



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

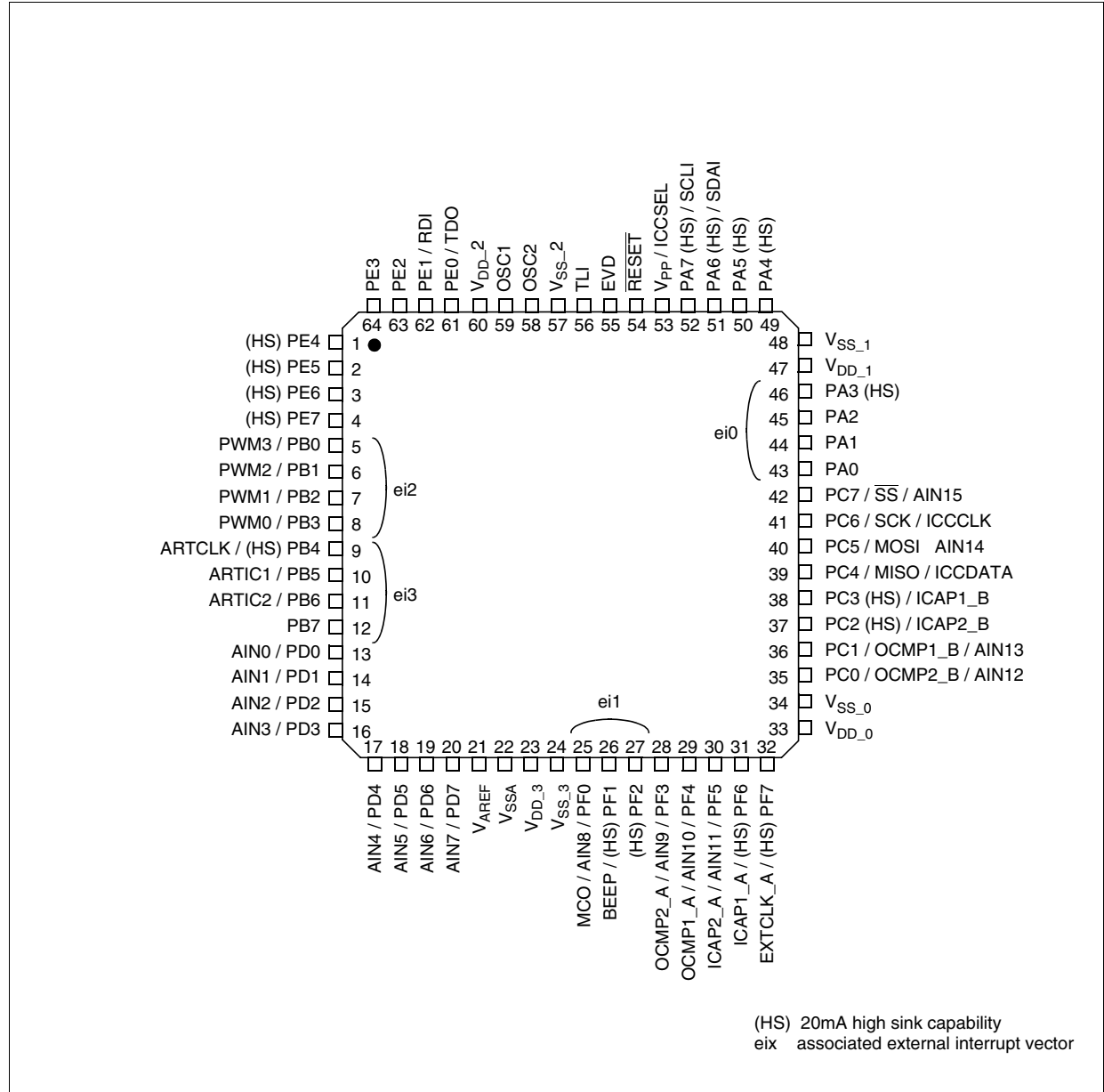
Product Status	Not For New Designs
Core Processor	ST7
Core Size	8-Bit
Speed	8MHz
Connectivity	I ² C, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	48
Program Memory Size	48KB (48K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	3.8V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f321bar7t6

Table of Contents

1 DESCRIPTION	7
2 PIN DESCRIPTION	8
3 REGISTER & MEMORY MAP	14
4 FLASH PROGRAM MEMORY	18
4.1 INTRODUCTION	18
4.2 MAIN FEATURES	18
4.3 STRUCTURE	18
4.3.1 Read-out Protection	18
4.4 ICC INTERFACE	19
4.5 ICP (IN-CIRCUIT PROGRAMMING)	20
4.6 IAP (IN-APPLICATION PROGRAMMING)	20
4.7 RELATED DOCUMENTATION	20
4.7.1 Register Description	20
5 CENTRAL PROCESSING UNIT	21
5.1 INTRODUCTION	21
5.2 MAIN FEATURES	21
5.3 CPU REGISTERS	21
6 SUPPLY, RESET AND CLOCK MANAGEMENT	24
6.1 PHASE LOCKED LOOP	24
6.2 MULTI-OSCILLATOR (MO)	25
6.3 RESET SEQUENCE MANAGER (RSM)	26
6.3.1 Introduction	26
6.3.2 Asynchronous External RESET pin	26
6.3.3 External Power-On RESET	27
6.3.4 Internal Low Voltage Detector (LVD) RESET	27
6.3.5 Internal Watchdog RESET	27
6.4 SYSTEM INTEGRITY MANAGEMENT (SI)	28
6.4.1 Low Voltage Detector (LVD)	28
6.4.2 Auxiliary Voltage Detector (AVD)	29
6.4.3 Low Power Modes	30
6.4.4 Register Description	31
7 INTERRUPTS	32
7.1 INTRODUCTION	32
7.2 MASKING AND PROCESSING FLOW	32
7.3 INTERRUPTS AND LOW POWER MODES	34
7.4 CONCURRENT & NESTED MANAGEMENT	34
7.5 INTERRUPT REGISTER DESCRIPTION	35
7.6 EXTERNAL INTERRUPTS	37
7.6.1 I/O Port Interrupt Sensitivity	37
7.7 EXTERNAL INTERRUPT CONTROL REGISTER (EICR)	39
8 POWER SAVING MODES	41
8.1 INTRODUCTION	41
8.2 SLOW MODE	41
8.3 WAIT MODE	42

2 PIN DESCRIPTION

Figure 2. 64-Pin LQFP 14x14 and 10x10 Package Pinout



CENTRAL PROCESSING UNIT (Cont'd)**Stack Pointer (SP)**

Read/Write

Reset Value: 01 FFh

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

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 2).

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.

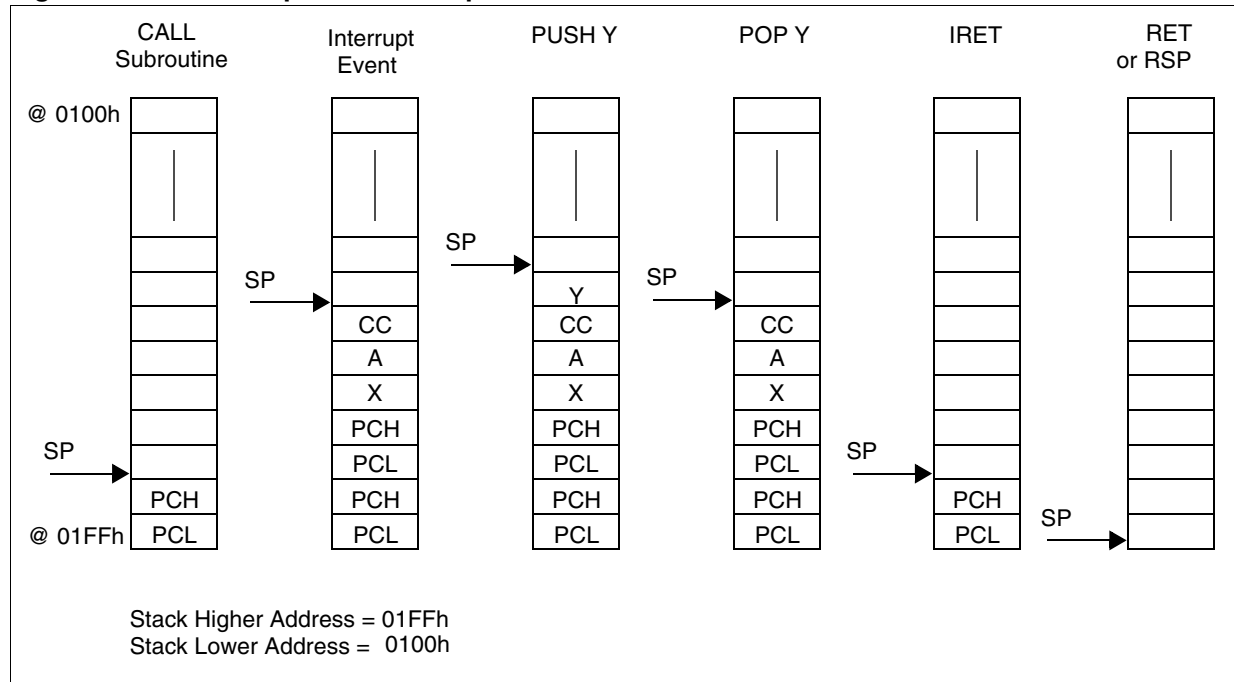
The least significant byte of the Stack Pointer (called S) can be directly accessed by a LD instruction.

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 under-flow.

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 2.

- 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 10. Stack Manipulation Example

6 SUPPLY, RESET AND CLOCK MANAGEMENT

The device includes a range of utility features for securing the application in critical situations (for example in case of a power brown-out), and reducing the number of external components. An overview is shown in Figure 12.

For more details, refer to dedicated parametric section.

Main features

- Optional PLL for multiplying the frequency by 2 (not to be used with internal RC oscillator)
- Reset Sequence Manager (RSM)
- Multi-Oscillator Clock Management (MO)
 - 5 Crystal/Ceramic resonator oscillators
 - 1 Internal RC oscillator
- System Integrity Management (SI)
 - Main supply Low voltage detection (LVD)
 - Auxiliary Voltage detector (AVD) with interrupt capability for monitoring the main supply or the EVD pin

6.1 PHASE LOCKED LOOP

If the clock frequency input to the PLL is in the range 2 to 4 MHz, the PLL can be used to multiply the frequency by two to obtain an f_{OSC2} of 4 to 8 MHz. The PLL is enabled by option byte. If the PLL is disabled, then $f_{OSC2} = f_{OSC}/2$.

Caution: The PLL is not recommended for applications where timing accuracy is required. See “PLL Characteristics” on page 150.

Figure 11. PLL Block Diagram

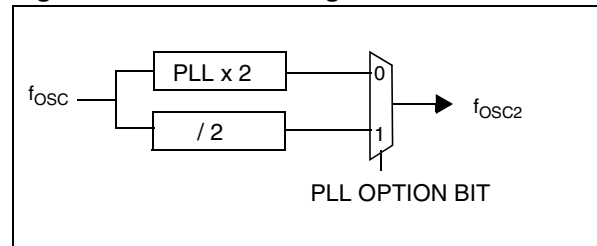
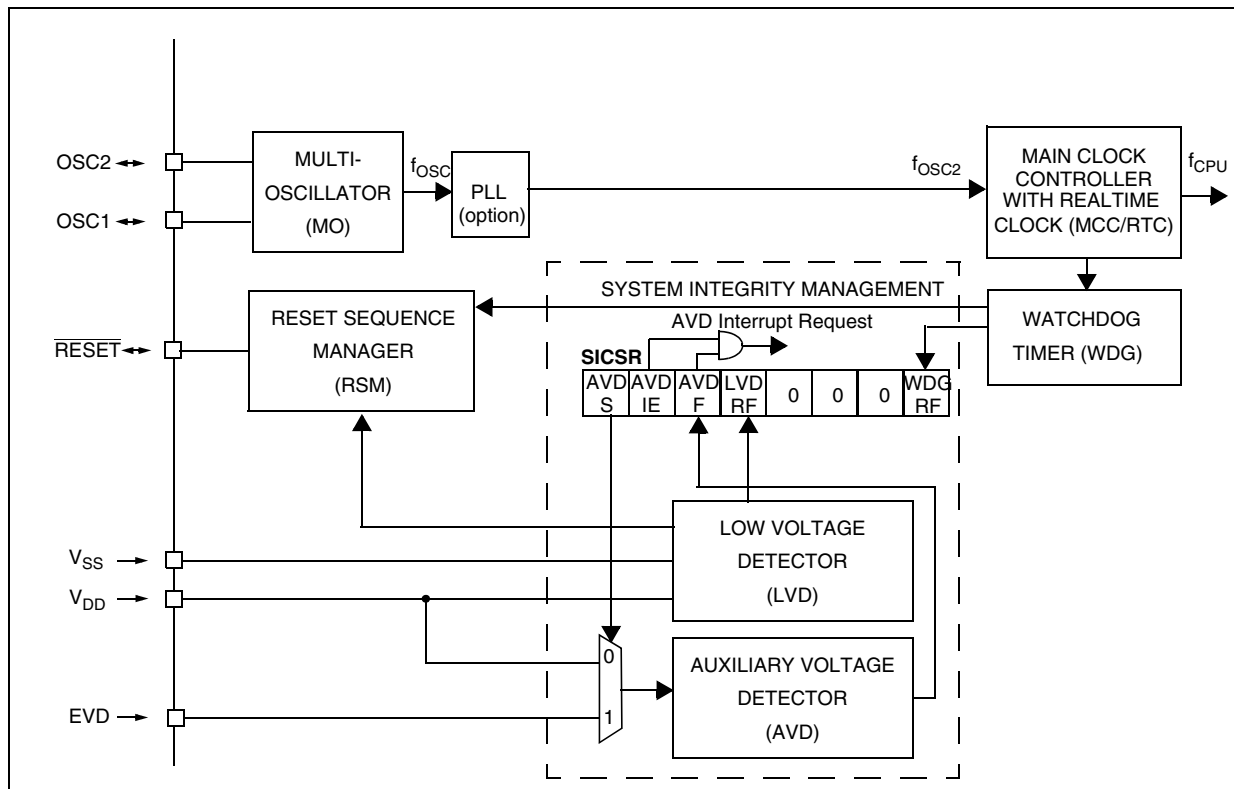


Figure 12. Clock, Reset and Supply Block Diagram



I/O PORTS (Cont'd)

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.

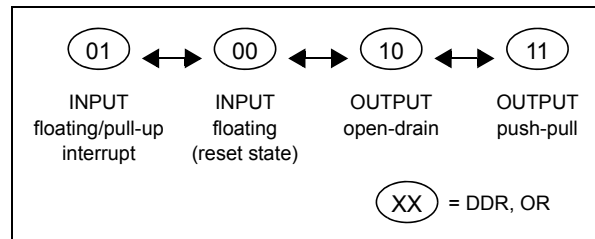
It is recommended not to change the voltage level or loading on any port pin while conversion is in progress. Furthermore it is recommended not to have clocking pins located close to a selected analog pin.

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

9.3 I/O PORT IMPLEMENTATION

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific 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 2 on page 4. Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 32. Interrupt I/O Port State Transitions**9.4 LOW POWER MODES**

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

9.5 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).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
External interrupt on selected external event	-	DDR _x OR _x	Yes	

10 ON-CHIP PERIPHERALS

10.1 WATCHDOG TIMER (WDG)

10.1.1 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.

10.1.2 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

10.1.3 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 the reset pin low for typically $30\mu s$.

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:

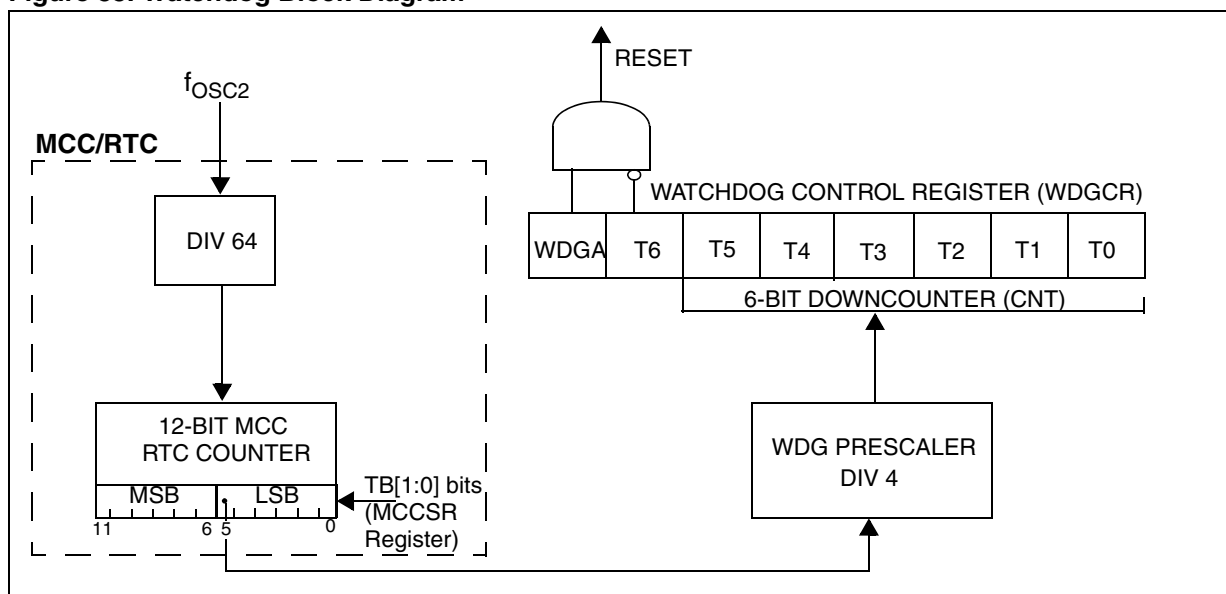
- The WDGA bit is set (watchdog enabled)
- The T6 bit is set to prevent generating an immediate reset
- The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset (see Figure 2. Approximate Timeout Duration). The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WDGCR register (see Figure 3).

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 33. Watchdog Block Diagram



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

10.2.5 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.

10.2.6 Interrupts

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

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

Note:

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

10.2.7 Register Description

MCC CONTROL/STATUS REGISTER (MCCR)

Read/Write

Reset Value: 0000 0000 (00h)

7

0

MCO	CP1	CP0	SMS	TB1	TB0	OIE	OIF
-----	-----	-----	-----	-----	-----	-----	-----

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

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 8.2 SLOW MODE and Section 10.2 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.

Counter Prescaler	Time Base		TB1	TB0
	$f_{OSC2}=4\text{MHz}$	$f_{OSC2}=8\text{MHz}$		
16000	4ms	2ms	0	0
32000	8ms	4ms	0	1
80000	20ms	10ms	1	0
200000	50ms	25ms	1	1

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

Bit 1 = **OIE** Oscillator interrupt enable

This bit set and cleared by software.

0: Oscillator interrupt disabled

1: Oscillator interrupt enabled

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

When this bit is set, calling the ST7 software HALT instruction enters the ACTIVE-HALT power saving mode.

ON-CHIP PERIPHERALS (Cont'd)**Output compare and Time base interrupt**

On overflow, the OVF flag of the ARTCSR register is set and an overflow interrupt request is generated if the overflow interrupt enable bit, OIE, in the ARTCSR register, is set. The OVF flag must be reset by the user software. This interrupt can be used as a time base in the application.

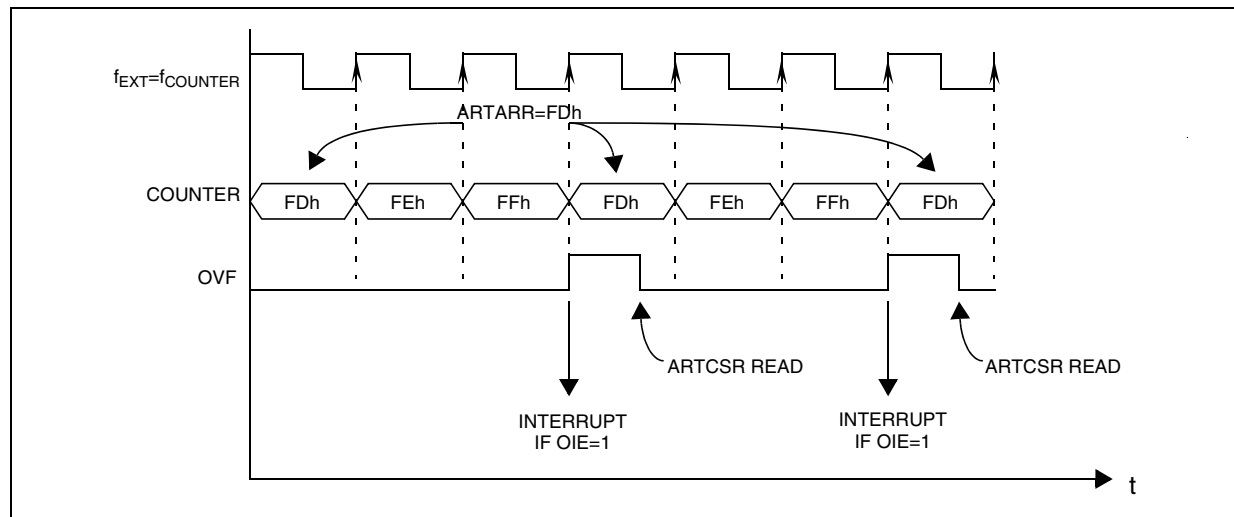
External clock and event detector mode

Using the f_{EXT} external prescaler input clock, the auto-reload timer can be used as an external clock event detector. In this mode, the ARTARR register is used to select the n_{EVENT} number of events to be counted before setting the OVF flag.

$$n_{EVENT} = 256 - ARTARR$$

Caution: The external clock function is not available in HALT mode. If HALT mode is used in the application, prior to executing the HALT instruction, the counter must be disabled by clearing the TCE bit in the ARTCSR register to avoid spurious counter increments.

Figure 41. External Event Detector Example (3 counts)



16-BIT TIMER (Cont'd)**10.4.3.3 Input Capture**

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

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

	MS Byte	LS Byte
ICiR	ICiHR	ICiLR

ICiR 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 1).
- Select the edge of the active transition on the ICAP2 pin with the IEDG2 bit (the ICAP2 pin must be configured as floating input or input with pull-up without interrupt if this configuration is available).

And select the following in the CR1 register:

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

When an input capture occurs:

- ICF i bit is set.
- The ICiR register contains the value of the free running counter on the active transition on the ICAP i pin (see Figure 6).
- 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 (that is, 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 ICiLR register.

Notes:

1. After reading the ICiHR register, transfer of input capture data is inhibited and ICF i 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 two input capture functions can be used together even if the timer also uses the two output compare functions.
4. In One Pulse mode and PWM mode only Input Capture 2 can be used.
5. The alternate inputs (ICAP1 and ICAP2) are always directly connected to the timer. So any transitions on these pins activates the input capture function.
Moreover if one of the ICAP i pins is configured as an input and the second one as an output, an interrupt can be generated if the user toggles the output pin and if the ICIE bit is set. This can be avoided if the input capture function i is disabled by reading the 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).

SERIAL PERIPHERAL INTERFACE (Cont'd)**10.5.6 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.

10.5.6.1 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 10.5.3.2), make sure the master drives a low level on the \overline{SS} pin when the slave enters Halt mode.

10.5.7 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 COMMUNICATIONS INTERFACE (Cont'd)**10.6.4.2 Transmitter**

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

Character Transmission

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the SCIDR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 1.).

Procedure

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

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

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

The TDRE bit is set by hardware and it indicates:

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

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

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

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

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

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

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

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

Break Characters

Setting the SBK bit loads the shift register with a break character. The break frame length depends on the M bit (see Figure 2.).

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

Idle Characters

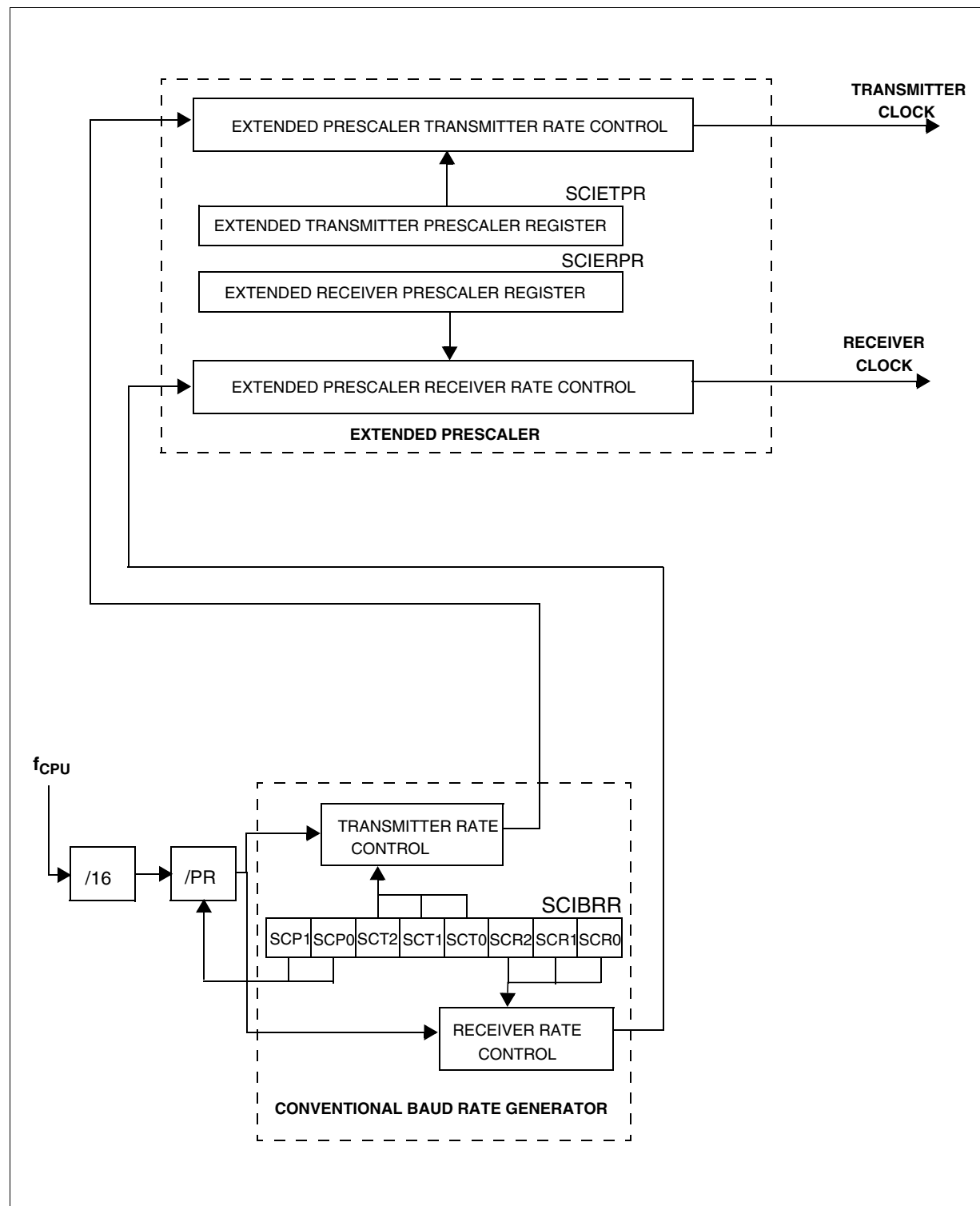
Setting the TE bit drives the SCI to send an idle frame before the first data frame.

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

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

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 63. SCI Baud Rate and Extended Prescaler Block Diagram



SERIAL COMMUNICATIONS INTERFACE (Cont'd)**CONTROL REGISTER 1 (SCICR1)**

Read/Write

Reset Value: x000 0000 (x0h)

7								0
R8	T8	SCID	M	WAKE	PCE	PS	PIE	

Bit 7 = R8 *Receive data bit 8.*

This bit is used to store the 9th bit of the received word when M = 1.

Bit 6 = T8 *Transmit data bit 8.*

This bit is used to store the 9th bit of the transmitted word when M = 1.

Bit 5 = SCID *Disabled for low power consumption*
 When this bit is set the SCI prescalers and outputs are stopped and the end of the current byte transfer in order to reduce power consumption. This bit is set and cleared by software.

0: SCI enabled

1: SCI prescaler and outputs disabled

Bit 4 = M *Word length.*

This bit determines the word length. It is set or cleared by software.

0: 1 Start bit, 8 Data bits, 1 Stop bit

1: 1 Start bit, 9 Data bits, 1 Stop bit

Note: The M bit must not be modified during a data transfer (both transmission and reception).

Bit 3 = WAKE *Wake-Up method.*

This bit determines the SCI Wake-Up method, it is set or cleared by software.

0: Idle Line

1: Address Mark

Bit 2 = PCE *Parity control enable.*

This bit selects the hardware parity control (generation and detection). When the parity control is enabled, the computed parity is inserted at the MSB position (9th bit if M = 1; 8th bit if M = 0) and parity is checked on the received data. This bit is set and cleared by software. Once it is set, PCE is active after the current byte (in reception and in transmission).

0: Parity control disabled

1: Parity control enabled

Bit 1 = PS *Parity selection.*

This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity is selected after the current byte.

0: Even parity

1: Odd parity

Bit 0 = PIE *Parity interrupt enable.*

This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). It is set and cleared by software.

0: Parity error interrupt disabled

1: Parity error interrupt enabled.

10.8 10-BIT A/D CONVERTER (ADC)

10.8.1 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.

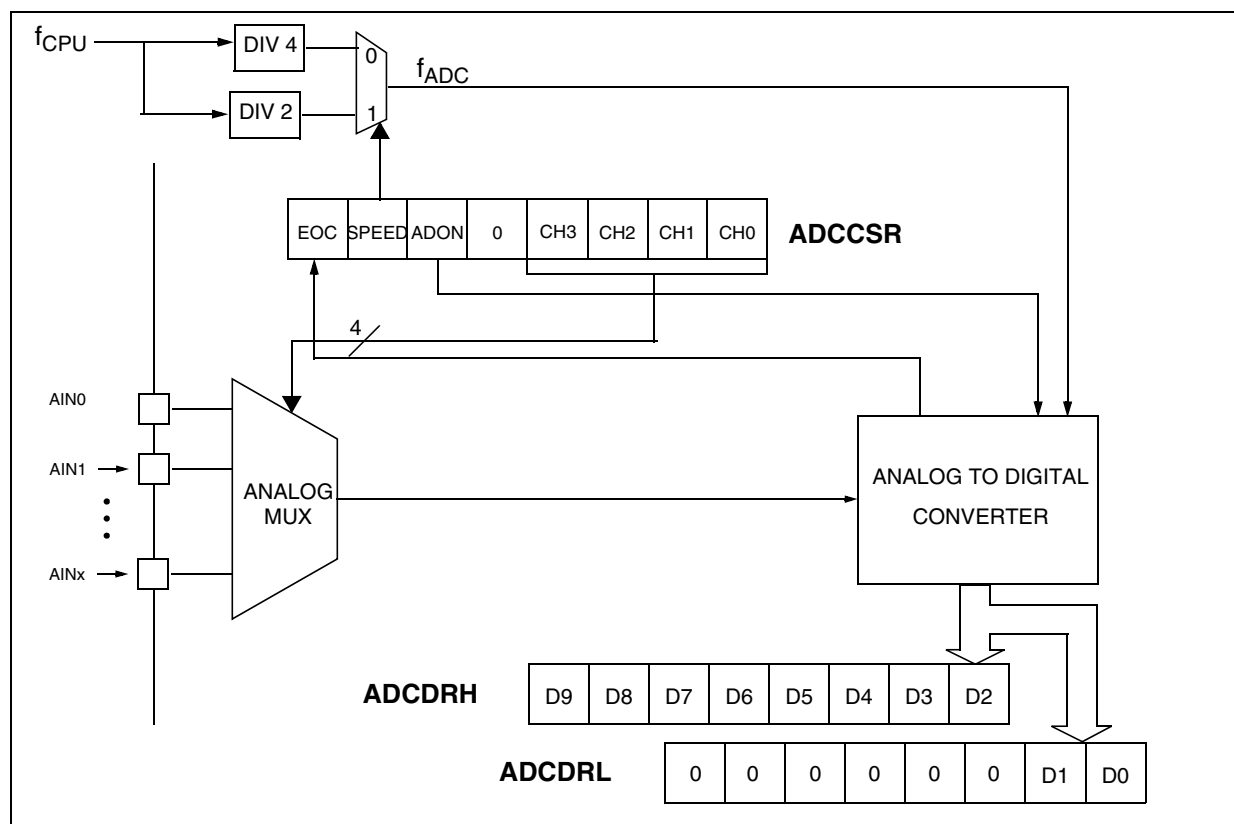
The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

10.8.2 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 69.

Figure 69. ADC Block Diagram



10-BIT A/D CONVERTER (Cont'd)

Table 25. 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	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

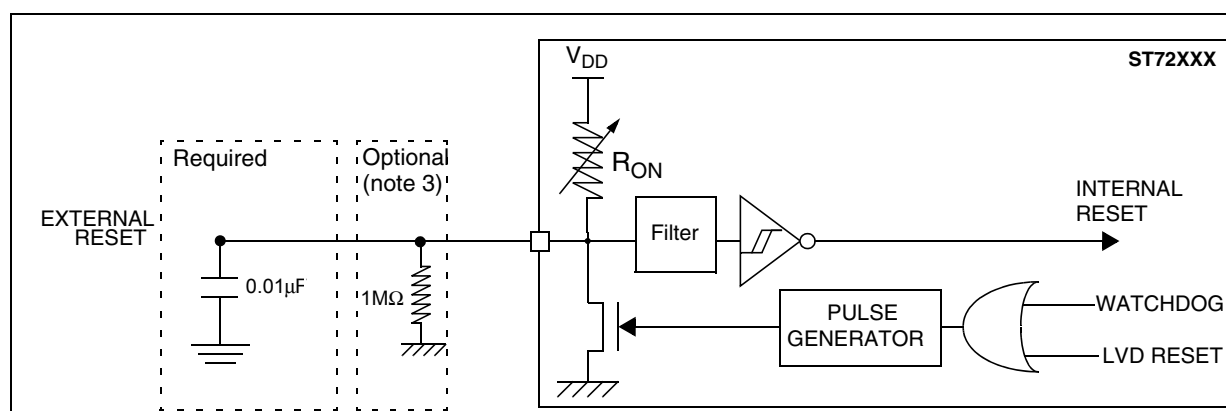
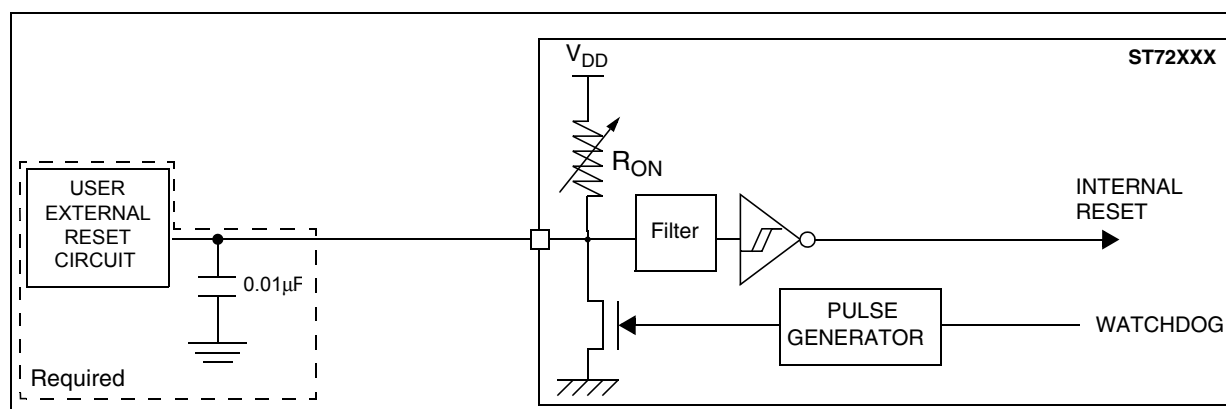
CLOCK AND TIMING CHARACTERISTICS (Cont'd)

Supplier	f _{osc} (MHz)	Typical Ceramic Resonators ¹⁾	
		Reference ²⁾	Recommended OSC RANGE Option bit configuration
Murata	2	CSTCC2M00G56A-R0	MP Mode ³⁾
	4	CSTCR4M00G55B-R0	MS Mode
	8	CSTCE8M00G55A-R0	HS Mode
	16	CSTCE16M0G53A-R0	HS Mode

Notes:

1. Resonator characteristics given by the ceramic resonator manufacturer.
 2. SMD = [-R0: Plastic tape package (\varnothing =180mm), -B0: Bulk]
LEAD = [-A0: Flat pack package (Radial taping Ho= 18mm), -B0: Bulk]
 3. LP mode is not recommended for 2 MHz resonator because the peak to peak amplitude is too small (>0.8V)
- For more information on these resonators, please consult www.murata.com

CONTROL PIN CHARACTERISTICS (Cont'd)

Figure 86. RESET pin protection when LVD is enabled.¹⁾²⁾³⁾⁴⁾Figure 87. RESET pin protection when LVD is disabled.¹⁾**Note 1:**

- The reset network protects the device against parasitic resets.
- The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).
- 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 12.9.1 on page 158. Otherwise the reset will not be taken into account internally.
- 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 is less than the absolute maximum value specified for $I_{INJ}(\text{RESET})$ in section 12.2.2 on page 139.

Note 2: When the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.

Note 3: In case a capacitive power supply is used, it is recommended to connect a 1MΩ pull-down resistor to the $\overline{\text{RESET}}$ pin to discharge any residual voltage induced by the capacitive effect of the power supply (this will add 5µA to the power consumption of the MCU).

Note 4: Tips when using the LVD:

1. Check that all recommendations related to reset circuit have been applied (see notes above).
2. Check that the power supply is properly decoupled (100nF + 10µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100nF + 1MΩ pull-down on the $\overline{\text{RESET}}$ pin.
3. The capacitors connected on the $\overline{\text{RESET}}$ pin and also the power supply are key to avoid any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: replace 10nF pull-down on the $\overline{\text{RESET}}$ pin with a 5µF to 20µF capacitor.

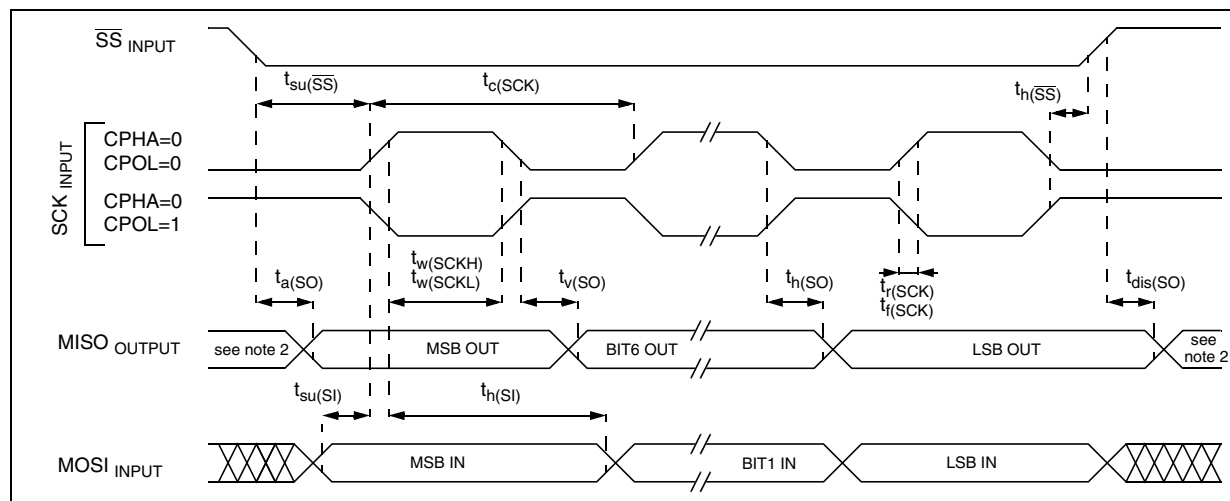
12.11 COMMUNICATION INTERFACE CHARACTERISTICS

12.11.1 SPI - Serial Peripheral Interface

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

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (\overline{SS} , SCK, MOSI, MISO).

Symbol	Parameter	Conditions	Min	Max	Unit
f_{SCK} $1/t_c(SCK)$	SPI clock frequency	Master $f_{CPU}=8MHz$	$f_{CPU}/128$ 0.0625	$f_{CPU}/4$ 2	MHz
		Slave $f_{CPU}=8MHz$	0	$f_{CPU}/2$ 4	
$t_r(SCK)$ $t_f(SCK)$	SPI clock rise and fall time		see I/O port pin description		
$t_{su}(\overline{SS})$	\overline{SS} setup time ⁴⁾	Slave	$t_{CPU} + 50$		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)		120	
$t_h(SO)$	Data output hold time		0		
$t_v(MO)$	Data output valid time	Master (after enable edge)		120	t_{CPU}
$t_h(MO)$	Data output hold time		0		

Figure 89. SPI Slave Timing Diagram with CPHA=0³⁾

Notes:

1. Data based on design simulation and/or characterisation results, not tested in production.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration.
3. Measurement points are done at CMOS levels: $0.3 \times V_{DD}$ and $0.7 \times V_{DD}$.
4. Depends on f_{CPU} . For example, if $f_{CPU} = 8 \text{ MHz}$, then $t_{CPU} = 1 / f_{CPU} = 125 \text{ ns}$ and $t_{su}(\overline{SS}) = 175 \text{ ns}$.

13.3 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.
- ECOPACK is an ST trademark. ECOPACK® specifications are available at www.st.com.

15 KNOWN LIMITATIONS

15.1 ALL DEVICES

15.1.1 Unexpected Reset Fetch

If an interrupt request occurs while a “POP CC” instruction is executed, the interrupt controller does not recognise 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.

15.1.2 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
jne OUT
TNZ Y
jne OUT
LD A,sema    ; check the semaphore status if
edge is detected
CP A,#01
jne 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
```