**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 | SPI |
| Peripherals | LVD, POR, PWM, WDT |
| Number of I/O | 13 |
| Program Memory Size | 1.5KB (1.5K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | 128 x 8 |
| RAM Size | 128 x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.4V ~ 5.5V |
| Data Converters | A/D 5x8b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 16-SOIC (0.154", 3.90mm Width) |
| Supplier Device Package | 16-SO |
| Purchase URL | https://www.e-xfl.com/product-detail/stmicroelectronics/st7flite09y0m6 |

# Table of Contents

# 1 DESCRIPTION

The ST7LITE0x and ST7SUPERLITE (ST7LITESx) are members of the ST7 microcontroller family. All ST7 devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

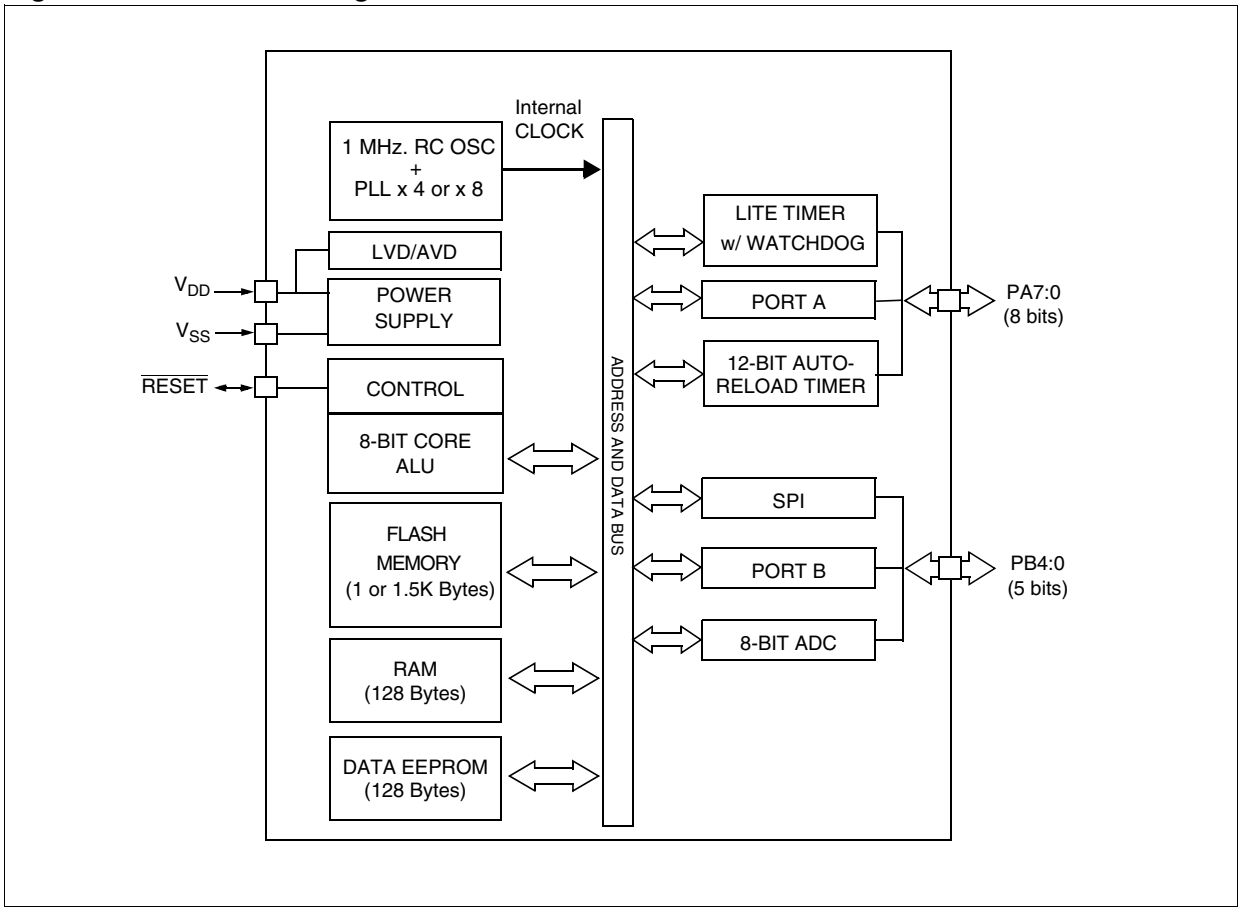
The ST7LITE0x and ST7SUPERLITE feature FLASH memory with byte-by-byte In-Circuit Programming (ICP) and In-Application Programming (IAP) capability.

Under software control, the ST7LITE0x and ST7SUPERLITE devices can be placed in WAIT, SLOW, or HALT mode, reducing power consumption when the application is in idle or standby state.

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

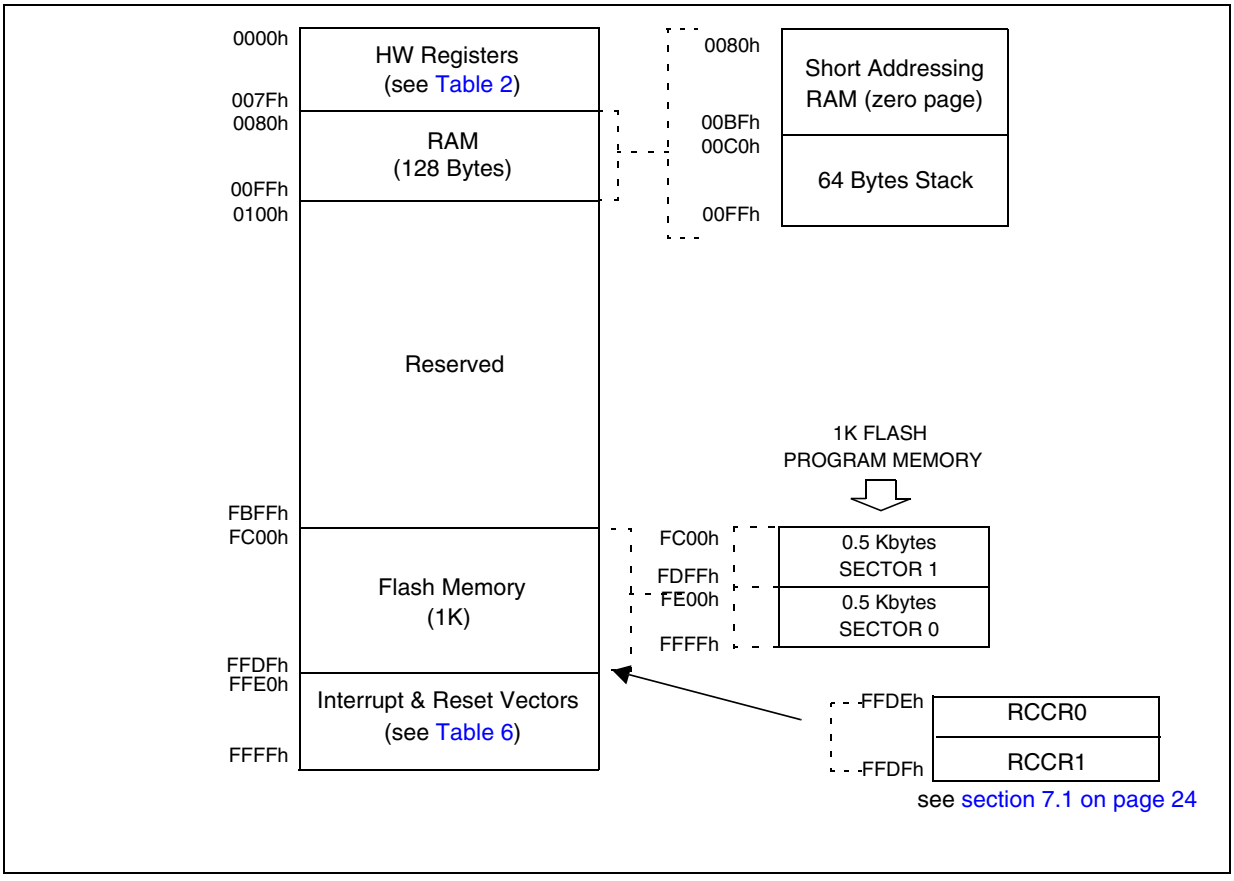
For easy reference, all parametric data are located in section 13 on page 81.

**Figure 1. General Block Diagram**



**REGISTER AND MEMORY MAP** (Cont'd)

**Figure 5. Memory Map (ST7SUPERLITE)**



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

### 4.4 ICC interface

ICP needs a minimum of 4 and up to 6 pins to be connected to the programming tool. These pins are:

- $\overline{\text{RESET}}$: device reset
- $V_{SS}$: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input serial data pin
- CLKIN: main clock input for external source
- $V_{DD}$: application board power supply (optional, see Note 3)

**Notes:**

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

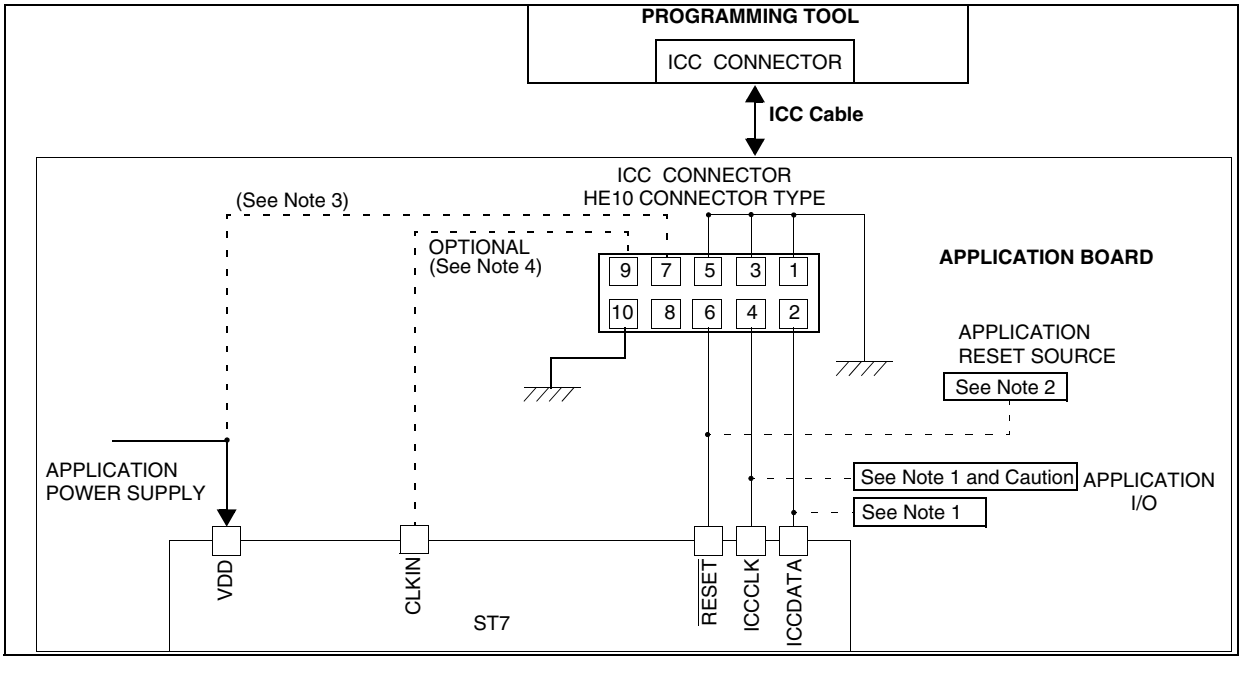
2. During the ICP session, the programming tool must control the $\overline{\text{RESET}}$ pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor<1K). A schottky diode can be used to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

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

4. Pin 9 has to be connected to the CLKIN pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte.

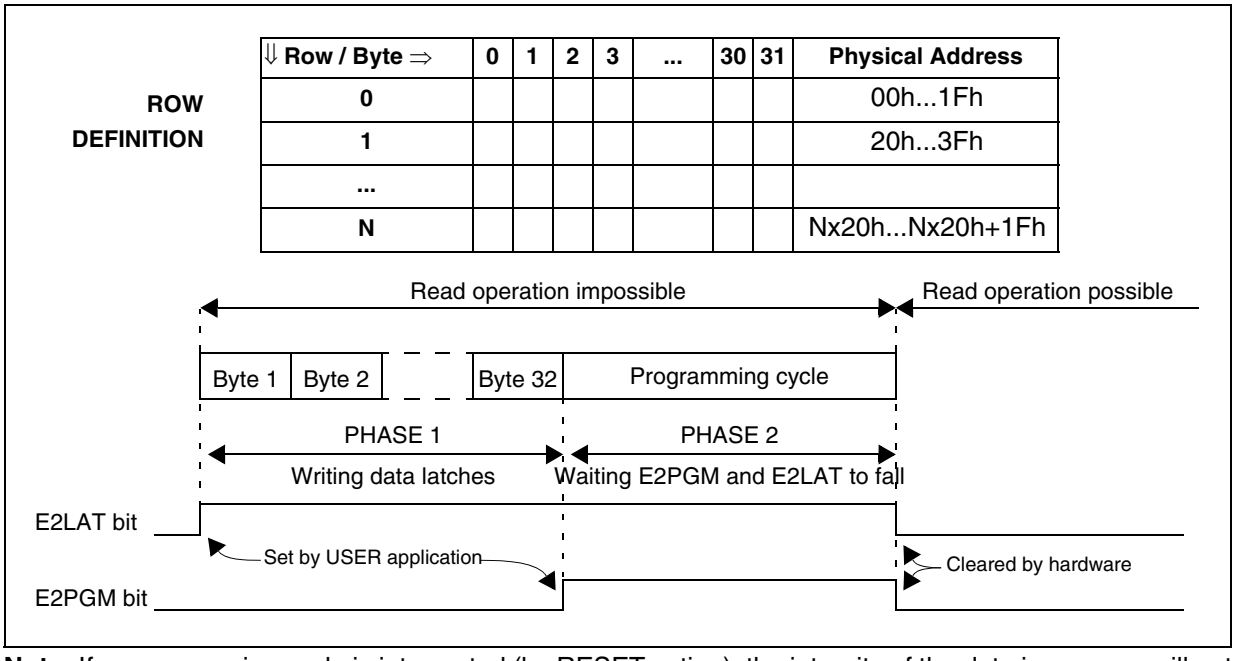
**Caution:** During normal operation, ICCCLK pin must be pulled- up, internally or externally (external pull-up of 10K mandatory in noisy environment). This is to avoid entering ICC mode unexpectedly during a reset. In the application, even if the pin is configured as output, any reset will put it back in input pull-up.

**Figure 6. Typical ICC Interface**



**DATA EEPROM** (Cont'd)

**Figure 9. Data E$^2$PROM Write Operation**




**Note:** If a programming cycle is interrupted (by RESET action), the integrity of the data in memory will not be guaranteed.

**DATA EEPROM** (Cont'd)

### 5.4 POWER SAVING MODES

#### Wait mode

The DATA EEPROM can enter WAIT mode on execution of the WFI instruction of the microcontroller or when the microcontroller enters Active Halt mode.The DATA EEPROM will immediately enter this mode if there is no programming in progress, otherwise the DATA EEPROM will finish the cycle and then enter WAIT mode.

#### Active Halt mode

Refer to Wait mode.

#### Halt mode

The DATA EEPROM immediately enters HALT mode if the microcontroller executes the HALT instruction. Therefore the EEPROM will stop the function in progress, and data may be corrupted.

### 5.5 ACCESS ERROR HANDLING

If a read access occurs while E2LAT = 1, then the data bus will not be driven.

If a write access occurs while E2LAT = 0, then the data on the bus will not be latched.

If a programming cycle is interrupted (by RESET action), the integrity of the data in memory will not be guaranteed.

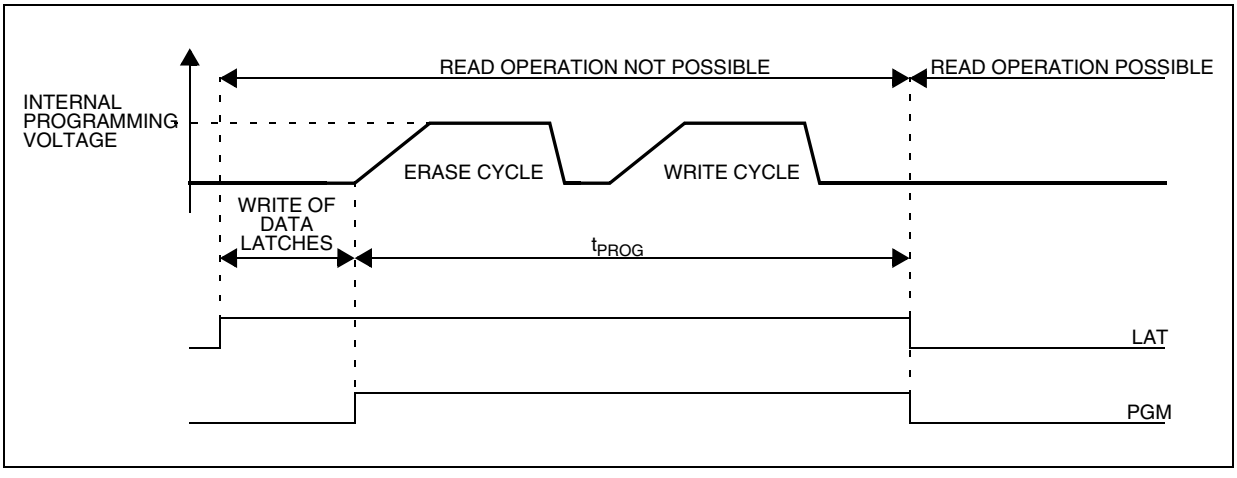
### 5.6 DATA EEPROM READ-OUT PROTECTION

The read-out protection is enabled through an option bit (see option byte section).

When this option is selected, the programs and data stored in the EEPROM memory are protected against read-out (including a re-write protection). In Flash devices, when this protection is removed by reprogramming the Option Byte, the entire Program memory and EEPROM is first automatically erased.

**Note:** Both Program Memory and data EEPROM are protected using the same option bit.

**Figure 10. Data EEPROM Programming Cycle**



**DATA EEPROM** (Cont'd)

### 5.7 REGISTER DESCRIPTION

#### EEPROM CONTROL/STATUS REGISTER (EECSR)

Read/Write

Reset Value: 0000 0000 (00h)

| 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | E2LAT | E2PGM |

Bits 7:2 = Reserved, forced by hardware to 0.

Bit 1 = **E2LAT** *Latch Access Transfer*
This bit is set by software. It is cleared by hardware at the end of the programming cycle. It can only be cleared by software if the E2PGM bit is cleared.
0: Read mode
1: Write mode

Bit 0 = **E2PGM** *Programming control and status*
This bit is set by software to begin the programming cycle. At the end of the programming cycle, this bit is cleared by hardware.
0: Programming finished or not yet started
1: Programming cycle is in progress

**Note**: If the E2PGM bit is cleared during the programming cycle, the memory data is not guaranteed.

**Table 4. DATA EEPROM Register Map and Reset Values**

| Address (Hex.) | Register Label | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0030h | **EECSR** | | | | | | | E2LAT | E2PGM |
| | Reset Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**RESET SEQUENCE MANAGER** (Cont'd)

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

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

A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see Figure 17). 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 the electrical characteristics section.

### 7.4.3 External Power-On RESET

If the LVD is disabled by option byte, to start up the microcontroller correctly, the user must ensure by means of an external reset circuit that the reset signal is held low until $V_{DD}$ 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.

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

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

- Power-On RESET
- Voltage Drop RESET

The device $\overline{\text{RESET}}$ pin acts as an output that is pulled low when $V_{DD}<V_{IT+}$ (rising edge) or $V_{DD}<V_{IT-}$ (falling edge) as shown in Figure 17.

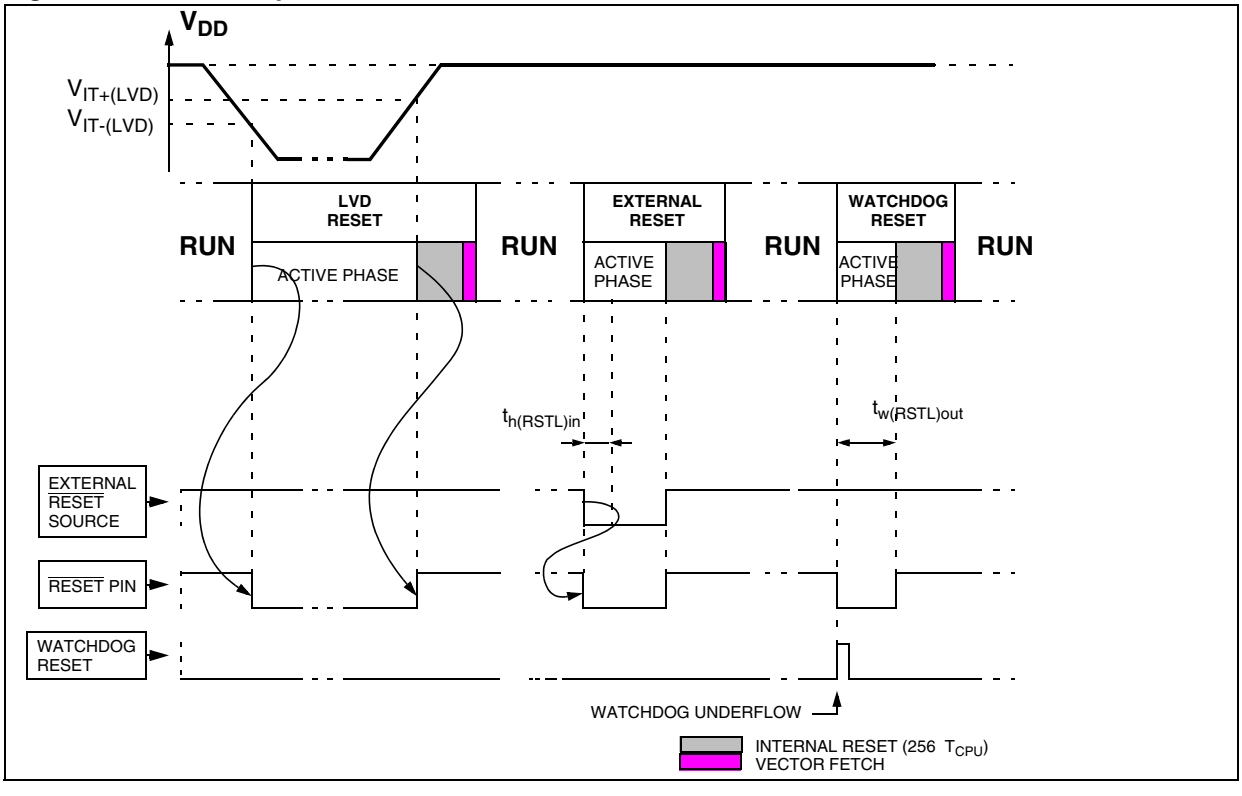
The LVD filters spikes on $V_{DD}$ larger than $t_{g(VDD)}$ to avoid parasitic resets.

### 7.4.5 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter overflow is shown in Figure 17.

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 17. RESET Sequences**



**POWER SAVING MODES** (Cont'd)

#### 9.4.2.1 HALT Mode Recommendations

– Make sure that an external event is available to wake up the microcontroller from Halt mode.

– When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as "Input Pull-up with Interrupt" before executing the HALT instruction. The main reason for this is that the I/O may be wrongly configured due to external interference or by an unforeseen logical condition.

– For the same reason, reinitialize the level 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 I bit 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).

**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.

### 10.3 UNUSED I/O PINS

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

### 10.4 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. |

### 10.5 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).

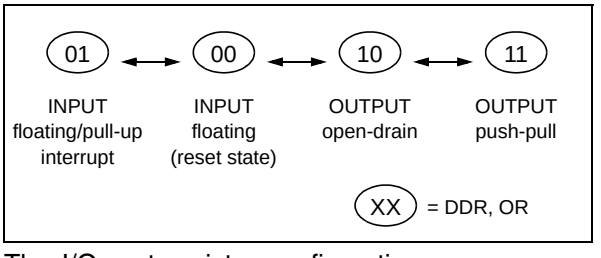
| Interrupt Event | Event Flag | Enable Control Bit | Exit from Wait | Exit from Halt |
|---|---|---|---|---|
| External interrupt on selected external event | - | DDRx ORx | Yes | Yes |

### 10.6 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.

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 30 Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

**Figure 30. Interrupt I/O Port State Transitions**




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

**Table 11. Port Configuration**

| Port | Pin name | Input (DDR=0) | | Output (DDR=1) | |
|---|---|---|---|---|---|
| | | OR = 0 | OR = 1 | OR = 0 | OR = 1 |
| Port A | PA7 | floating | pull-up interrupt | open drain | push-pull |
| | PA6:1 | floating | pull-up | open drain | push-pull |
| | PA0 | floating | pull-up interrupt | open drain | push-pull |
| Port B | PB4 | floating | pull-up | open drain | push-pull |
| | PB3 | floating | pull-up interrupt | open drain | push-pull |
| | PB2:1 | floating | pull-up | open drain | push-pull |
| | PB0 | floating | pull-up interrupt | open drain | push-pull |

# 11 ON-CHIP PERIPHERALS

## 11.1 LITE TIMER (LT)

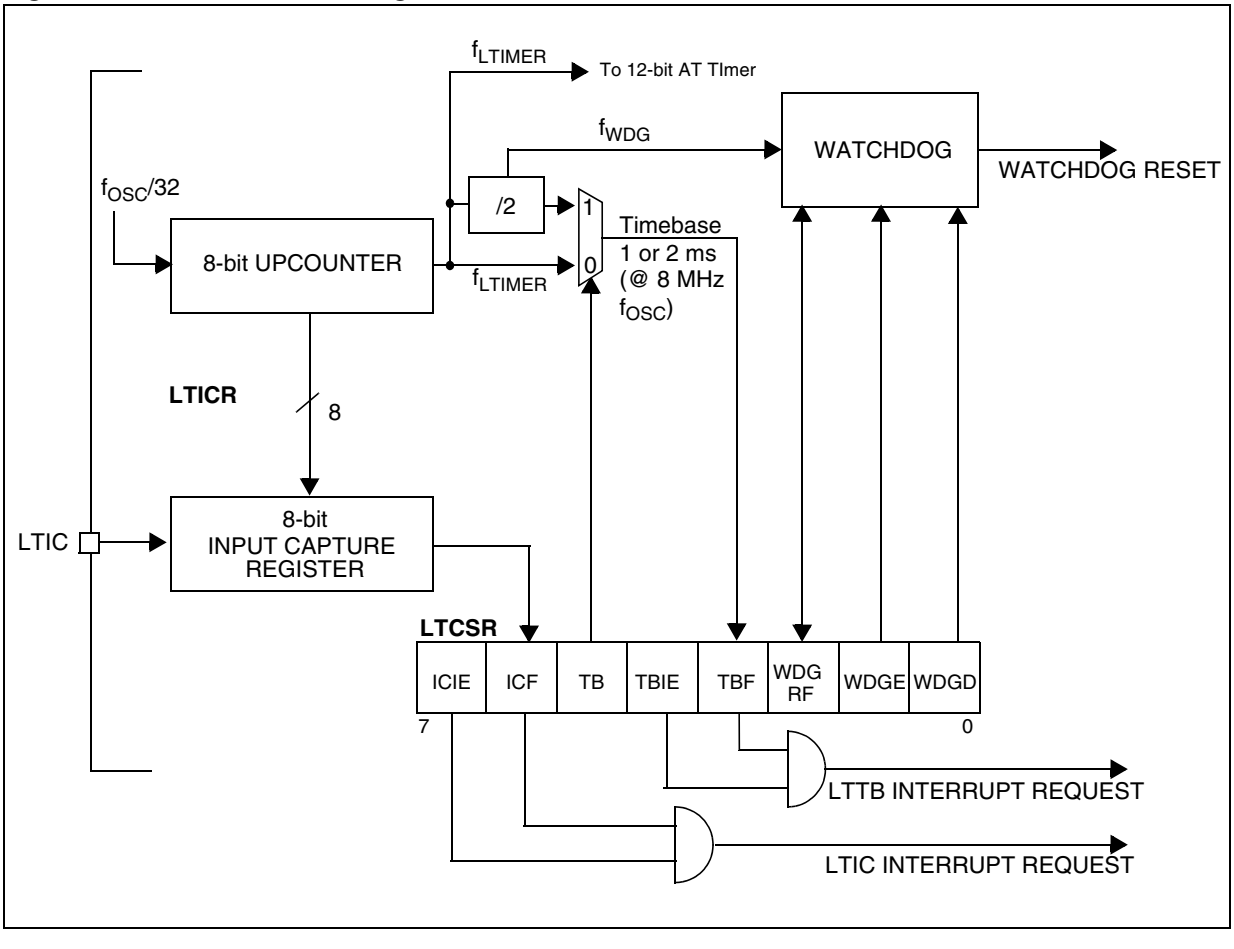
### 11.1.1 Introduction

The Lite Timer can be used for general-purpose timing functions. It is based on a free-running 8-bit upcounter with two software-selectable timebase periods, an 8-bit input capture register and watchdog function.

### 11.1.2 Main Features

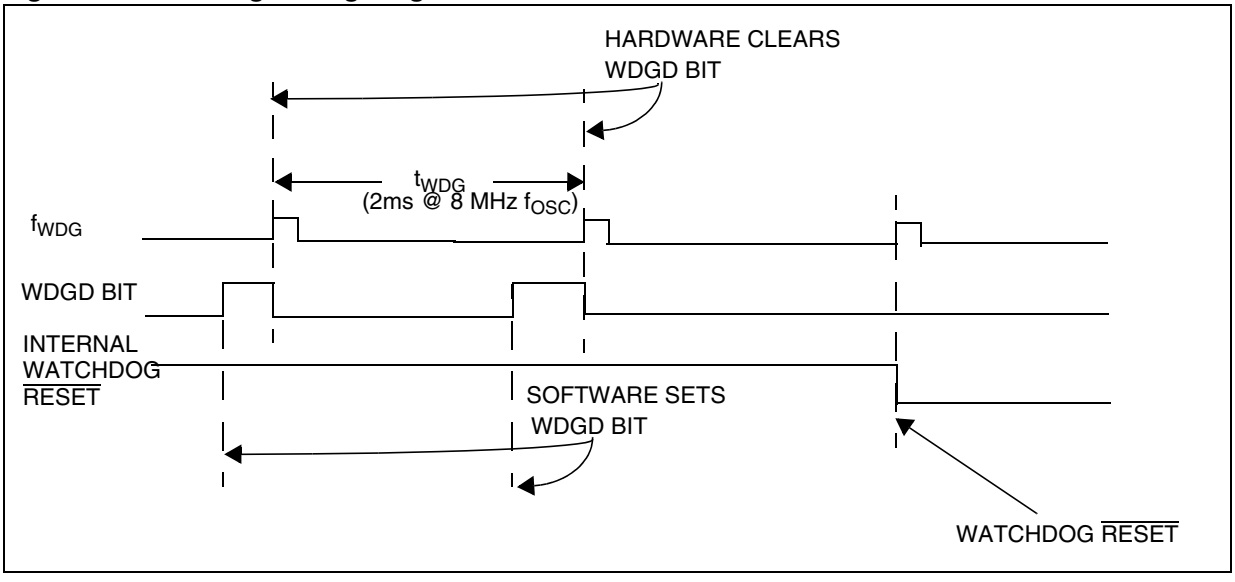
- Realtime Clock
  - 8-bit upcounter
  - 1 ms or 2 ms timebase period (@ 8 MHz $f_{OSC}$)
  - Maskable timebase interrupt
- Input Capture
  - 8-bit input capture register (LTICR)
  - Maskable interrupt with wakeup from Halt Mode capability
- Watchdog
  - Enabled by hardware or software (configurable by option byte)
  - Optional reset on HALT instruction (configurable by option byte)
  - Automatically resets the device unless disable bit is refreshed
  - Software reset (Forced Watchdog reset)
  - Watchdog reset status flag

**Figure 31. Lite Timer Block Diagram**



**LITE TIMER** (Cont'd)

**Figure 32. Watchdog Timing Diagram**



## 11.2 12-BIT AUTORELOAD TIMER (AT)

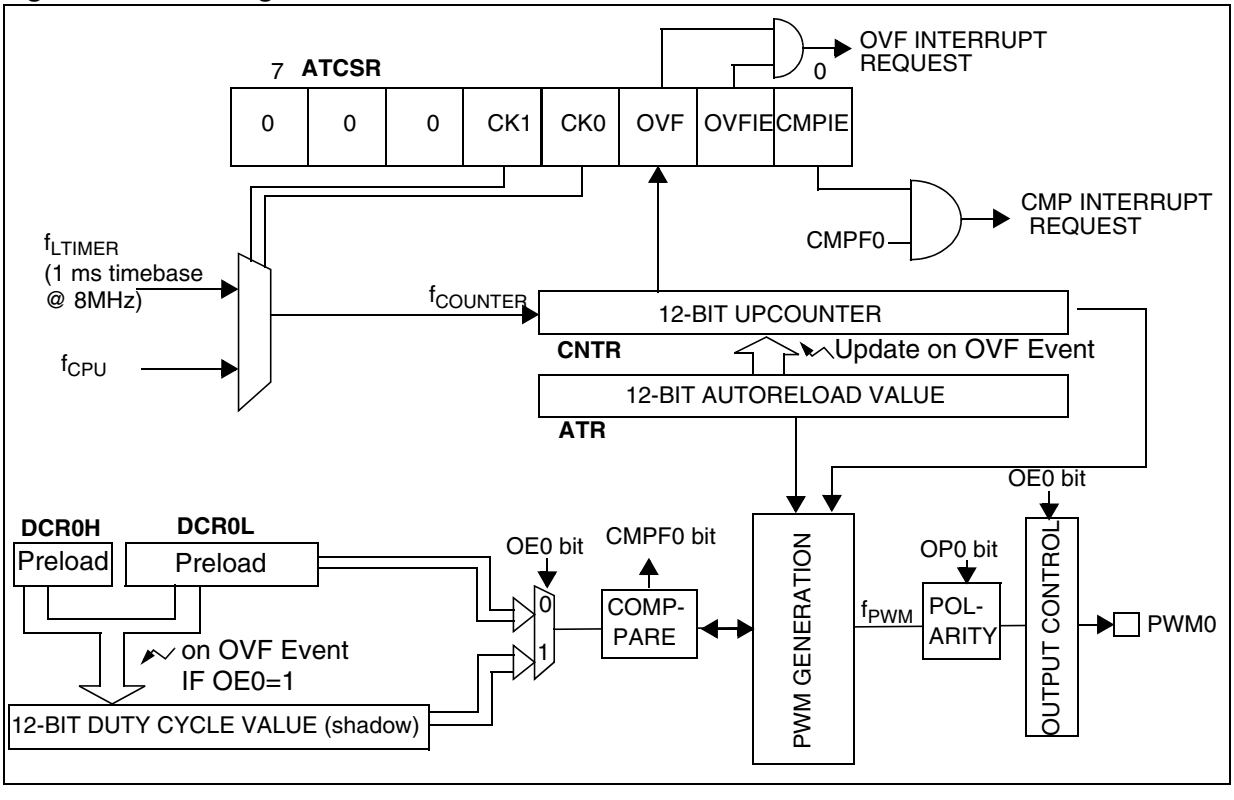
### 11.2.1 Introduction

The 12-bit Autoreload Timer can be used for general-purpose timing functions. It is based on a free-running 12-bit upcounter with a PWM output channel.

### 11.2.2 Main Features

- 12-bit upcounter with 12-bit autoreload register (ATR)
- Maskable overflow interrupt
- PWM signal generator
- Frequency range 2KHz-4MHz (@ 8 MHz $f_{CPU}$)
  - Programmable duty-cycle
  - Polarity control
  - Maskable Compare interrupt
- Output Compare Function

**Figure 34. Block Diagram**



**12-BIT AUTORELOAD TIMER** (Cont'd)

### 11.2.6 Register Description

**TIMER CONTROL STATUS REGISTER (ATCSR)**
Read / Write
Reset Value: 0000 0000 (00h)

| 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | CK1 | CK0 | OVF | OVFIE | CMPIE |

Bit 7:5 = Reserved, must be kept cleared.

Bit 4:3 = **CK[1:0]** *Counter Clock Selection.*
These bits are set and cleared by software and cleared by hardware after a reset. They select the clock frequency of the counter.

| Counter Clock Selection | CK1 | CK0 |
|---|---|---|
| OFF | 0 | 0 |
| $f_{LTIMER}$ (1 ms timebase @ 8 MHz) | 0 | 1 |
| $f_{CPU}$ | 1 | 0 |
| Reserved | 1 | 1 |

Bit 2 = **OVF** *Overflow Flag.*
This bit is set by hardware and cleared by software by reading the ATCSR register. It indicates the transition of the counter from FFFh to ATR value.
0: No counter overflow occurred
1: Counter overflow occurred

**Caution:**
When set, the OVF bit stays high for 1 $f_{COUNTER}$ cycle, (up to 1ms depending on the clock selection).

Bit 1 = **OVFIE** *Overflow Interrupt Enable.*
This bit is read/write by software and cleared by hardware after a reset.
0: OVF interrupt disabled
1: OVF interrupt enabled

Bit 0 = **CMPIE** *Compare Interrupt Enable*.
This bit is read/write by software and clear by hardware after a reset. It allows to mask the interrupt generation when CMPF bit is set.
0: CMPF interrupt disabled
1: CMPF interrupt enabled

**COUNTER REGISTER HIGH (CNTRH)**
Read only
Reset Value: 0000 0000 (00h)

| 15 | | | | | | | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | CN11 | CN10 | CN9 | CN8 |

**COUNTER REGISTER LOW (CNTRL)**
Read only
Reset Value: 0000 0000 (00h)

| 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|
| CN7 | CN6 | CN5 | CN4 | CN3 | CN2 | CN1 | CN0 |

Bits 15:12 = Reserved, must be kept cleared.

Bits 11:0 = **CNTR[11:0]** *Counter Value*.
This 12-bit register is read by software and cleared by hardware after a reset. The counter is incremented continuously as soon as a counter clock is selected. To obtain the 12-bit value, software should read the counter value in two consecutive read operations. The CNTRH register can be incremented between the two reads, and in order to be accurate when $f_{TIMER}=f_{CPU}$, the software should take this into account when CNTRL and CNTRH are read. If CNTRL is close to its highest value, CNTRH could be incremented before it is read.

When a counter overflow occurs, the counter restarts from the value specified in the ATR register.

**Table 17. ADC Register Map and Reset Values**

| Address (Hex.) | Register Label | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| 34h | **ADCCSR** | EOC | SPEED | ADON | | | CH2 | CH1 | CH0 |
| | Reset Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 35h | **ADCDR** | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| | Reset Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 36h | **ADCAMP** | 0 | 0 | 0 | 0 | SLOW | AMPSEL | 0 | 0 |
| | Reset Value | | | | | 0 | 0 | | |

## 12.2 INSTRUCTION GROUPS

The ST7 family devices use an Instruction Set consisting of 63 instructions. The instructions may be subdivided into 13 main groups as illustrated in the following table:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Load and Transfer | LD | CLR | | | | | | |
| Stack operation | PUSH | POP | RSP | | | | | |
| Increment/Decrement | INC | DEC | | | | | | |
| Compare and Tests | CP | TNZ | BCP | | | | | |
| Logical operations | AND | OR | XOR | CPL | NEG | | | |
| Bit Operation | BSET | BRES | | | | | | |
| Conditional Bit Test and Branch | BTJT | BTJF | | | | | | |
| Arithmetic operations | ADC | ADD | SUB | SBC | MUL | | | |
| Shift and Rotates | SLL | SRL | SRA | RLC | RRC | SWAP | SLA | |
| Unconditional Jump or Call | JRA | JRT | JRF | JP | CALL | CALLR | NOP | RET |
| Conditional Branch | JRxx | | | | | | | |
| Interruption management | TRAP | WFI | HALT | IRET | | | | |
| Condition Code Flag modification | SIM | RIM | SCF | RCF | | | | |

**Using a prebyte**

The instructions are described with 1 to 4 bytes.

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.

### 12.2.1 Illegal Opcode Reset

In order to provide enhanced robustness to the device against unexpected behavior, a system of illegal opcode detection is implemented. If a code to be executed does not correspond to any opcode or prebyte value, a reset is generated. This, combined with the Watchdog, allows the detection and recovery from an unexpected fault or interference.

**Note:** A valid prebyte associated with a valid opcode forming an unauthorized combination does not generate a reset.

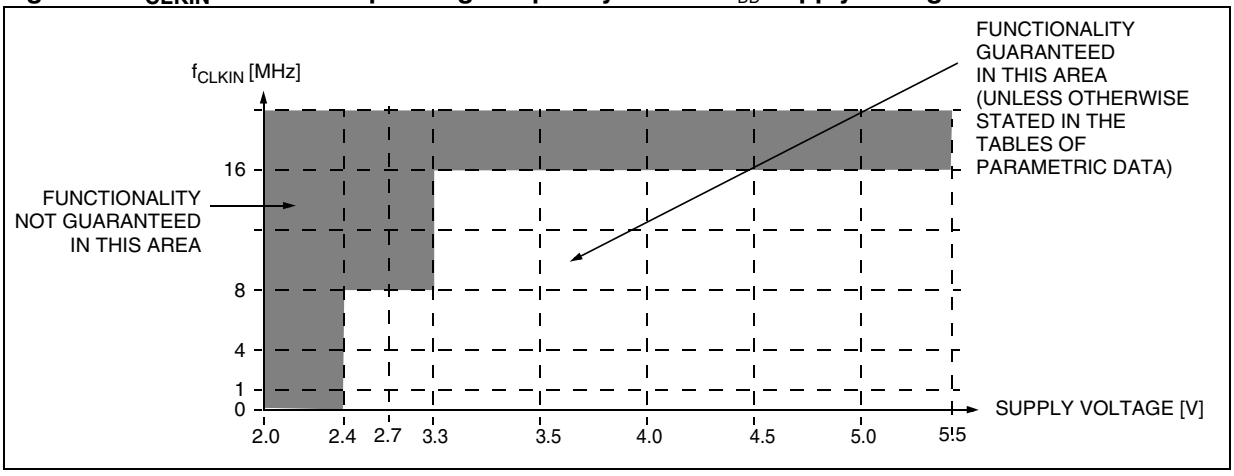
## 13.3 OPERATING CONDITIONS

### 13.3.1 General Operating Conditions: Suffix 6 Devices

$T_A$ = -40 to +85°C unless otherwise specified.

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{DD}$ | Supply voltage | $f_{OSC}$ = 8 MHz. max., | 2.4 | 5.5 | V |
| | | $f_{OSC}$ = 16 MHz. max. | 3.3 | 5.5 | |
| $f_{CLKIN}$ | External clock frequency on CLKIN pin | 3.3V≤ $V_{DD}$≤5.5V | up to 16 | | MHz |
| | | 2.4V≤$V_{DD}$<3.3V | up to 8 | | |

**Figure 48. $f_{CLKIN}$ Maximum Operating Frequency Versus $V_{DD}$ Supply Voltage**




**Note:** For further information on clock management and $f_{CLKIN}$ description, refer to Figure 14 in section 7 on page 24

## 13.5 CLOCK AND TIMING CHARACTERISTICS

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

### 13.5.1 General Timings

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

### 13.5.2 External Clock Source

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{CLKINH}$ | CLKIN input pin high level voltage | see Figure 61 | 0.7x$V_{DD}$ | | $V_{DD}$ | V |
| $V_{CLKINL}$ | CLKIN input pin low level voltage | | $V_{SS}$ | | 0.3x$V_{DD}$ | |
| $t_{w(CLKINH)}$<br>$t_{w(CLKINL)}$ | CLKIN high or low time [4] | | 15 | | | ns |
| $t_{r(CLKIN)}$<br>$t_{f(CLKIN)}$ | CLKIN rise or fall time [4] | | | | 15 | |
| $I_L$ | CLKIN Input leakage current | $V_{SS} \le V_{IN} \le V_{DD}$ | | | ±1 | µA |

**Notes:**

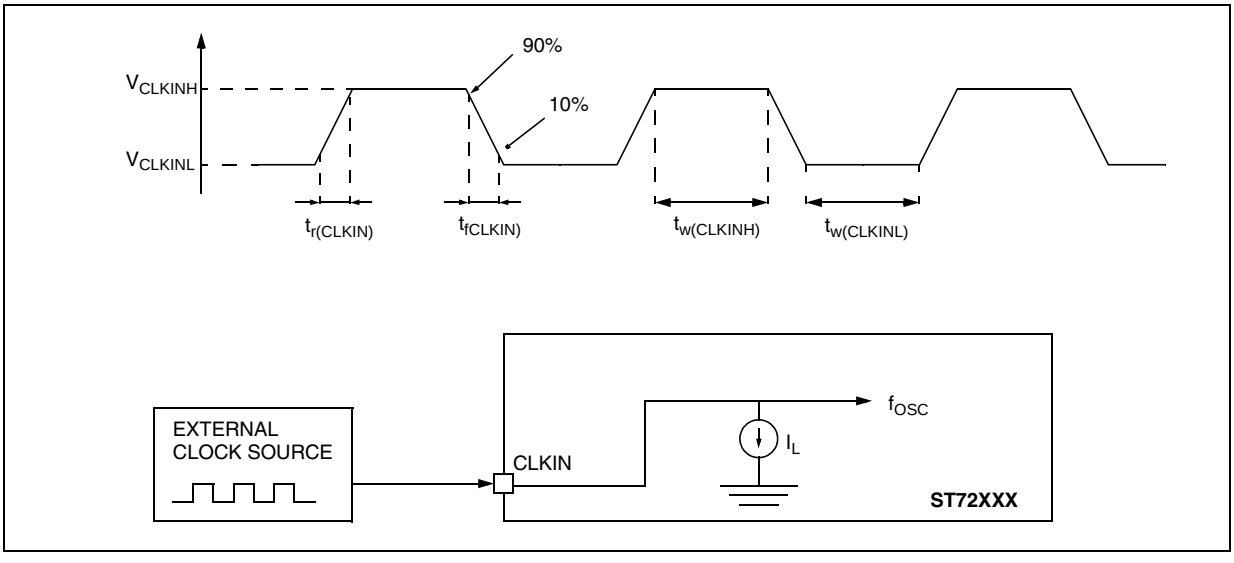
**1.** Guaranteed by Design. Not tested in production.

**2.** Data based on typical application software.

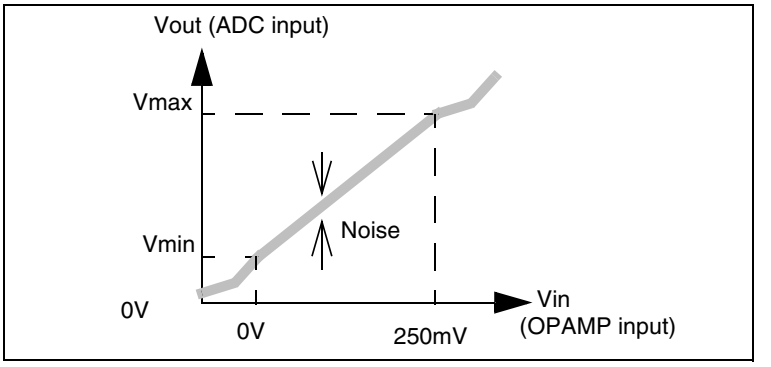
**3.** Time measured between interrupt event and interrupt vector fetch. $\Delta t_{c(INST)}$ is the number of $t_{CPU}$ cycles needed to finish the current instruction execution.

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

**Figure 61. Typical Application with an External Clock Source**



**ADC CHARACTERISTICS** (Cont'd)




| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DD(AMP)}$ | Amplifier operating voltage | | 4.5 | | 5.5 | V |
| $V_{IN}$ | Amplifier input voltage | $V_{DD}$=5V | 0 | | 250 | mV |
| $V_{OFFSET}$ | Amplifier offset voltage | | | 200 | | mV |
| $V_{STEP}$ | Step size for monotonicity[3)] | | 5 | | | mV |
| Linearity | Output Voltage Response | | Linear | | | |
| Gain factor | Amplified Analog input Gain[2)] | | 7[1)] | 8 | 9[1)] | |
| Vmax | Output Linearity Max Voltage | $V_{INmax}$ = 250mV, $V_{DD}$=5V | | 2.2 | 2.4 | V |
| Vmin | Output Linearity Min Voltage | | | 200 | | mV |

**Notes:**

**1.** Data based on characterization results over the whole temperature range, not tested in production.

**2.** For precise conversion results it is recommended to calibrate the amplifier at the following two points:

– offset at $V_{INmin}$ = 0V

– gain at full scale (for example $V_{IN}$=250mV)

**3.** Monotonicity guaranteed if $V_{IN}$ increases or decreases in steps of min. 5mV.

# 17 REVISION HISTORY

**Table 25. Revision History**

| Date | Revision | Description of changes |
|---|---|---|
| 27-Oct-04 | 3 | Revision number incremented from 2.5 to 3.0 due to Internal Document Management System change<br>Changed all references of ADCDAT to ADCDR<br>Added EMU3 Emulator Programming Capability in Table 23<br>Clarification of read-out protection<br>Altered note 1 for section 13.2.3 on page 82 removing references to $\overline{\text{RESET}}$<br>Alteration of $f_{CPU}$ for SLOW and SLOW-WAIT modes in Section 13.4.1 table and Figure 59 on page 90<br>Removed sentence relating to an effective change only after overflow for CK[1:0], page 56<br>Added illegal opcode detection to page 1, section 8.4 on page 32, section 12 on page 75<br>Clarification of Flash read-out protection, section 4.5.1 on page 15<br>$f_{PLL}$ value of 1MHz quoted as Typical instead of a Minimum in section 14.3.5.2 on page 92<br>Updated $F_{SCK}$ in section 13.10.1 on page 102 to $F_{CPU}/4$ and $F_{CPU}/2$<br>section 8.4.4 on page 36: Changed wording in AVDIE and AVDF bit descriptions to "...when the AVDF bit is set"<br>Socket Board development kit details added in Table 24 on page 115<br>PWM Signal diagram corrected, Figure 36 on page 55<br>Corrected count of reserved bits between 003Bh to 007Fh, Table 2 on page 11<br>Inserted note that RCCR0 and RCCR1 are erased if read-only flag is reset, section 7.1 on page 24 |
| 21-July-06 | 4 | Added QFN20 package<br>Modified section 2 on page 6<br>Changed Read operation paragraph in section 5.3 on page 17<br>Modified note below Figure 9 on page 18 and modified section 5.5 on page 19<br>Modified note to section 7.1 on page 24<br>Added note on illegal opcode reset to section 7.4.1 on page 27<br>Added note 2 to EICR description on page 31<br>Modified External Interrupt Function in section 10.2.1 on page 42<br>Changed text on input capture before section 11.1.4 on page 51<br>Modified text in section 11.1.5 on page 51<br>Added important note in section 11.3.3.3 on page 62<br>Changed note 1 in section 13.2 on page 82<br>Modified values in section 13.2.2 on page 82<br>Modified note 2 in section 13.3.4.1 on page 85 and section 13.3.4.2 on page 86<br>Added note on clock stability and frequency accuracy to section 13.3.4.1 on page 85, section 13.3.4.2 on page 86, section 7.1 on page 24 and to OSC option bit in Section 15.1 on page 113<br>Changed $I_S$ value and note 2 in section 13.8.1 on page 95<br>Added note in Figure 62 on page 95<br>Changed Figure 76 on page 101 and removed EMC protection circuitry in Figure 77 on page 101 (device works correctly without these components)<br>Changed section 13.10.1 on page 102 ($t_{su(\overline{SS})}$, $t_{v(MO)}$ and $t_{h(MO)}$)<br>Modified Figure 79 (CPHA=1) and Figure 80 on page 103 ($t_{v(MO)}$ , $t_{h(MO)}$)<br>Added ECOPACK information to section 14 on page 109<br>Modified Figure 88 on page 110 (A1 and A swapped in the diagram)<br>Modified Table 21 on page 112<br>Modified section 15.2 on page 114<br>Updated option list on page 116<br>Changed section 15.3 on page 117<br>Removed erratasheet section<br>Added Section 16.4 and section 16.5 on page 121 |
| | | Revision History continued overleaf ... |