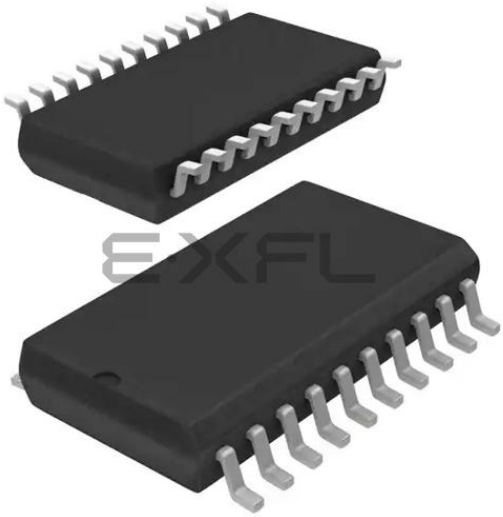


Welcome to E-XFL.COM

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	16MHz
Connectivity	LINbusSCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	15
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	384 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SOIC (0.295", 7.50mm Width)
Supplier Device Package	20-SO
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7flite39f2m6tr

To obtain the most recent version of this datasheet,
please check at www.st.com

Please also pay special attention to the Section “KNOWN LIMITATIONS” on page 169.

1 INTRODUCTION

The ST7LITE3 is a member of the ST7 microcontroller family. All ST7 devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

The ST7LITE3 features FLASH memory with byte-by-byte In-Circuit Programming (ICP) and In-Application Programming (IAP) capability.

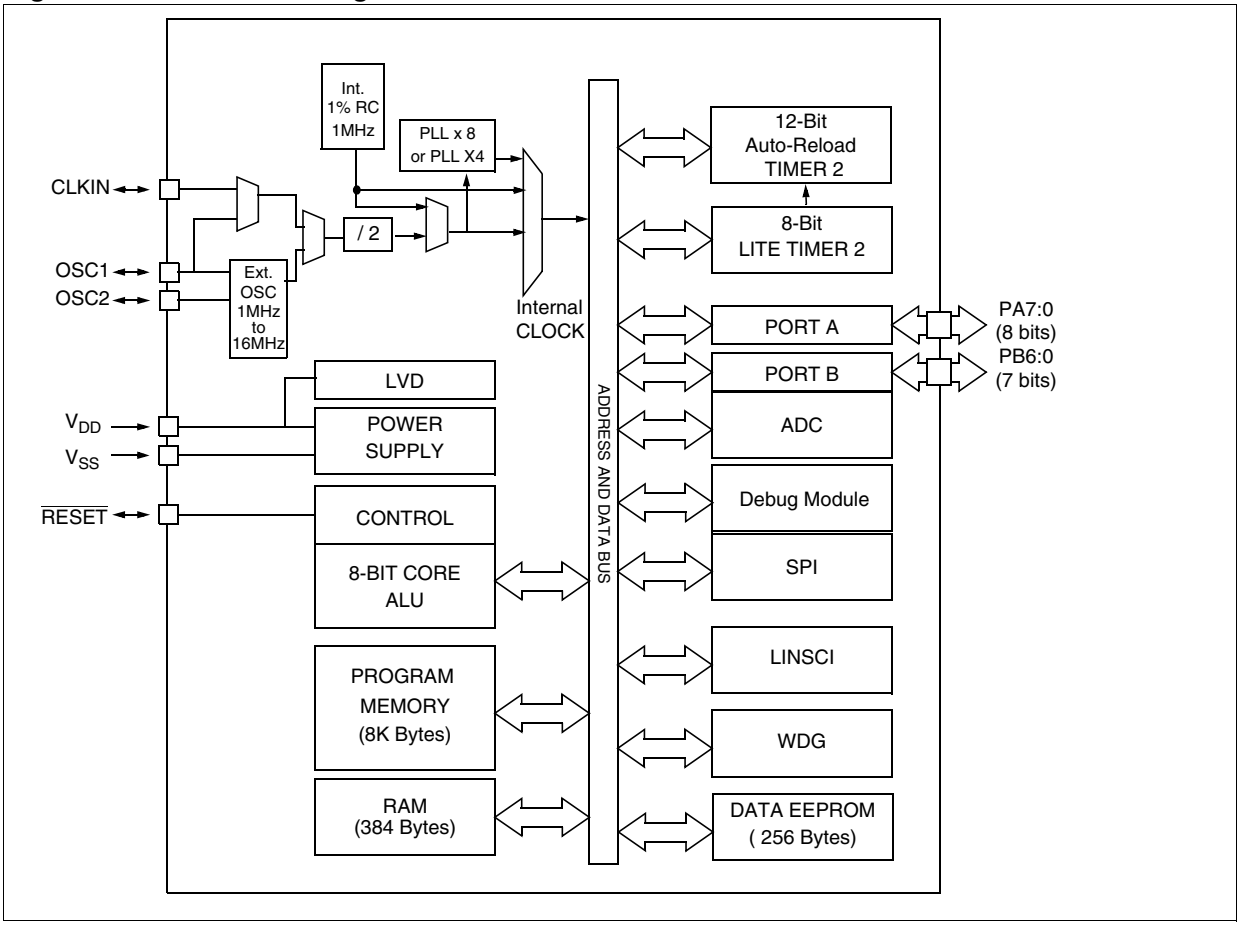
Under software control, the ST7LITE3 device can be placed in WAIT, SLOW, or HALT mode, reducing power consumption when the application is in idle or standby state.

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 131.

The devices feature an on-chip Debug Module (DM) to support in-circuit debugging (ICD). For a description of the DM registers, refer to the ST7 ICC Protocol Reference Manual.

Figure 1. General Block Diagram



2 PIN DESCRIPTION

Figure 2. 20-Pin QFN Package Pinout

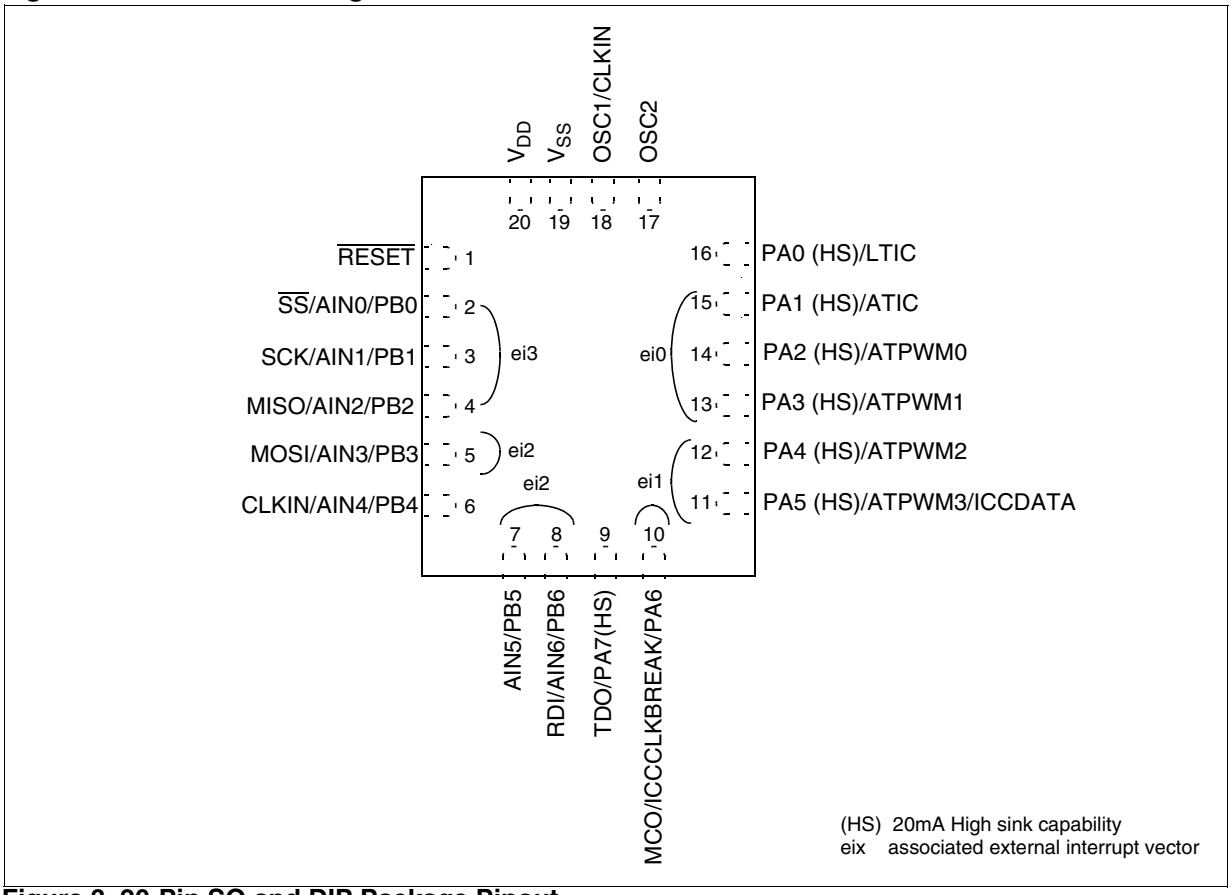
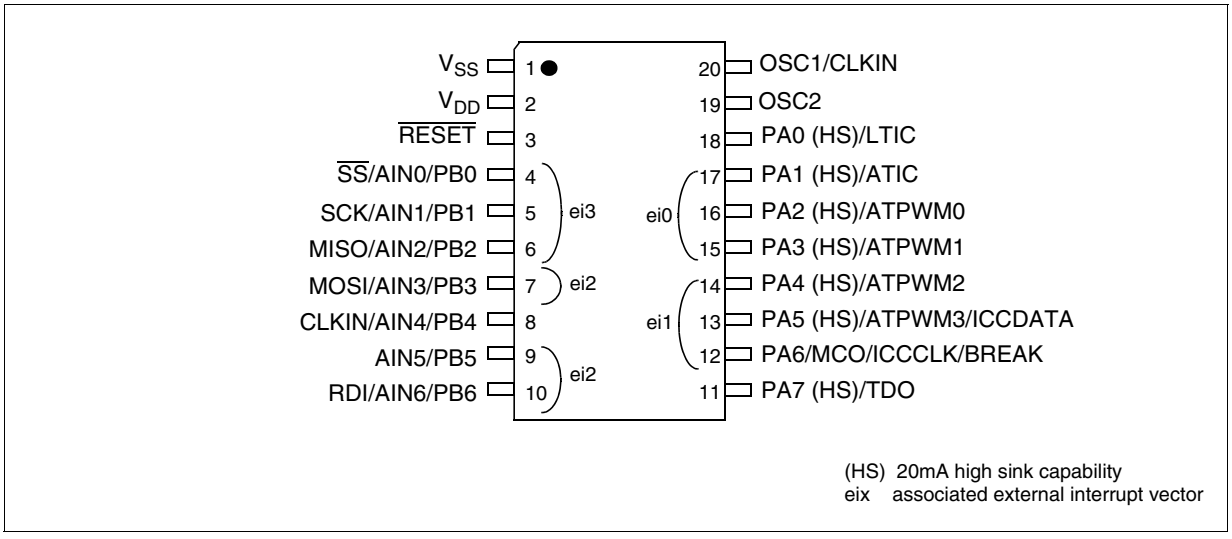
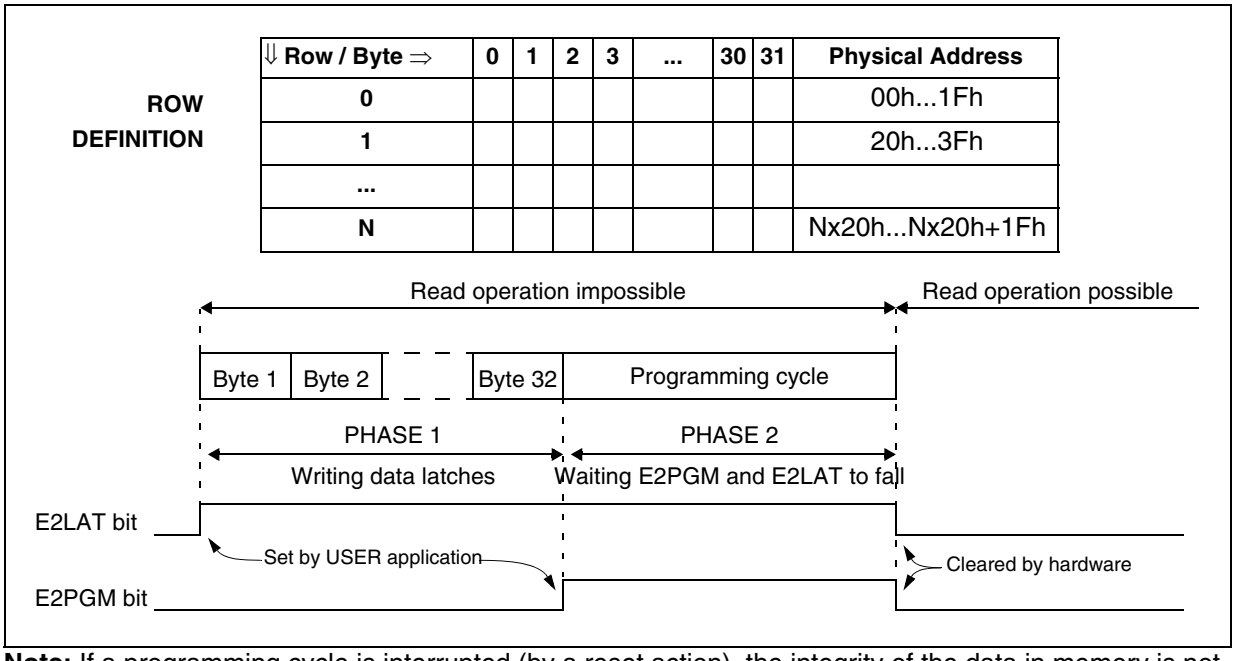


Figure 3. 20-Pin SO and DIP Package Pinout



DATA EEPROM (Cont'd)

Figure 8. Data E^2PROM Write Operation



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

CPU REGISTERS (cont'd)

CONDITION CODE REGISTER (CC)

Read/Write

Reset Value: 111x1xxx

7							0
1	1	1	H	I	N	Z	C

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

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

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

This bit is set by hardware when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instruction. It is reset by hardware during the same instructions.
0: No half carry has occurred.
1: A half carry has occurred.

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

Bit 3 = **I** *Interrupt mask*

This bit is set by hardware when entering in interrupt or by software to disable all interrupts except the TRAP software interrupt. This bit is cleared by software.
0: Interrupts are enabled.
1: Interrupts are disabled.

This bit is controlled by the RIM, SIM and IRET instructions and is tested by the JRM and JRNM instructions.

Note: Interrupts requested while I is set are latched and can be processed when I is cleared. By default an interrupt routine is not interruptible because the I bit is set by hardware at the start of the routine and reset by the IRET instruction at the end of the routine. If the I bit is cleared by software in the interrupt routine, pending interrupts are serviced regardless of the priority level of the current interrupt routine.

Bit 2 = **N** *Negative*

This bit is set and cleared by hardware. It is representative of the result sign of the last arithmetic, logical or data manipulation. It is a copy of the 7th bit of the result.
0: The result of the last operation is positive or null.
1: The result of the last operation is negative (that is, the most significant bit is a logic 1).

This bit is accessed by the JRMI and JRPL instructions.

Bit 1 = **Z** *Zero*

This bit is set and cleared by hardware. This bit indicates that the result of the last arithmetic, logical or data manipulation is zero.
0: The result of the last operation is different from zero.
1: The result of the last operation is zero.

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

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

This bit is set and cleared by hardware and software. It indicates an overflow or an underflow has occurred during the last arithmetic operation.
0: No overflow or underflow has occurred.
1: An overflow or underflow has occurred.

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

CPU REGISTERS (Cont'd)

STACK POINTER (SP)

Read/Write

Reset Value: 01FFh

15							8
0	0	0	0	0	0	0	1

7							0
1	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 11).

Since the stack is 128 bytes deep, the 9 most significant bits are forced by hardware. Following an

7.5 RESET SEQUENCE MANAGER (RSM)

7.5.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 15:

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

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to section 12.2.1 on page 128 for further details.

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

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

The basic RESET sequence consists of 3 phases as shown in Figure 14:

- Active Phase depending on the RESET source
- 256 or 4096 CPU clock cycle delay (see table below)
- RESET vector fetch

Caution: When the ST7 is unprogrammed or fully erased, the Flash is blank and the RESET vector is not programmed. For this reason, it is recommended to keep the RESET pin in low state until programming mode is entered, in order to avoid unwanted behavior.

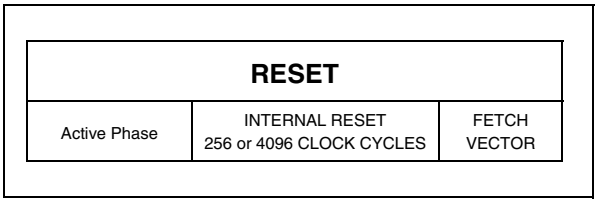
The 256 or 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay is automatically selected depending on the clock source chosen by option byte:

The RESET vector fetch phase duration is 2 clock cycles.

Clock Source	CPU clock cycle delay
Internal RC Oscillator	256
External clock (connected to CLKIN pin)	256
External Crystal/Ceramic Oscillator (connected to OSC1/OSC2 pins)	4096

If the PLL is enabled by option byte, it outputs the clock after an additional delay of $t_{STARTUP}$ (see Figure 12).

Figure 14. RESET Sequence Phases



7.5.2 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 16). This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

POWER SAVING MODES (Cont'd)

Similarities with Halt mode

The following AWUFH mode behaviour is the same as normal Halt mode:

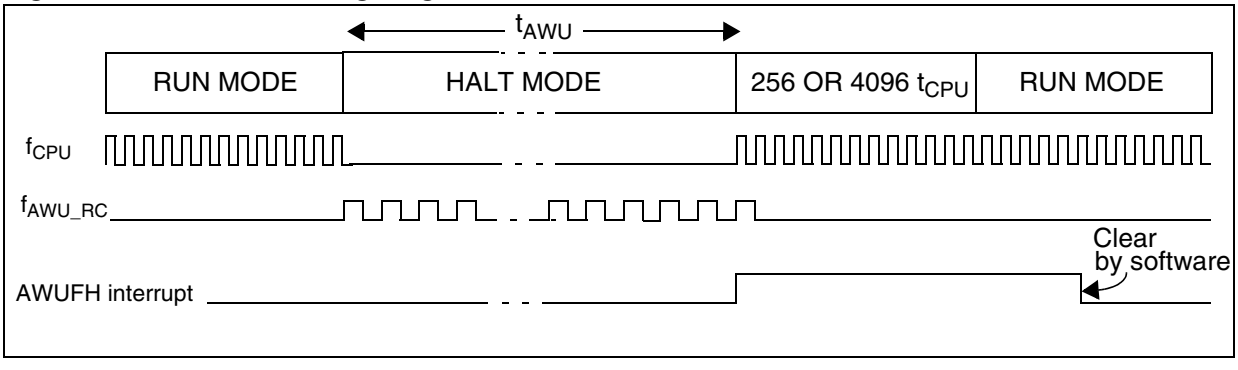
– The MCU can exit AWUFH mode by means of any interrupt with exit from Halt capability or a reset (see Section 9.4 HALT MODE).

– When entering AWUFH mode, the I bit in the CC register is forced to 0 to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately.

– In AWUFH mode, the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. None of the peripherals are clocked except those which get their clock supply from another clock generator (such as an external or auxiliary oscillator like the AWU oscillator).

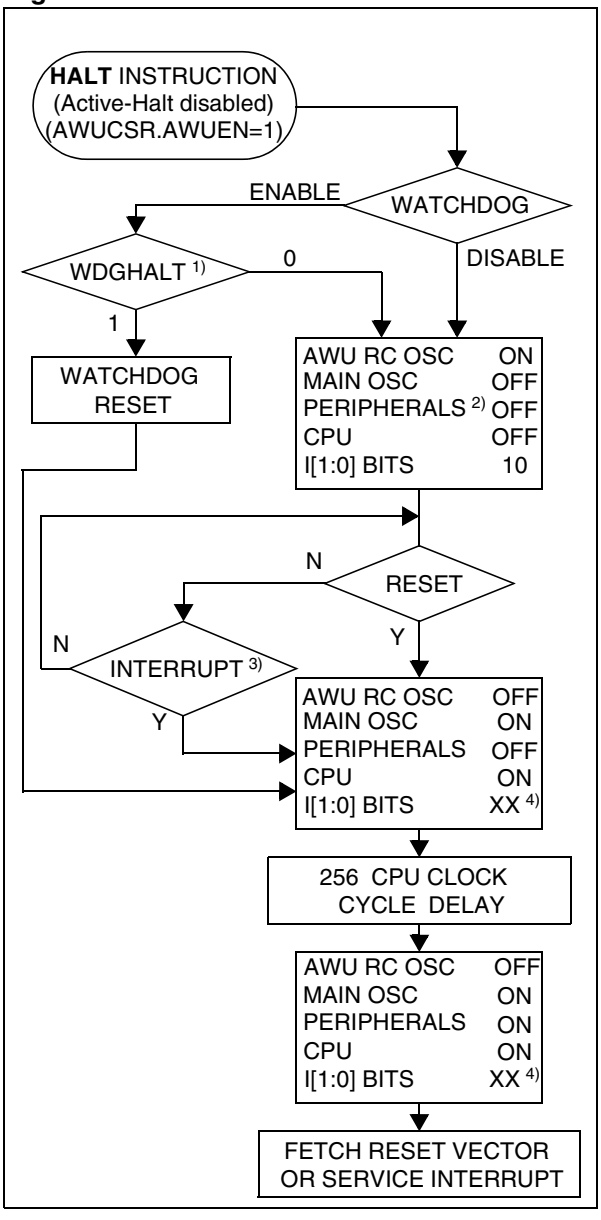
– The compatibility of Watchdog operation with AWUFH mode is configured by the WDGHALT option bit in the option byte. Depending on this setting, the HALT instruction when executed while the Watchdog system is enabled, can generate a Watchdog RESET.

Figure 29. AWUF Halt Timing Diagram



POWER SAVING MODES (Cont'd)

Figure 30. AWUFH Mode Flow-chart



Notes:

1. WDGHALT is an option bit. See option byte section for more details.

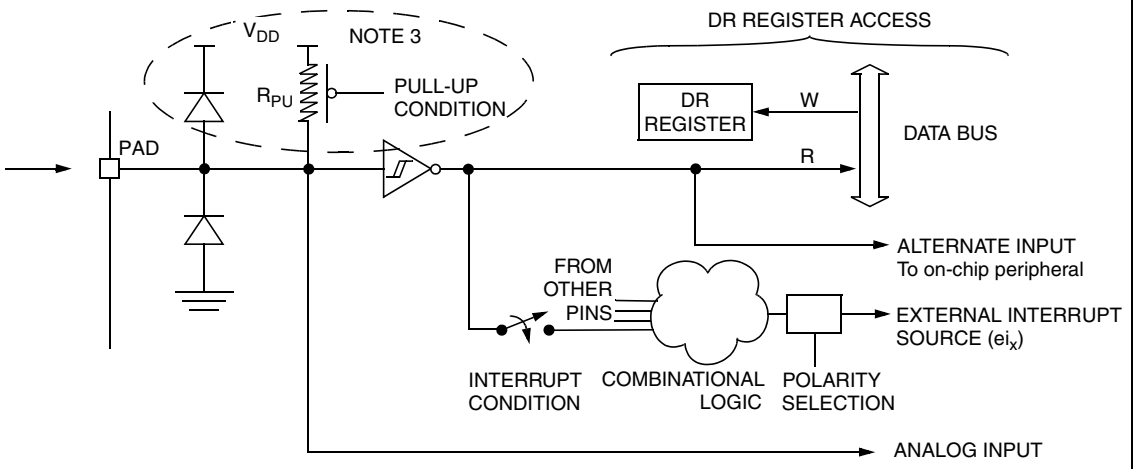
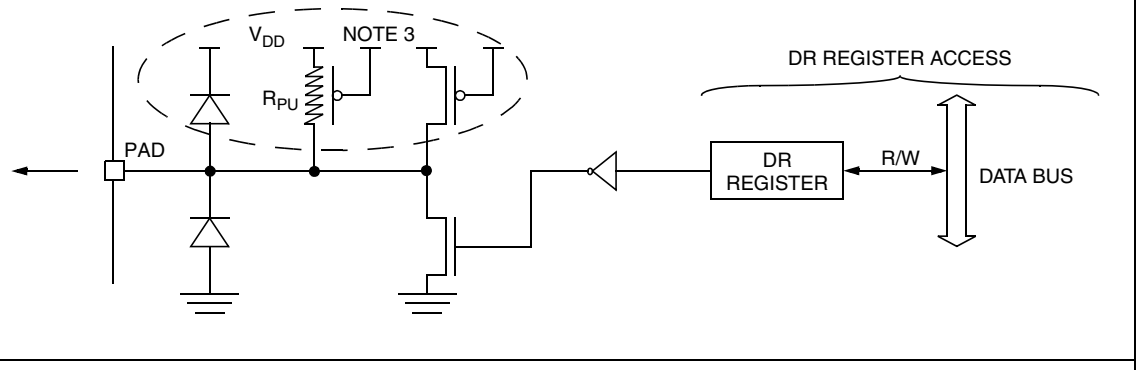
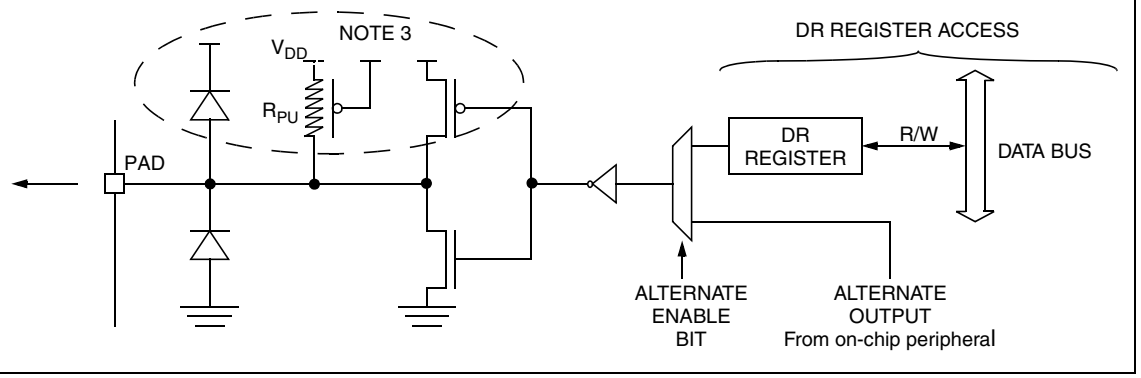
2. Peripheral clocked with an external clock source can still be active.

3. Only an AWUFH interrupt and some specific interrupts can exit the MCU from HALT mode (such as external interrupt). Refer to Table 6, "Interrupt Mapping," on page 36 for more details.

4. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped.

I/O PORTS (Cont'd)

Table 10. I/O Configurations

	Hardware Configuration
INPUT [1]	
OPEN-DRAIN OUTPUT [2]	
PUSH-PULL OUTPUT [2]	

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

DUAL 12-BIT AUTORELOAD TIMER 3 (Cont'd)

■ **Long input capture**

Pulses that last between 8µs and 2s can be measured with an accuracy of 4µs if f_{OSC} = 8MHz in the following conditions:

– The 12-bit AT3 Timer is clocked by the Lite Timer (RTC pulse: CK[1:0] = 01 in the ATCSR register)

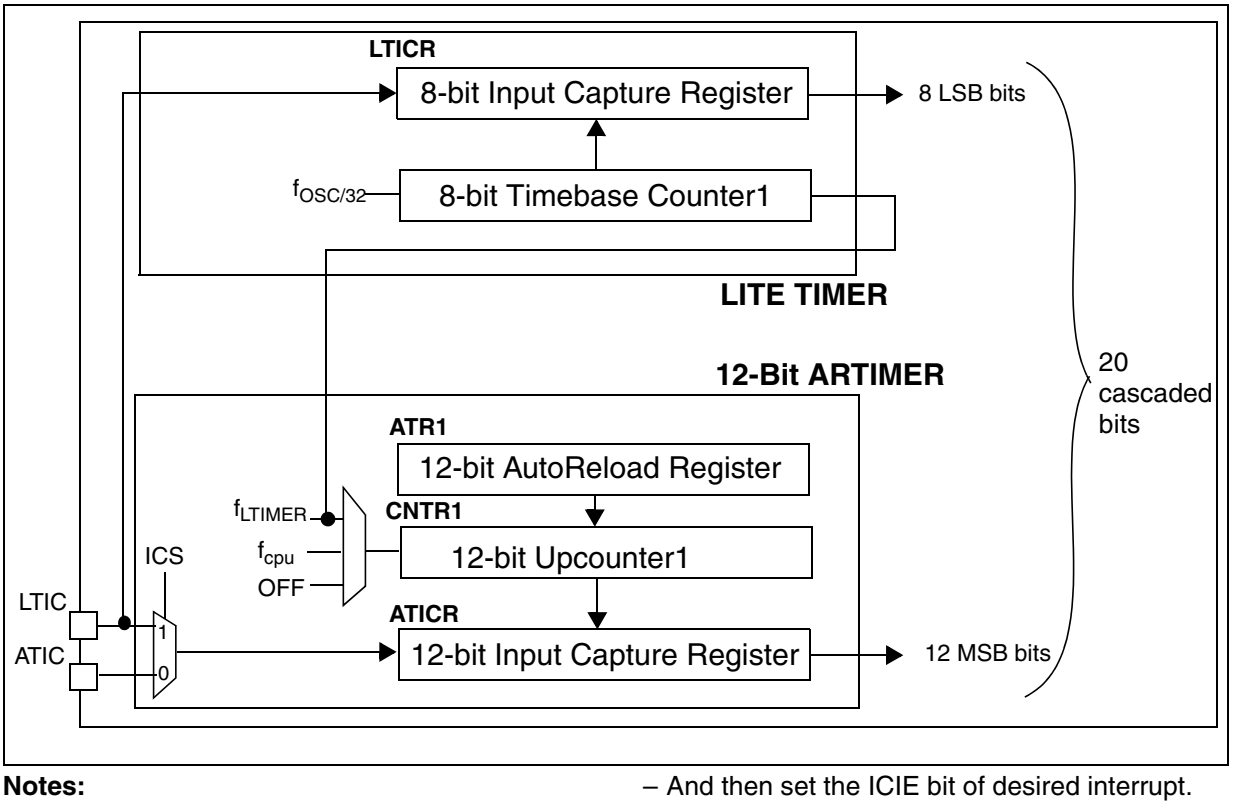
– The ICS bit in the ATCSR2 register is set so that the LTIC pin is used to trigger the AT3 Timer capture.

– The signal to be captured is connected to LTIC pin

– Input Capture registers LTICR, ATICRH and ATICRL are read

This configuration allows to cascade the Lite Timer and the 12-bit AT3 Timer to get a 20-bit input capture value. Refer to Figure 44.

Figure 44. Long Range Input Capture Block Diagram



Notes:

1. Since the input capture flags (ICF) for both timers (AT3 Timer and LT Timer) are set when signal transition occurs, software must mask one interrupt by clearing the corresponding ICIE bit before setting the ICS bit.

2. If the ICS bit changes (from 0 to 1 or from 1 to 0), a spurious transition might occur on the input capture signal because of different values on LTIC and ATIC. To avoid this situation, it is recommended to do as follows:

– First, reset both ICIE bits.

– Then set the ICS bit.

– Reset both ICF bits.

– And then set the ICIE bit of desired interrupt.

3. How to compute a pulse length with long input capture feature.

As both timers are used, computing a pulse length is not straight-forward. The procedure is as follows:

– At the first input capture on the rising edge of the pulse, we assume that values in the registers are as follows:
LTICR = LT1
ATICRH = ATH1
ATICRL = ATL1
Hence ATICR1 [11:0] = ATH1 & ATL1

Refer to Figure 45 on page 65.

DUAL 12-BIT AUTORELOAD TIMER 3 (Cont'd)

11.2.6 Register Description

TIMER CONTROL STATUS REGISTER (ATCSR)
Read / Write
Reset Value: 0x00 0000 (x0h)

7	6						0
0	ICF	ICIE	CK1	CK0	OVF1	OVFIE1	CMPIE

Bit 7 = Reserved.

Bit 6 = **ICF** *Input Capture Flag.*
This bit is set by hardware and cleared by software by reading the ATICR register (a read access to ATICRH or ATICRL will clear this flag). Writing to this bit does not change the bit value.
0: No input capture
1: An input capture has occurred

Bit 5 = **ICIE** *IC Interrupt Enable.*
This bit is set and cleared by software.
0: Input capture interrupt disabled
1: Input capture interrupt enabled

Bits 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
OFF	1	1
f_{LTIMER} (1 ms timebase @ 8 MHz)	0	1
f_{CPU}	1	0

Bit 2 = **OVF1** *Overflow Flag.*
This bit is set by hardware and cleared by software by reading the TCSR register. It indicates the transition of the counter1 CNTR1 from FFh to ATR1 value.
0: No counter overflow occurred
1: Counter overflow occurred

Bit 1 = **OVFIE1** *Overflow Interrupt Enable.*
This bit is read/write by software and cleared by hardware after a reset.
0: Overflow interrupt disabled.
1: Overflow interrupt enabled.

Bit 0 = **CMPIE** *Compare Interrupt Enable*.
This bit is read/write by software and cleared by hardware after a reset. It can be used to mask the interrupt generated when any of the CMPFx bit is set.
0: Output compare interrupt disabled.
1: Output Compare interrupt enabled.

COUNTER REGISTER 1 HIGH (CNTR1H)
Read only
Reset Value: 0000 0000 (000h)

15							8
0	0	0	0	CNTR1_11	CNTR1_10	CNTR1_9	CNTR1_8

COUNTER REGISTER 1 LOW (CNTR1L)
Read only
Reset Value: 0000 0000 (000h)

7							0
CNTR1_7	CNTR1_6	CNTR1_5	CNTR1_4	CNTR1_3	CNTR1_2	CNTR1_1	CNTR1_0

Bits 15:12 = Reserved.

Bits 11:0 = **CNTR1[11:0]** *Counter Value*.
This 12-bit register is read by software and cleared by hardware after a reset. The counter CNTR1 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 CNTR1H 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 CNTR1L and CNTR1H are read. If CNTR1L is close to its highest value, CNTR1H could be incremented before it is read.
When a counter overflow occurs, the counter restarts from the value specified in the ATR1 register.

DUAL 12-BIT AUTORELOAD TIMER 3 (Cont'd)

AUTORELOAD REGISTER (ATR1H)
Read / Write
Reset Value: 0000 0000 (00h)

15							8
0	0	0	0	ATR11	ATR10	ATR9	ATR8

AUTORELOAD REGISTER (ATR1L)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
ATR7	ATR6	ATR5	ATR4	ATR3	ATR2	ATR1	ATR0

Bits 11:0 = **ATR1[11:0]** *Autoreload Register 1.*
This is a 12-bit register which is written by software. The ATR1 register value is automatically loaded into the upcounter CNTR1 when an overflow occurs. The register value is used to set the PWM frequency.

PWM OUTPUT CONTROL REGISTER (PWMCR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	OE3	0	OE2	0	OE1	0	OE0

Bits 7:0 = **OE[3:0]** *PWMx output enable.*
These bits are set and cleared by software and cleared by hardware after a reset.
0: PWM mode disabled. PWMx Output Alternate Function disabled (I/O pin free for general purpose I/O)
1: PWM mode enabled

PWMx CONTROL STATUS REGISTER (PWMxCSR)
Read / Write
Reset Value: 0000 0000 (00h)

7	6						0
0	0	0	0	0	0	OPx	CMPFx

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

Bit 1 = **OPx** *PWMx Output Polarity.*
This bit is read/write by software and cleared by hardware after a reset. This bit selects the polarity of the PWM signal.
0: The PWM signal is not inverted.
1: The PWM signal is inverted.

Bit 0 = **CMPFx** PWMx *Compare Flag.*
This bit is set by hardware and cleared by software by reading the PWMxCSR register. It indicates that the upcounter value matches the Active DCRx register value.
0: Upcounter value does not match DCRx value.
1: Upcounter value matches DCRx value.

BREAK CONTROL REGISTER (BREAKCR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	BA	BPEN	PWM3	PWM2	PWM1	PWM0

Bits 7:6 = Reserved. Forced by hardware to 0.

Bit 5 = **BA** *Break Active.*
This bit is read/write by software, cleared by hardware after reset and set by hardware when the BREAK pin is low. It activates/deactivates the Break function.
0: Break not active
1: Break active

11.3 LITE TIMER 2 (LT2)

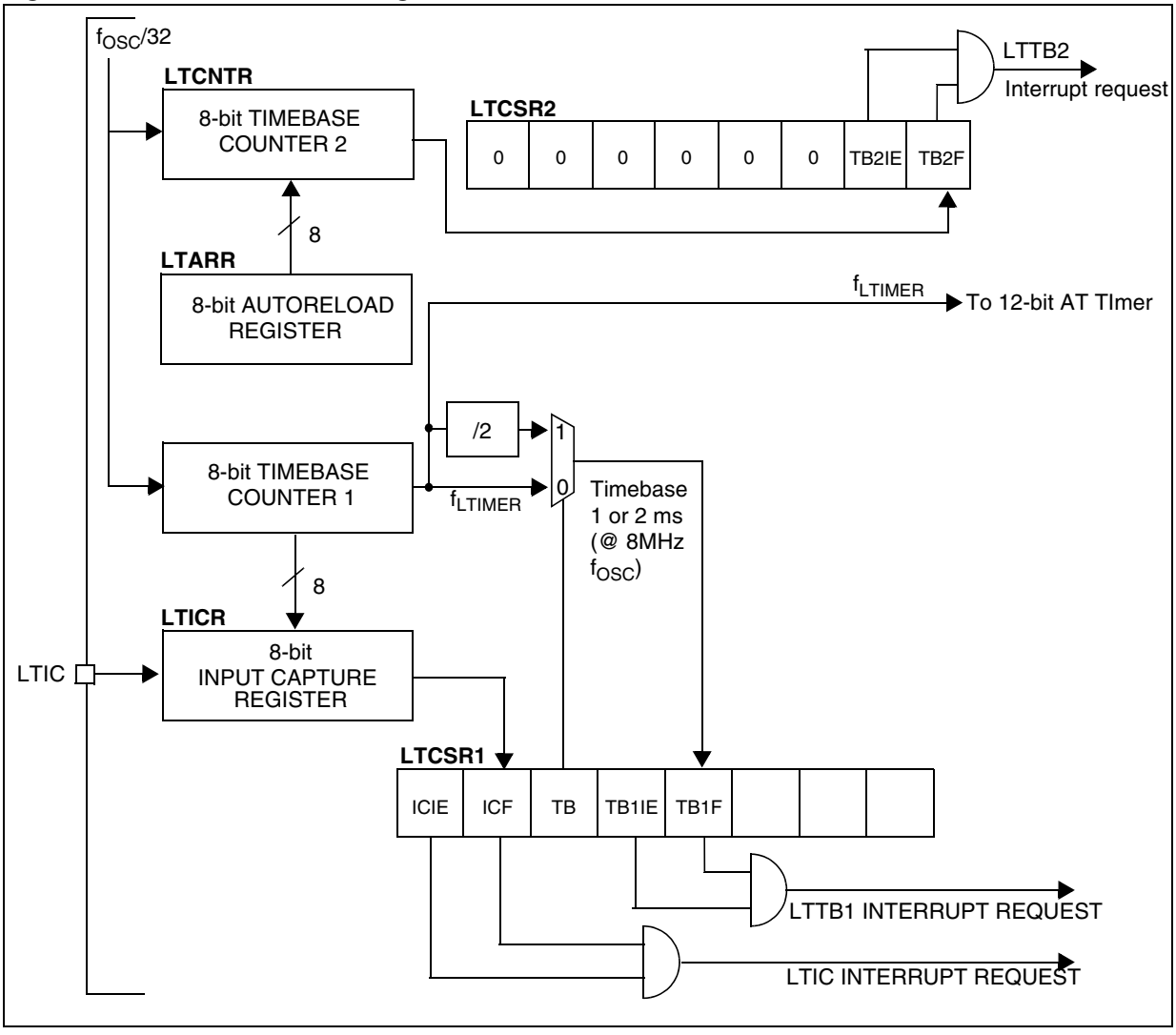
11.3.1 Introduction

The Lite Timer can be used for general-purpose timing functions. It is based on two free-running 8-bit upcounters and an 8-bit input capture register.

11.3.2 Main Features

- Realtime Clock (RTC)
  - One 8-bit upcounter 1 ms or 2 ms timebase period (@ 8 MHz f_{OSC})
  - One 8-bit upcounter with autoreload and programmable timebase period from 4µs to 1.024ms in 4µs increments (@ 8 MHz f_{OSC})
  - 2 Maskable timebase interrupts
- Input Capture
  - 8-bit input capture register (LTICR)
  - Maskable interrupt with wakeup from Halt Mode capability

Figure 46. Lite Timer 2 Block Diagram



11.4.8 Register Description

SPI CONTROL REGISTER (SPICR)

Read/Write

Reset Value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPIE** *Serial Peripheral Interrupt Enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SPI interrupt is generated whenever an End of Transfer event, Master Mode Fault or Overrun error occurs (SPIF = 1, MODF = 1 or OVR = 1 in the SPICSR register)

Bit 6 = **SPE** *Serial Peripheral Output Enable*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see Section 11.4.5.1 Master Mode Fault (MODF)). The SPE bit is cleared by reset, so the SPI peripheral is not initially connected to the external pins.
0: I/O pins free for general purpose I/O
1: SPI I/O pin alternate functions enabled

Bit 5 = **SPR2** *Divider Enable*
This bit is set and cleared by software and is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to Table 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 11.4.5.1 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 functions of the MISO and MOSI pins are reversed.

Bit 3 = **CPOL** *Clock Polarity*
This bit is set and cleared by software. This bit determines the idle state of the serial Clock. The CPOL bit affects both the master and slave modes.
0: SCK pin has a low level idle state
1: SCK pin has a high level idle state

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

Bit 2 = **CPHA** *Clock Phase*
This bit is set and cleared by software.
0: The first clock transition is the first data capture edge.
1: The second clock transition is the first capture edge.

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

Bits 1:0 = **SPR[1:0]** *Serial Clock Frequency*
These bits are set and cleared by software. Used with the SPR2 bit, they select the baud rate of the SPI serial clock SCK output by the SPI in master mode.

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

Table 18. SPI Master Mode SCK Frequency

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

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

11.5.9.9 Error due to LIN Synch measurement

The LIN Synch Field is measured over eight bit times.

This measurement is performed using a counter clocked by the CPU clock. The edge detections are performed using the CPU clock cycle.

This leads to a precision of 2 CPU clock cycles for the measurement which lasts 16*8*LDIV clock cycles.

Consequently, this error (D_{MEAS}) is equal to:

$2 / (128*LDIV_{MIN})$.

$LDIV_{MIN}$ corresponds to the minimum LIN prescaler content, leading to the maximum baud rate, taking into account the maximum deviation of +/-15%.

11.5.9.10 Error due to Baud Rate Quantization

The baud rate can be adjusted in steps of 1 / (16 * LDIV). The worst case occurs when the "real" baud rate is in the middle of the step.

This leads to a quantization error (D_{QUANT}) equal to $1 / (2*16*LDIV_{MIN})$.

11.5.9.11 Impact of Clock Deviation on Maximum Baud Rate

The choice of the nominal baud rate ($LDIV_{NOM}$) will influence both the quantization error (D_{QUANT}) and the measurement error (D_{MEAS}). The worst case occurs for $LDIV_{MIN}$.

Consequently, at a given CPU frequency, the maximum possible nominal baud rate (LPR_{MIN}) should be chosen with respect to the maximum tolerated deviation given by the equation:

$D_{TRA} + 2 / (128*LDIV_{MIN}) + 1 / (2*16*LDIV_{MIN}) + D_{REC} + D_{TCL} < 3.75\%$

Example:

A nominal baud rate of 20Kbits/s at T_{CPU} = 125ns (8 MHz) leads to $LDIV_{NOM}$ = 25d.

$LDIV_{MIN} = 25 - 0.15*25 = 21.25$

$D_{MEAS} = 2 / (128*LDIV_{MIN}) * 100 = 0.00073\%$

$D_{QUANT} = 1 / (2*16*LDIV_{MIN}) * 100 = 0.0015\%$

LIN Slave systems

For LIN Slave systems (the LINE and LSLV bits are set), receivers wake up by LIN Synch Break or LIN Identifier detection (depending on the LHDM bit).

Hot Plugging Feature for LIN Slave Nodes

In LIN Slave Mute Mode (the LINE, LSLV and RWU bits are set) it is possible to hot plug to a network during an ongoing communication flow. In this case the SCI monitors the bus on the RDI line until 11 consecutive dominant bits have been detected and discards all the other bits received.

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

CONTROL REGISTER 2 (SCICR2)

Read/Write
Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

Bits 7:2 Same function as in SCI mode; please refer to Section 11.5.8 SCI Mode Register Description.

Bit 1 = **RWU** *Receiver wake-up.*
This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wake-up sequence is recognized.
0: Receiver in active mode
1: Receiver in mute mode

Notes:

– Mute mode is recommended for detecting only the Header and avoiding the reception of any other characters. For more details, please refer to Section 11.5.9.3 LIN Reception.

– In LIN slave mode, when RDRF is set, the software can not set or clear the RWU bit.

Bit 0 = **SBK** *Send break.*
This bit set is used to send break characters. It is set and cleared by software.
0: No break character is transmitted
1: Break characters are transmitted

Note: If the SBK bit is set to "1" and then to "0", the transmitter will send a BREAK word at the end of the current word.

CONTROL REGISTER 3 (SCICR3)

Read/Write
Reset Value: 0000 0000 (00h)

7							0
LDUM	LINE	LSLV	LASE	LHDM	LHIE	LHDF	LSF

Bit 7 = **LDUM** *LIN Divider Update Method.*
This bit is set and cleared by software and is also cleared by hardware (when RDRF = 1). It is only used in LIN Slave mode. It determines how the LIN Divider can be updated by software.
0: LDIV is updated as soon as LPR is written (if no Auto Synchronization update occurs at the same time).
1: LDIV is updated at the next received character (when RDRF = 1) after a write to the LPR register

Notes:

- If no write to LPR is performed between the setting of LDUM bit and the reception of the next character, LDIV will be updated with the old value.

- After LDUM has been set, it is possible to reset the LDUM bit by software. In this case, LDIV can be modified by writing into LPR / LPFR registers.

Bits 6:5 = **LINE, LSLV** *LIN Mode Enable Bits.*
These bits configure the LIN mode:

LINE	LSLV	Meaning
0	x	LIN mode disabled
1	0	LIN Master Mode
	1	LIN Slave Mode

The LIN Master configuration enables:

The capability to send LIN Synch Breaks (13 low bits) using the SBK bit in the SCICR2 register.

The LIN Slave configuration enables:

– The LIN Slave Baud Rate generator. The LIN Divider (LDIV) is then represented by the LPR and LPFR registers. The LPR and LPFR registers are read/write accessible at the address of the SCIBRR register and the address of the SCIETPR register

– Management of LIN Headers.

– LIN Synch Break detection (11-bit dominant).

– LIN Wake-Up method (see LHDM bit) instead of the normal SCI Wake-Up method.

– Inhibition of Break transmission capability (SBK has no effect)

– LIN Parity Checking (in conjunction with the PCE bit)

Bit 4 = **LASE** *LIN Auto Synch Enable.*
This bit enables the Auto Synch Unit (ASU). It is set and cleared by software. It is only usable in LIN Slave mode.

0: Auto Synch Unit disabled
1: Auto Synch Unit enabled.

Bit 3 = **LHDM** *LIN Header Detection Method*
This bit is set and cleared by software. It is only usable in LIN Slave mode. It enables the Header Detection Method. In addition if the RWU bit in the

13 ELECTRICAL CHARACTERISTICS

13.1 PARAMETER CONDITIONS

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

13.1.1 Minimum and Maximum values

Unless otherwise specified the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage and frequencies by tests in production on 100% of the devices with an ambient temperature at T_A=25°C and T_A=$T_A$max (given by the selected temperature range).

Data based on characterization results, design simulation and/or technology characteristics are indicated in the table footnotes and are not tested in production. Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value plus or minus three times the standard deviation (mean±3Σ).

13.1.2 Typical values

Unless otherwise specified, typical data are based on T_A=25°C, V_{DD}=5V (for the 4.5V≤V_{DD}≤5.5V voltage range) and V_{DD}=3.3V (for the 3V≤V_{DD}≤4V voltage range). They are given only as design guidelines and are not tested.

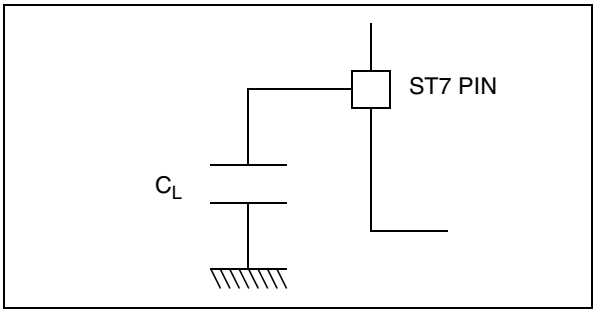
13.1.3 Typical curves

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

13.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in Figure 67.

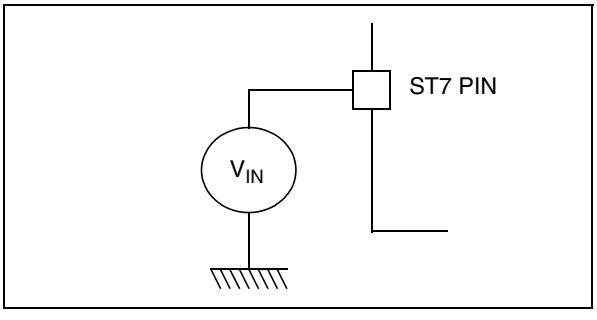
Figure 67. Pin loading conditions



13.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in Figure 68.

Figure 68. Pin input voltage



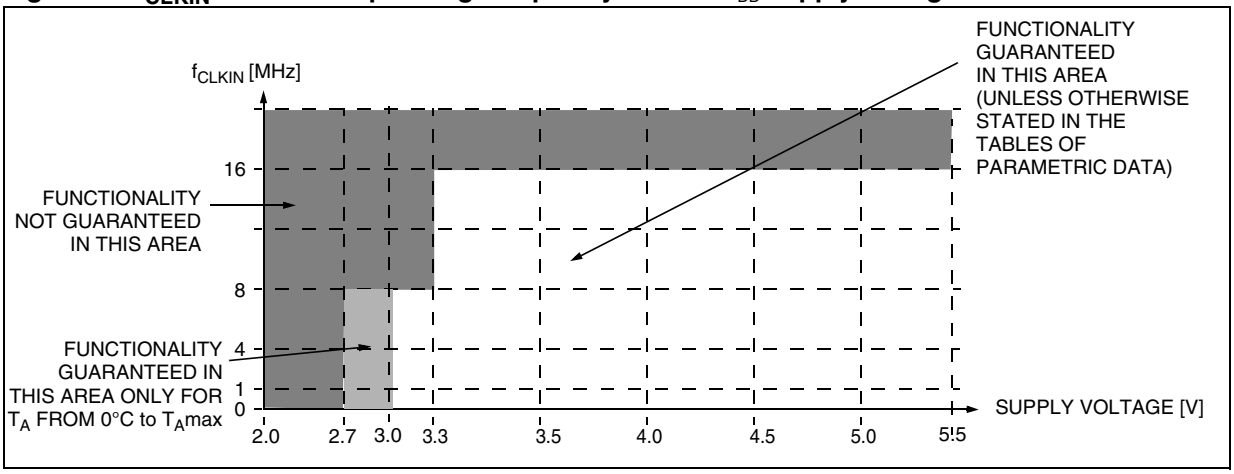
13.3 OPERATING CONDITIONS

13.3.1 General Operating Conditions: Suffix 6 Devices

T_A = -40 to +125°C unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	Supply voltage	f_{OSC} = 8 MHz. max., T_A = 0 to 125°C	2.7	5.5	V
		f_{OSC} = 8 MHz. max., T_A = - 40 to 125°C	3.0	5.5	
		f_{OSC} = 16 MHz. max.	3.3	5.5	
f_{CLKIN}	External clock frequency on CLKIN pin	$V_{DD} \geq 3.3V$	up to 16		MHz
		$V_{DD} \geq 3.0V$	up to 8		

Figure 69. f_{CLKIN} Maximum Operating Frequency Versus V_{DD} Supply Voltage



OPERATING CONDITIONS (Cont'd)

The RC oscillator and PLL characteristics are temperature-dependent and are grouped in two tables.

13.3.1.1 Devices tested for T_A = -40 to +125°C @ V_{DD} = 4.5 to 5.5V

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC} 1)	Internal RC oscillator frequency	RCCR = FF (reset value), T_A=25°C,V_{DD}=5V		630		kHz
		RCCR = RCCR0 2),T_A=25°C,V_{DD}=5V	995	1000	1005	
ACC_{RC}	Accuracy of Internal RC oscillator with RCCR=RCCR0 2)	T_A=25°C, V_{DD}=4.5 to 5.5V	-1		+1	%
		T_A=-40 to +85°C, V_{DD}=4.5 to 5.5V	-2 3)		+5 3)	%
		T_A=-40 to +125°C, V_{DD}=4.5 to 5.5V	-3		+5	%
$I_{DD(RC)}$	RC oscillator current consumption	T_A=25°C,V_{DD}=5V		600 4)5)		µA
$t_{su(RC)}$	RC oscillator setup time	T_A=25°C,V_{DD}=5V			10 2)	µs
f_{PLL}	x8 PLL input clock			1		MHz
t_{LOCK}	PLL Lock time 8)			2		ms
t_{STAB}	PLL Stabilization time 8)			4		ms
ACC_{PLL}	x8 PLL Accuracy	f_{RC} = 1MHz@T_A=25°C,V_{DD}=4.5 to 5.5V		0.1 7)		%
		f_{RC} = 1MHz@T_A=-40 to +125°C,V_{DD}=5V		0.1 7)		%
$t_{w(JIT)}$	PLL jitter period	f_{RC} = 1MHz		8 6)		kHz
JIT_{PLL}	PLL jitter ($\Delta f_{CPU}/f_{CPU}$)			1 6)		%
$I_{DD(PLL)}$	PLL current consumption	T_A=25°C		550 4)		µA

Notes:

1. If the RC oscillator clock is selected, to improve clock stability and frequency accuracy, it is recommended to place a decoupling capacitor, typically 100nF, between the V_{DD} and V_{SS} pins as close as possible to the ST7 device.

2. See "INTERNAL RC OSCILLATOR ADJUSTMENT" on page 23

3. Min value is obtained for hot temperature and max value is obtained for cold temperature.

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

5. Measurement made with RC calibrated at 1MHz.

6. Guaranteed by design.

7. Averaged over a 4ms period. After the LOCKED bit is set, a period of t_{STAB} is required to reach ACC_{PLL} accuracy.

8. After the LOCKED bit is set ACC_{PLL} is max. 10% until t_{STAB} has elapsed. See Figure 12 on page 24.

Figure 85. Typical I_{PU} vs. V_{DD} with $V_{IN}=V_{SS}$