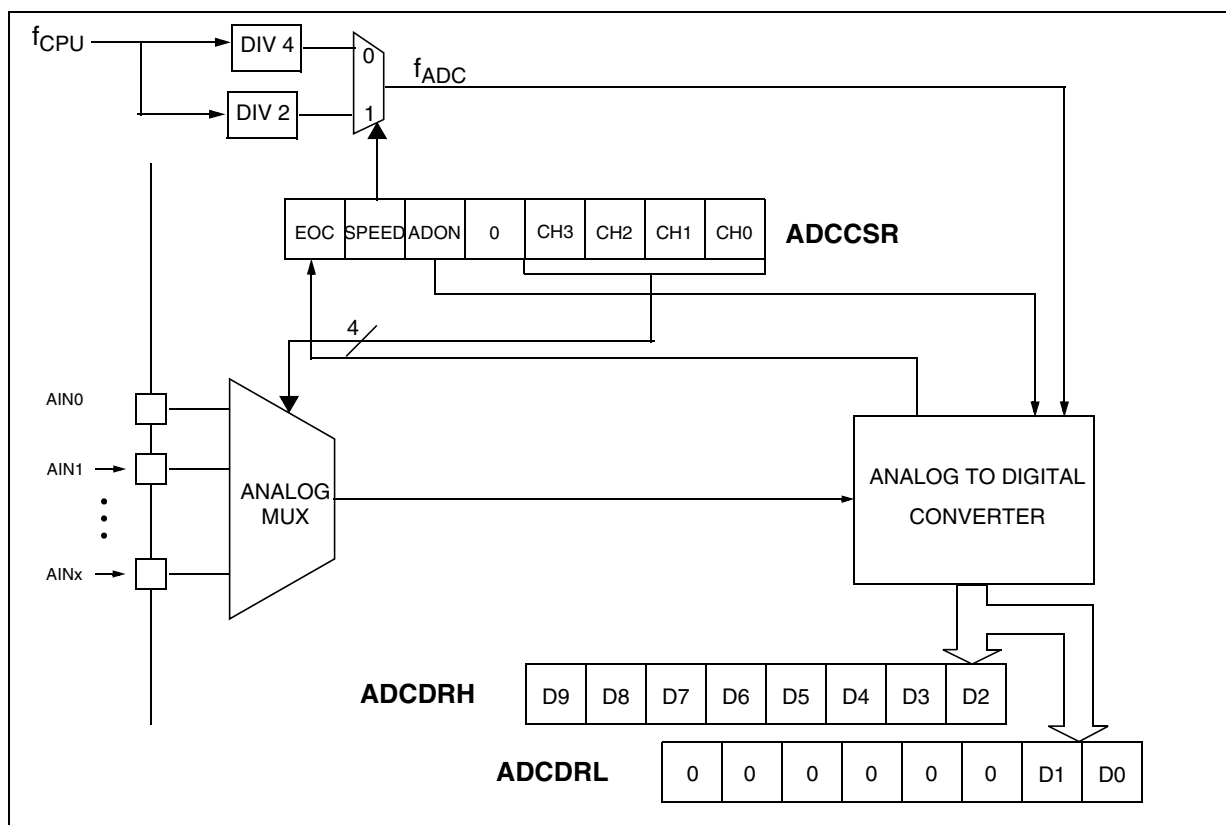
The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

### 10.6.2 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 57.

**Figure 57. ADC Block Diagram**



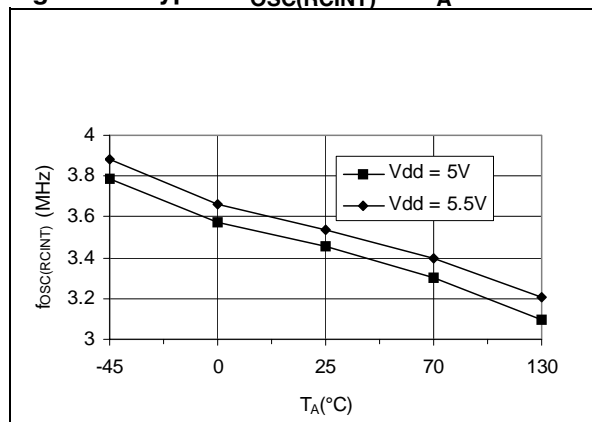


## INSTRUCTION SET OVERVIEW (Cont'd)

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	$A = A \cdot M$	A	M				N	Z	
BCP	Bit compare A, Memory	tst (A . M)	A	M				N	Z	
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 >								

**CLOCK CHARACTERISTICS** (Cont'd)**12.6.4 RC Oscillators**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$f_{\text{OSC}}(\text{RCINT})$	Internal RC oscillator frequency See Figure 67	$T_A = 25^\circ\text{C}$ , $V_{\text{DD}} = 5\text{V}$	2	3.5	5.6	MHz

**Figure 67. Typical  $f_{\text{OSC}}(\text{RCINT})$  vs  $T_A$** 

**Note:** To reduce disturbance to the RC oscillator, it is recommended to place decoupling capacitors between  $V_{\text{DD}}$  and  $V_{\text{SS}}$  as shown in Figure 86

## 12.11 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 characterisation results, not tested in production.

### 12.11.1 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			$t_{CPU}$
		$f_{CPU}=8MHz$	250			ns
$f_{EXT}$	Timer external clock frequency		0		$f_{CPU}/4$	MHz
$f_{PWM}$	PWM repetition rate		0		$f_{CPU}/4$	MHz
$Res_{PWM}$	PWM resolution				16	bit

## 13.2 THERMAL CHARACTERISTICS

Symbol	Ratings	Value	Unit
$R_{thJA}$	Package thermal resistance (junction to ambient)		
	TQFP44 10x10	52	°C/W
	TQFP32 7x7	70	
	SDIP42 600mil	55	
	SDIP32 200mil	50	
$P_D$	Power dissipation <sup>1)</sup>	500	mW
$T_{Jmax}$	Maximum junction temperature <sup>2)</sup>	150	°C

**Notes:**

1. The power dissipation is obtained from the formula  $P_D = P_{INT} + P_{PORT}$  where  $P_{INT}$  is the chip internal power ( $I_{DD} \times V_{DD}$ ) and  $P_{PORT}$  is the port power dissipation determined by the user.
2. The average chip-junction temperature can be obtained from the formula  $T_J = T_A + P_D \times R_{thJA}$ .

Table 30. ST7 Application Notes

IDENTIFICATION	DESCRIPTION
AN 978	ST7 VISUAL DEVELOP SOFTWARE KEY DEBUGGING FEATURES
AN 983	KEY FEATURES OF THE COSMIC ST7 C-COMPILER PACKAGE
AN 985	EXECUTING CODE IN ST7 RAM
AN 986	USING THE INDIRECT ADDRESSING MODE WITH ST7
AN 987	ST7 SERIAL TEST CONTROLLER PROGRAMMING
AN 988	STARTING WITH ST7 ASSEMBLY TOOL CHAIN
AN 989	GETTING STARTED WITH THE ST7 HIWARE C TOOLCHAIN
AN1039	ST7 MATH UTILITY ROUTINES
AN1064	WRITING OPTIMIZED HIWARE C LANGUAGE FOR ST7
AN1106	TRANSLATING ASSEMBLY CODE FROM HC05 TO ST7
AN1446	USING THE ST72521 EMULATOR TO DEBUG A ST72324 TARGET APPLICATION
AN1478	PORTING AN ST7 PANTA PROJECT TO CODEWARRIOR IDE
AN1575	ON-BOARD PROGRAMMING METHODS FOR XFLASH AND HDFLASH ST7 MCUS
AN1576	IN-APPLICATION PROGRAMMING (IAP) DRIVERS FOR ST7 HDFLASH OR XFLASH MCUS
AN1635	ST7 CUSTOMER ROM CODE RELEASE INFORMATION
AN1754	DATA LOGGING PROGRAM FOR TESTING ST7 APPLICATIONS VIA ICC
AN1796	FIELD UPDATES FOR FLASH BASED ST7 APPLICATIONS USING A PC COMM PORT
<b>SYSTEM OPTIMIZATION</b>	
AN1711	SOFTWARE TECHNIQUES FOR COMPENSATING ST7 ADC ERRORS