

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	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 ~ 125°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/st7flite39f2m3

DATA EEPROM (Cont'd)

5.3 MEMORY ACCESS

The Data EEPROM memory read/write access modes are controlled by the E2LAT bit of the EEPROM Control/Status register (EECSR). The flowchart in [Figure 7](#) describes these different memory access modes.

Read Operation (E2LAT=0)

The EEPROM can be read as a normal ROM location when the E2LAT bit of the EECSR register is cleared.

On this device, Data EEPROM can also be used to execute machine code. Take care not to write to the Data EEPROM while executing from it. This would result in an unexpected code being executed.

Write Operation (E2LAT=1)

To access the write mode, the E2LAT bit has to be set by software (the E2PGM bit remains cleared). When a write access to the EEPROM area occurs,

the value is latched inside the 32 data latches according to its address.

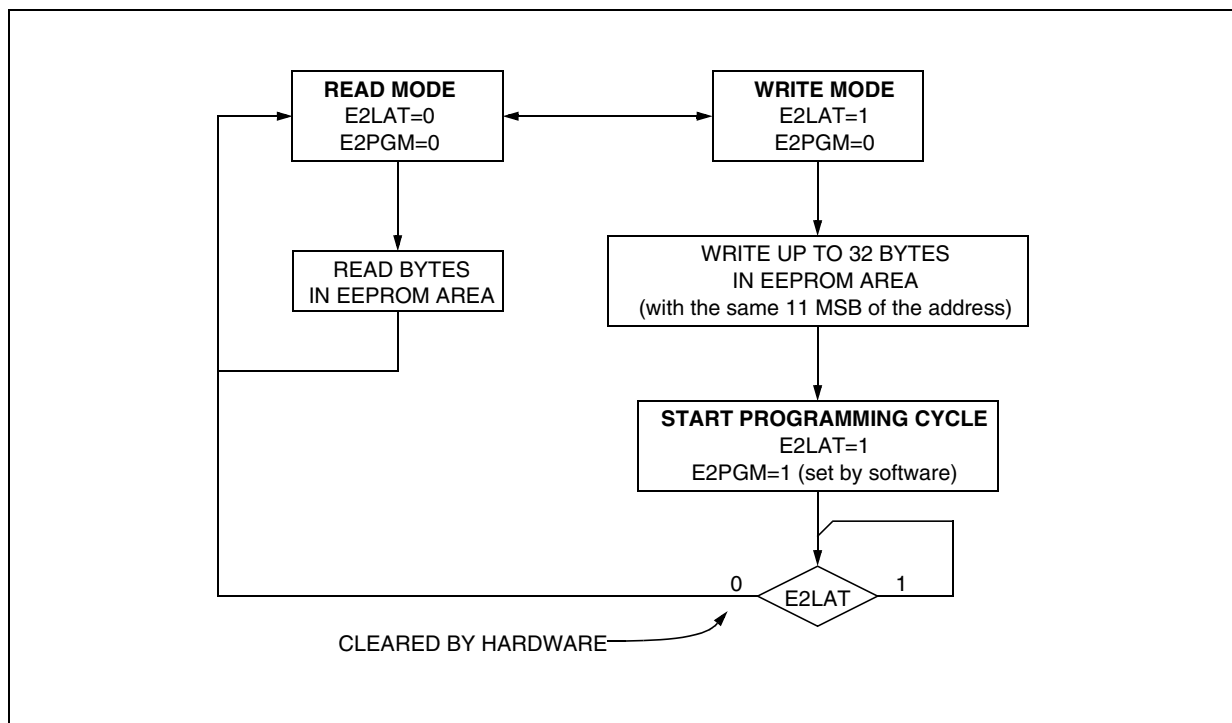
When PGM bit is set by the software, all the previous bytes written in the data latches (up to 32) are programmed in the EEPROM cells. The effective high address (row) is determined by the last EEPROM write sequence. To avoid wrong programming, the user must take care that all the bytes written between two programming sequences have the same high address: only the five Least Significant Bits of the address can change.

At the end of the programming cycle, the PGM and LAT bits are cleared simultaneously.

Note: Care should be taken during the programming cycle. Writing to the same memory location will over-program the memory (logical AND between the two write access data result) because the data latches are only cleared at the end of the programming cycle and by the falling edge of the E2LAT bit.

It is not possible to read the latched data. This note is illustrated by the [Figure 9](#).

Figure 7. Data EEPROM Programming Flowchart



6 CENTRAL PROCESSING UNIT

6.1 INTRODUCTION

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

6.2 MAIN FEATURES

- 63 basic instructions
- Fast 8-bit by 8-bit multiply
- 17 main addressing modes
- Two 8-bit index registers
- 16-bit stack pointer
- Low power modes
- Maskable hardware interrupts
- Non-maskable software interrupt

6.3 CPU REGISTERS

The six CPU registers shown in [Figure 10](#) are not present in the memory mapping and are accessed by specific instructions.

Accumulator (A)

The Accumulator is an 8-bit general purpose register used to hold operands and the results of the arithmetic and logic calculations and to manipulate data.

Index Registers (X and Y)

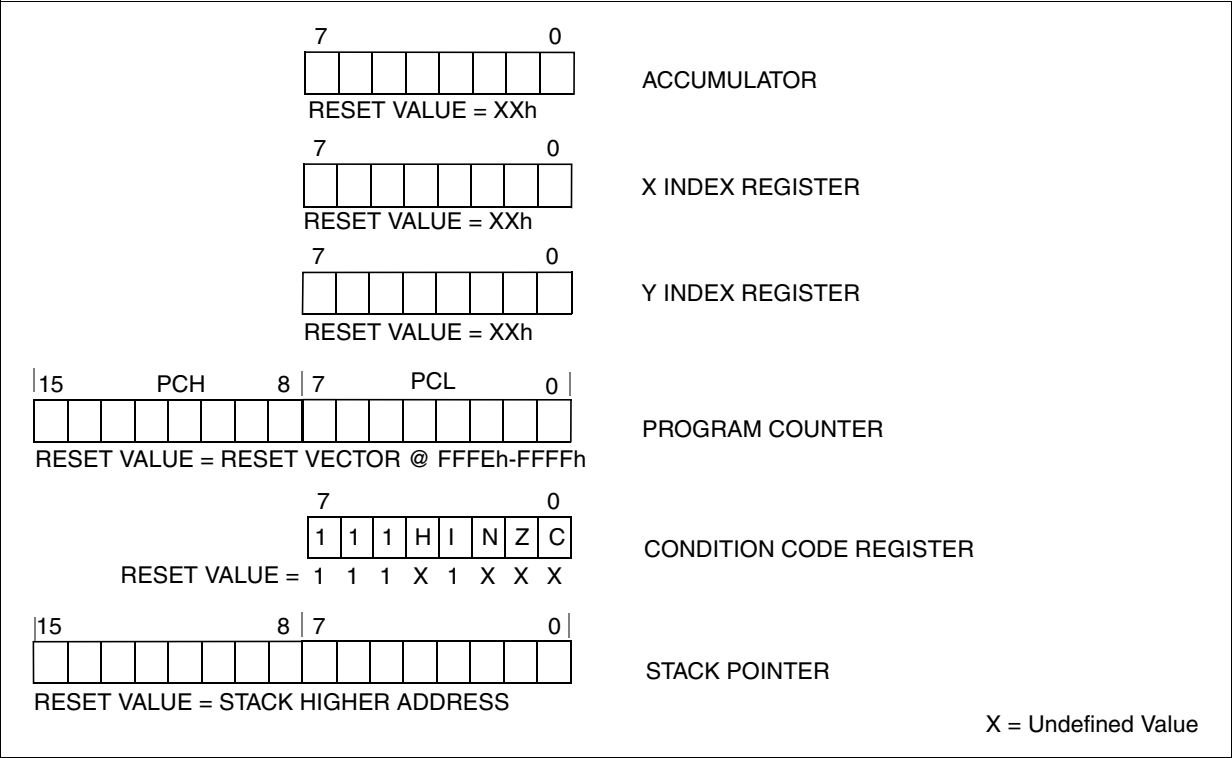
In indexed addressing modes, these 8-bit registers are used to create either effective addresses or temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

The Y register is not affected by the interrupt automatic procedures (not pushed to and popped from the stack).

Program Counter (PC)

The program counter is a 16-bit register containing the address of the next instruction to be executed by the CPU. It is made of two 8-bit registers PCL (Program Counter Low which is the LSB) and PCH (Program Counter High which is the MSB).

Figure 10. CPU Registers



7.4 MULTI-OSCILLATOR (MO)

The main clock of the ST7 can be generated by four different source types coming from the multi-oscillator block (1 to 16MHz or 32kHz):

- an external source
- 5 crystal or ceramic resonator oscillators
- an internal high frequency RC oscillator

Each oscillator is optimized for a given frequency range in terms of consumption and is selectable through the option byte. The associated hardware configurations are shown in [Table 5](#). Refer to the electrical characteristics section for more details.

External Clock Source

In this external clock mode, a clock signal (square, sinus or triangle) with ~50% duty cycle has to drive the OSC1 pin while the OSC2 pin is tied to ground.

Note: when the Multi-Oscillator is not used, PB4 is selected by default as external clock.

Crystal/Ceramic Oscillators

This family of oscillators has the advantage of producing a very accurate rate on the main clock of the ST7. The selection within a list of 4 oscillators with different frequency ranges has to be done by option byte in order to reduce consumption (refer to [section 15.1 on page 161](#) for more details on the frequency ranges). In this mode of the multi-oscillator, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. The loading capacitance values must be adjusted according to the selected oscillator.

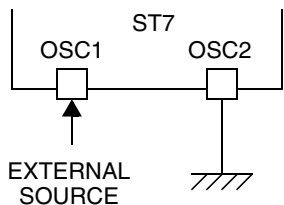
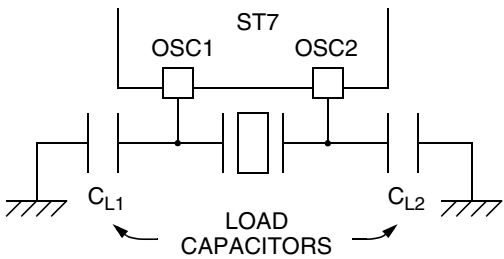
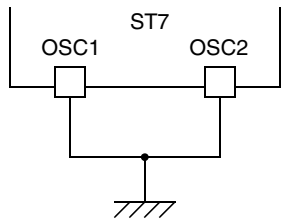
These oscillators are not stopped during the RESET phase to avoid losing time in the oscillator start-up phase.

Internal RC Oscillator

In this mode, the tunable 1%RC oscillator is used as main clock source. The two oscillator pins have to be tied to ground.

The calibration is done through the RCCR[7:0] and SICSRR[6:5] registers.

Table 5. ST7 Clock Sources

	Hardware Configuration
External Clock	
Crystal/Ceramic Resonators	
Internal RC Oscillator	

INTERRUPTS (Cont'd)

Bit 3:2 = **ei1[1:0]** *ei1 pin selection*

These bits are written by software. They select the Port A I/O pin used for the ei1 external interrupt according to the table below.

External Interrupt I/O pin selection

ei11	ei10	I/O Pin
0	0	No interrupt*
0	1	PA4
1	0	PA5
1	1	PA6

* Reset State

Bit 1:0 = **ei0[1:0]** *ei0 pin selection*

These bits are written by software. They select the

Port A I/O pin used for the ei0 external interrupt according to the table below.

External Interrupt I/O pin selection

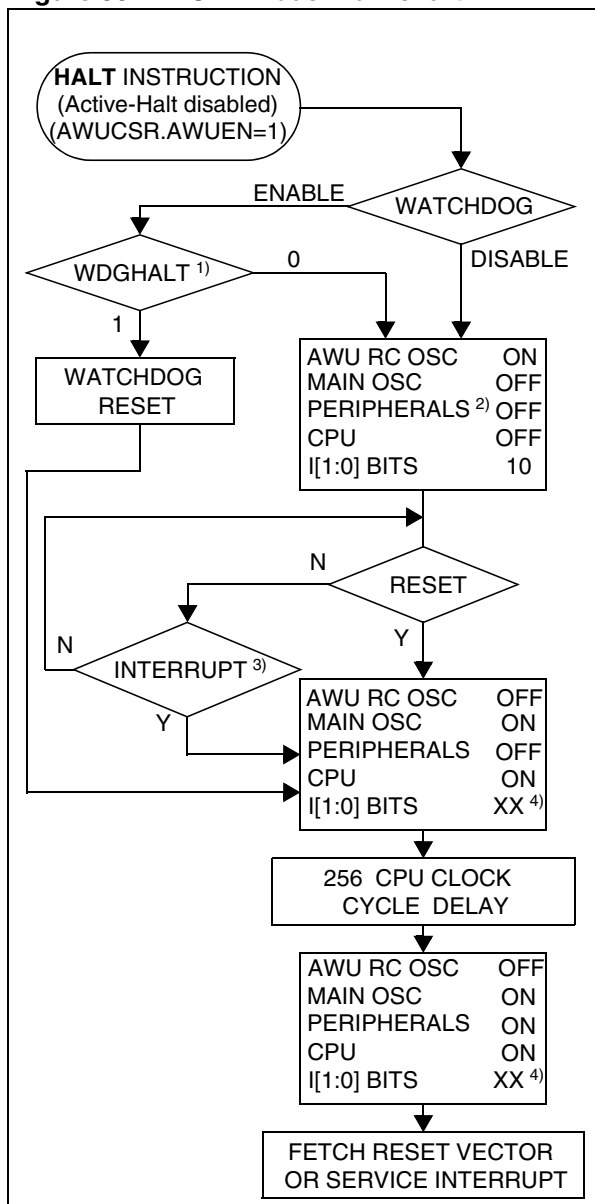
ei01	ei00	I/O Pin
0	0	No Interrupt*
0	1	PA1
1	0	PA2
1	1	PA3

* Reset State

Bits 1:0 = Reserved.

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.

10 I/O PORTS

10.1 INTRODUCTION

The I/O ports allow data transfer. An I/O port can contain up to 8 pins. Each pin can be programmed independently either as a digital input or digital output. In addition, specific pins may have several other functions. These functions can include external interrupt, alternate signal input/output for on-chip peripherals or analog input.

10.2 FUNCTIONAL DESCRIPTION

A Data Register (DR) and a Data Direction Register (DDR) are always associated with each port. The Option Register (OR), which allows input/output options, may or may not be implemented. The following description takes into account the OR register. Refer to the Port Configuration table for device specific information.

An I/O pin is programmed using the corresponding bits in the DDR, DR and OR registers: bit x corresponding to pin x of the port.

Figure 31 shows the generic I/O block diagram.

10.2.1 Input Modes

Clearing the DDRx bit selects input mode. In this mode, reading its DR bit returns the digital value from that I/O pin.

If an OR bit is available, different input modes can be configured by software: floating or pull-up. Refer to I/O Port Implementation section for configuration.

Notes:

1. Writing to the DR modifies the latch value but does not change the state of the input pin.
2. Do not use read/modify/write instructions (BSET/BRES) to modify the DR register.

External Interrupt Function

Depending on the device, setting the ORx bit while in input mode can configure an I/O as an input with interrupt. In this configuration, a signal edge or level input on the I/O generates an interrupt request via the corresponding interrupt vector (eix).

Falling or rising edge sensitivity is programmed independently for each interrupt vector. The External Interrupt Control Register (EICR) or the Miscellaneous Register controls this sensitivity, depending on the device.

A device may have up to 7 external interrupts. Several pins may be tied to one external interrupt vector. Refer to Pin Description to see which ports have external interrupts.

If several I/O interrupt pins on the same interrupt vector are selected simultaneously, they are logically combined. For this reason if one of the interrupt pins is tied low, it may mask the others.

External interrupts are hardware interrupts. Fetching the corresponding interrupt vector automatically clears the request latch. Modifying the sensitivity bits will clear any pending interrupts.

10.2.2 Output Modes

Setting the DDRx bit selects output mode. Writing to the DR bits applies a digital value to the I/O through the latch. Reading the DR bits returns the previously stored value.

If an OR bit is available, different output modes can be selected by software: push-pull or open-drain. Refer to I/O Port Implementation section for configuration.

DR Value and Output Pin Status

DR	Push-Pull	Open-Drain
0	V_{OL}	V_{OL}
1	V_{OH}	Floating

10.2.3 Alternate Functions

Many ST7s I/Os have one or more alternate functions. These may include output signals from, or input signals to, on-chip peripherals. The Device Pin Description table describes which peripheral signals can be input/output to which ports.

A signal coming from an on-chip peripheral can be output on an I/O. To do this, enable the on-chip peripheral as an output (enable bit in the peripheral's control register). The peripheral configures the I/O as an output and takes priority over standard I/O programming. The I/O's state is readable by addressing the corresponding I/O data register.

Configuring an I/O as floating enables alternate function input. It is not recommended to configure an I/O as pull-up as this will increase current consumption. Before using an I/O as an alternate input, configure it without interrupt. Otherwise spurious interrupts can occur.

Configure an I/O as input floating for an on-chip peripheral signal which can be input and output.

Caution:

I/Os which can be configured as both an analog and digital alternate function need special attention. The user must control the peripherals so that the signals do not arrive at the same time on the same pin. If an external clock is used, only the clock alternate function should be employed on that I/O pin and not the other alternate function.

DUAL 12-BIT AUTORELOAD TIMER 3 (Cont'd)**■ Long input capture**

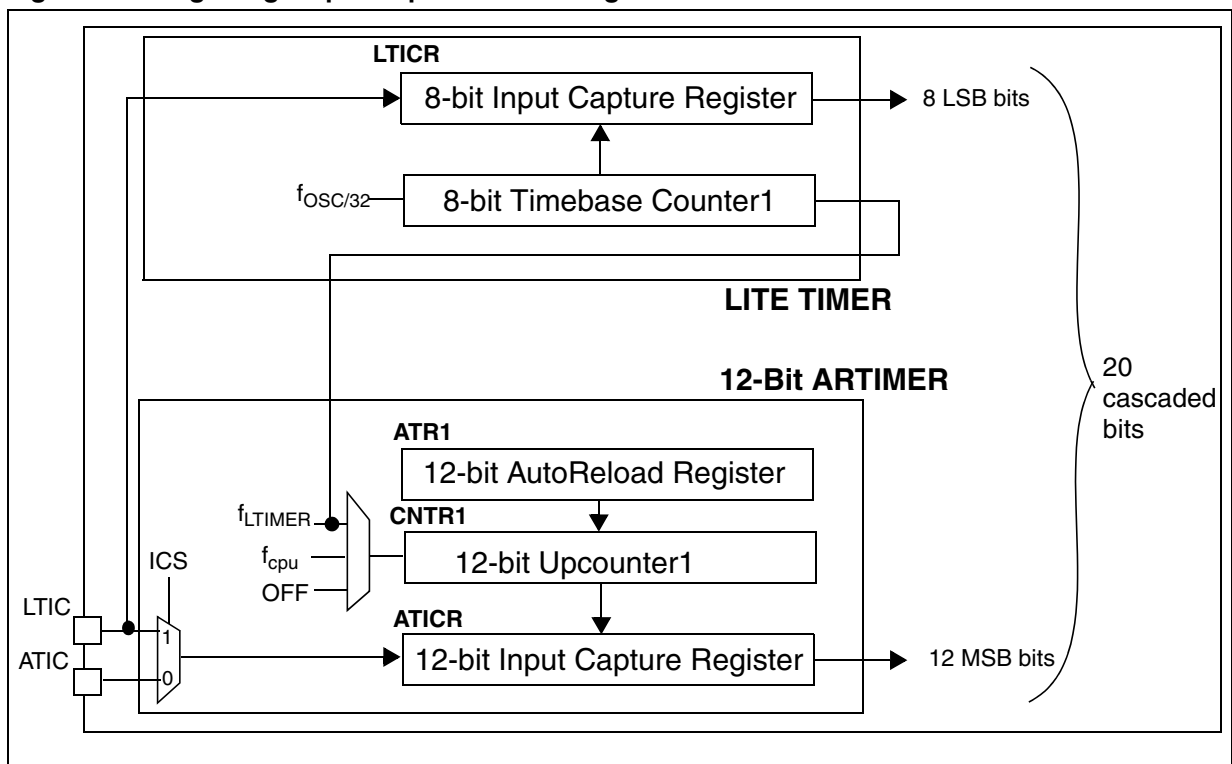
Pulses that last between 8µs and 2s can be measured with an accuracy of 4µs if $f_{OSC} = 8\text{MHz}$ in the following conditions:

- 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.

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.

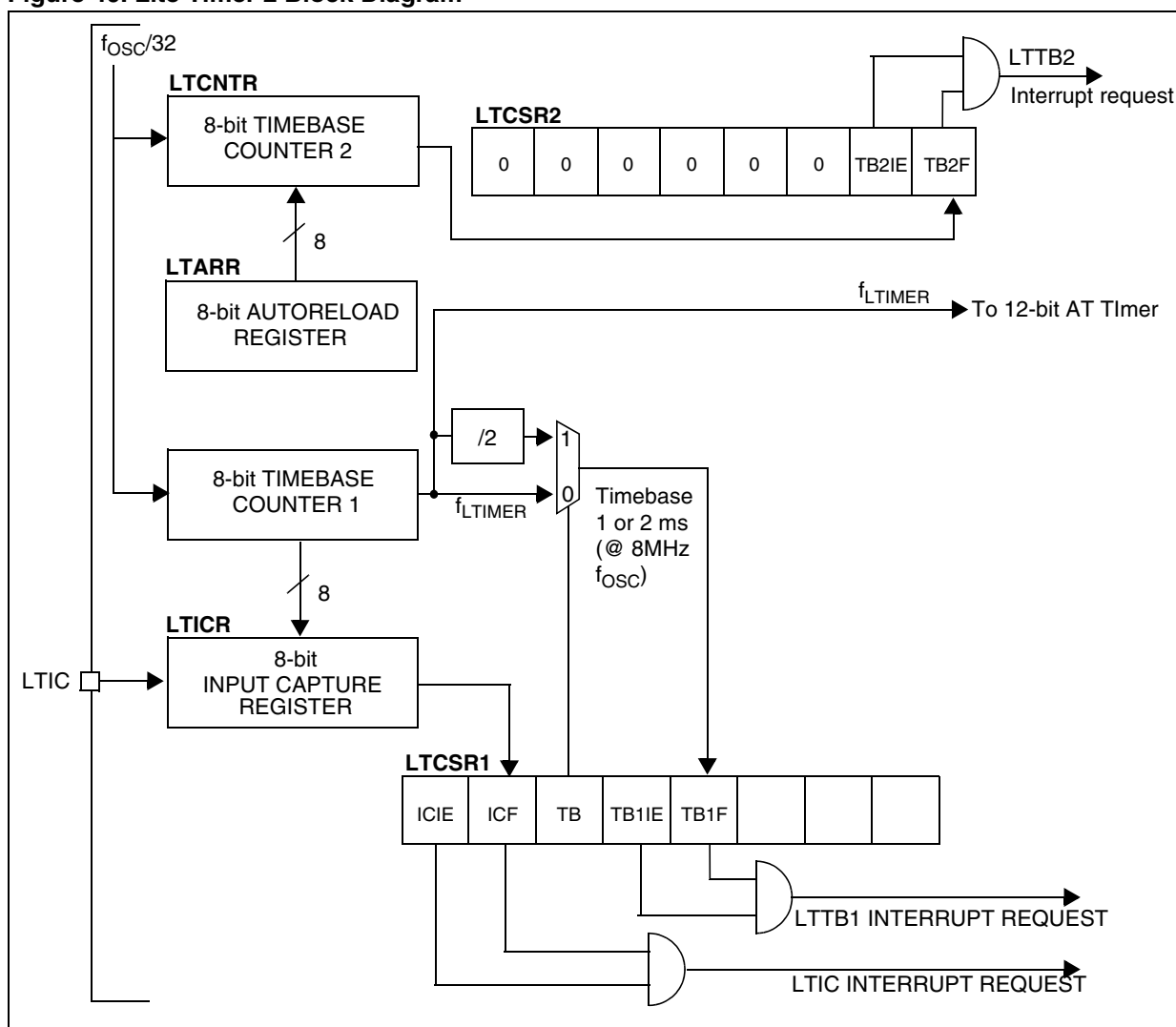
11.3.2 Main Features

■ 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 4 μ s to 1.024ms in 4 μ s 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



SERIAL PERIPHERAL INTERFACE (cont'd)

11.4.3.3 Master Mode Operation

In master mode, the serial clock is output on the SCK pin. The clock frequency, polarity and phase are configured by software (refer to the description of the SPICSR register).

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

How to operate the SPI in master mode

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

1. Write to the SPICR register:
 - Select the clock frequency by configuring the SPR[2:0] bits.
 - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits. [Figure 52](#) shows the four possible configurations.
Note: The slave must have the same CPOL and CPHA settings as the master.
2. Write to the SPICSR register:
 - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the SS pin high for the complete byte transmit sequence.
3. Write to the SPICR register:
 - Set the MSTR and SPE bits
Note: MSTR and SPE bits remain set only if SS is high).

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

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

11.4.3.4 Master Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware.
- An interrupt request is generated if the SPIE bit is set and the interrupt mask in the CCR register is cleared.

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

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

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

11.4.3.5 Slave Mode Operation

In slave mode, the serial clock is received on the SCK pin from the master device.

To operate the SPI in slave mode:

1. Write to the SPICSR register to perform the following actions:
 - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see [Figure 52](#)).
Note: The slave must have the same CPOL and CPHA settings as the master.
 - Manage the \overline{SS} pin as described in [Section 11.4.3.2](#) and [Figure 50](#). If CPHA = 1 SS must be held low continuously. If CPHA = 0 SS must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

11.4.3.6 Slave Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

- The SPIF bit is set by hardware.
- An interrupt request is generated if SPIE bit is set and interrupt mask in the CCR register is cleared.

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

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

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

The SPIF bit can be cleared during a second transmission; however, it must be cleared before the second SPIF bit in order to prevent an Overrun condition (see [Section 11.4.5.2](#)).

SERIAL PERIPHERAL INTERFACE (cont'd)**11.4.5 Error Flags****11.4.5.1 Master Mode Fault (MODF)**

Master mode fault occurs when the master device's \overline{SS} pin is pulled low.

When a Master mode fault occurs:

- The MODF bit is set and an SPI interrupt request is generated if the SPIE bit is set.
- The SPE bit is reset. This blocks all output from the device and disables the SPI peripheral.
- The MSTR bit is reset, thus forcing the device into slave mode.

Clearing the MODF bit is done through a software sequence:

1. A read access to the SPICSR register while the MODF bit is set.
2. A write to the SPICR register.

Notes: To avoid any conflicts in an application with multiple slaves, the \overline{SS} pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits may be restored to their original state during or after this clearing sequence.

Hardware does not allow the user to set the SPE and MSTR bits while the MODF bit is set except in the MODF bit clearing sequence.

In a slave device, the MODF bit can not be set, but in a multimaster configuration the device can be in slave mode with the MODF bit set.

The MODF bit indicates that there might have been a multimaster conflict and allows software to handle this using an interrupt routine and either perform a reset or return to an application default state.

11.4.5.2 Overrun Condition (OVR)

An overrun condition occurs when the master device has sent a data byte and the slave device has not cleared the SPIF bit issued from the previously transmitted byte.

When an Overrun occurs:

- The OVR bit is set and an interrupt request is generated if the SPIE bit is set.

In this case, the receiver buffer contains the byte sent after the SPIF bit was last cleared. A read to the SPIDR register returns this byte. All other bytes are lost.

The OVR bit is cleared by reading the SPICSR register.

11.4.5.3 Write Collision Error (WCOL)

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

Write collisions can occur both in master and slave mode. See also [Section 11.4.3.2 Slave Select Management](#).

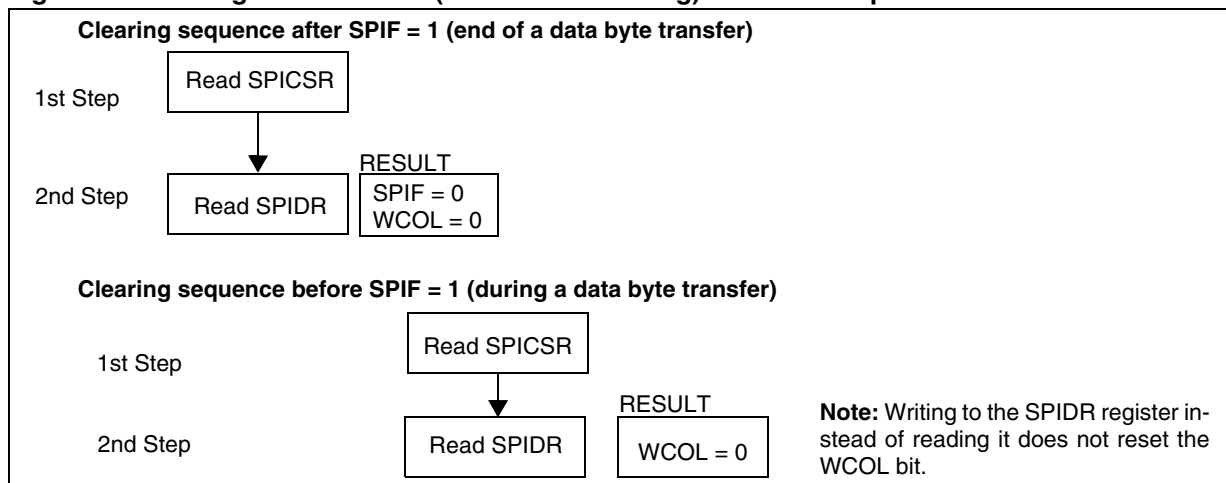
Note: A "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

The WCOL bit in the SPICSR register is set if a write collision occurs.

No SPI interrupt is generated when the WCOL bit is set (the WCOL bit is a status flag only).

Clearing the WCOL bit is done through a software sequence (see [Figure 53](#)).

Figure 53. Clearing the WCOL Bit (Write Collision Flag) Software Sequence



SERIAL PERIPHERAL INTERFACE (cont'd)**SPI CONTROL/STATUS REGISTER (SPICSR)**

Read/Write (some bits Read Only)

Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	OVR	MODF	-	SOD	SSM	SSI

Bit 7 = SPIF Serial Peripheral Data Transfer Flag (Read only)

This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE = 1 in the SPICR register. It is cleared by a software sequence (an access to the SPICSR register followed by a write or a read to the SPIDR register).

0: Data transfer is in progress or the flag has been cleared.

1: Data transfer between the device and an external device has been completed.

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

Bit 6 = WCOL Write Collision status (Read only)

This bit is set by hardware when a write to the SPIDR register is done during a transmit sequence. It is cleared by a software sequence (see [Figure 53](#)).

0: No write collision occurred

1: A write collision has been detected

Bit 5 = OVR SPI Overrun error (Read only)

This bit is set by hardware when the byte currently being received in the shift register is ready to be transferred into the SPIDR register while SPIF = 1 (See [Section 11.4.5.2](#)). An interrupt is generated if SPIE = 1 in the SPICR register. The OVR bit is cleared by software reading the SPICSR register.

0: No overrun error

1: Overrun error detected

Bit 4 = MODF Mode Fault flag (Read only)

This bit is set by hardware when the \overline{SS} pin is pulled low in master mode (see [Section 11.4.5.1 Master Mode Fault \(MODF\)](#)). An SPI interrupt can be generated if SPIE = 1 in the SPICR register. This bit is cleared by a software sequence (An access to the SPICSR register while MODF = 1 followed by a write to the SPICR register).

0: No master mode fault detected

1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

Bit 2 = SOD SPI Output Disable

This bit is set and cleared by software. When set, it disables the alternate function of the SPI output (MOSI in master mode / MISO in slave mode)

0: SPI output enabled (if SPE = 1)

1: SPI output disabled

Bit 1 = SSM \overline{SS} Management

This bit is set and cleared by software. When set, it disables the alternate function of the SPI \overline{SS} pin and uses the SSI bit value instead. See [Section 11.4.3.2 Slave Select Management](#).

0: Hardware management (\overline{SS} managed by external pin)

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

Bit 0 = SSI \overline{SS} Internal Mode

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

0: Slave selected

1: Slave deselected

SPI DATA I/O REGISTER (SPIDR)

Read/Write

Reset Value: Undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

The SPIDR register is used to transmit and receive data on the serial bus. In a master device, a write to this register will initiate transmission/reception of another byte.

Notes: During the last clock cycle the SPIF bit is set, a copy of the received data byte in the shift register is moved to a buffer. When the user reads the serial peripheral data I/O register, the buffer is actually being read.

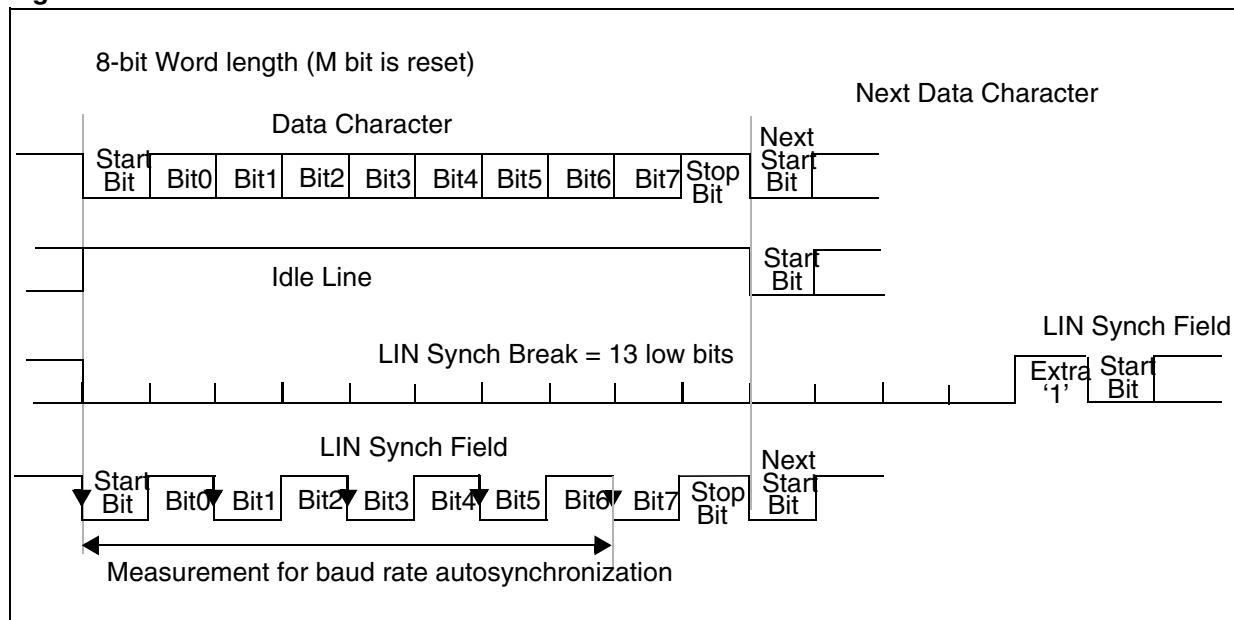
While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Warning: A write to the SPIDR register places data directly into the shift register for transmission.

A read to the SPIDR register returns the value located in the buffer and not the content of the shift register (see [Figure 48](#)).

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

Figure 58. LIN Characters



LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)**11.5.10 LIN Mode Register Description****STATUS REGISTER (SCISR)**

Read Only

Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	LHE	NF	FE	PE

Bits 7:4 = Same function as in SCI mode; please refer to [Section 11.5.8 SCI Mode Register Description](#).

Bit 3 = **LHE** LIN Header Error.

During LIN Header this bit signals three error types:

- The LIN Synch Field is corrupted and the SCI is blocked in LIN Synch State (LSF bit = 1).
- A timeout occurred during LIN Header reception
- An overrun error was detected on one of the header field (see OR bit description in [Section 11.5.8 SCI Mode Register Description](#)).

An interrupt is generated if RIE = 1 in the SCICR2 register. If blocked in the LIN Synch State, the LSF bit must first be reset (to exit LIN Synch Field state and then to be able to clear LHE flag). Then it is cleared by the following software sequence: An access to the SCISR register followed by a read to the SCIDR register.

0: No LIN Header error

1: LIN Header error detected

Note:

Apart from the LIN Header this bit signals an Over-run Error as in SCI mode (see description in [Section 11.5.8 SCI Mode Register Description](#)).

Bit 2 = **NF** Noise flag

In LIN Master mode (LINE bit = 1 and LSLV bit = 0), this bit has the same function as in SCI mode; please refer to [Section 11.5.8 SCI Mode Register Description](#).

In LIN Slave mode (LINE bit = 1 and LSLV bit = 1) this bit has no meaning.

Bit 1 = **FE** Framing error.

In LIN slave mode, this bit is set only when a real

framing error is detected (if the stop bit is dominant (0) and at least one of the other bits is recessive (1). It is not set when a break occurs, the LHDF bit is used instead as a break flag (if the LHDM bit = 0). It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Framing error

1: Framing error detected

Bit 0 = **PE** Parity error.

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

0: No LIN parity error

1: LIN Parity error detected

CONTROL REGISTER 1 (SCICR1)

Read/Write

Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE	PS	PIE

Bits 7:3 = Same function as in SCI mode; please refer to [Section 11.5.8 SCI Mode Register Description](#).

Bit 2 = **PCE** Parity control enable.

This bit is set and cleared by software. It selects the hardware parity control for LIN identifier parity check.

0: Parity control disabled

1: Parity control enabled

When a parity error occurs, the PE bit in the SCISR register is set.

Bit 1 = Reserved

Bit 0 = Same function as in SCI mode; please refer to [Section 11.5.8 SCI Mode Register Description](#).

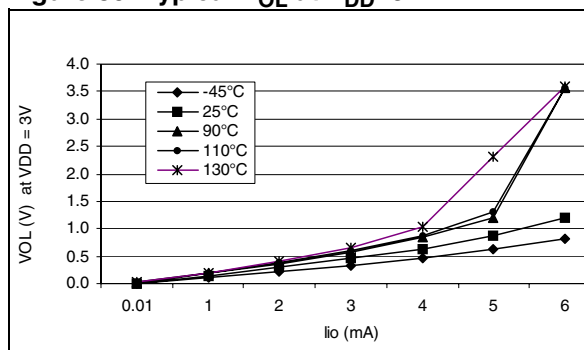
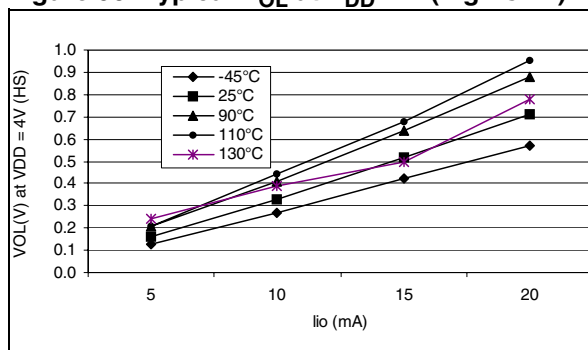
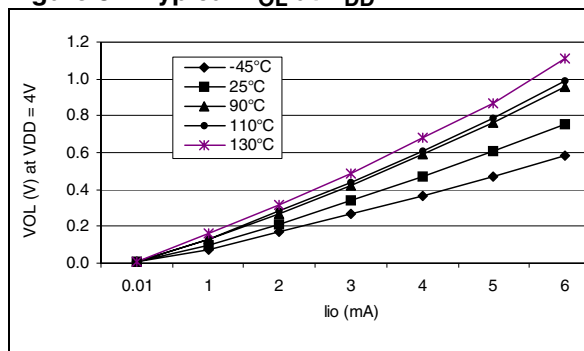
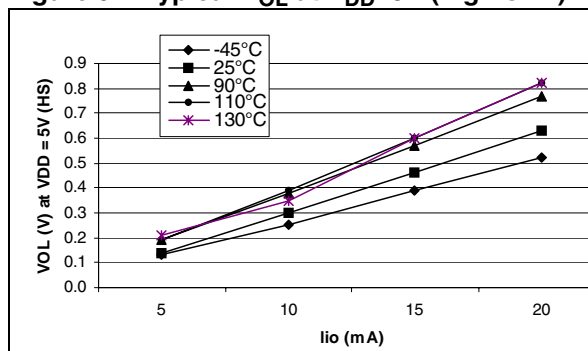
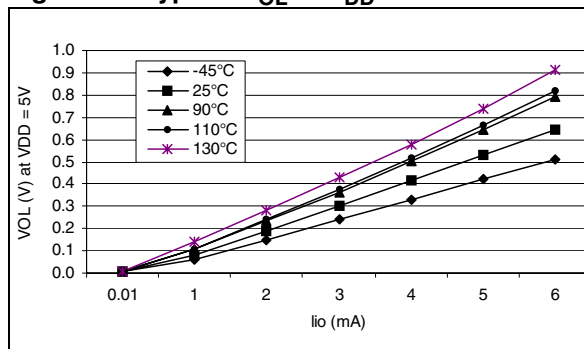
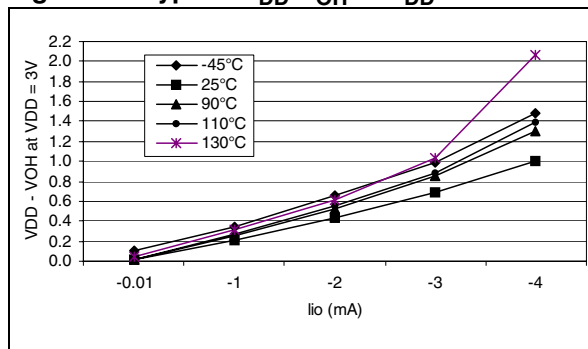
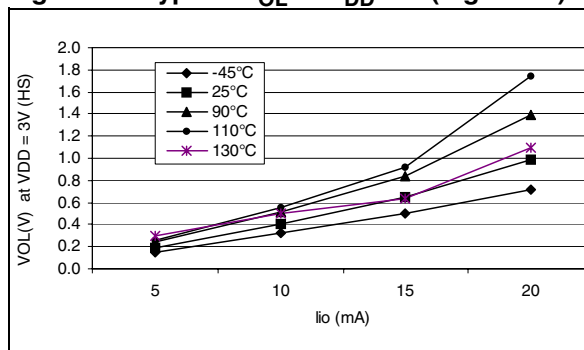
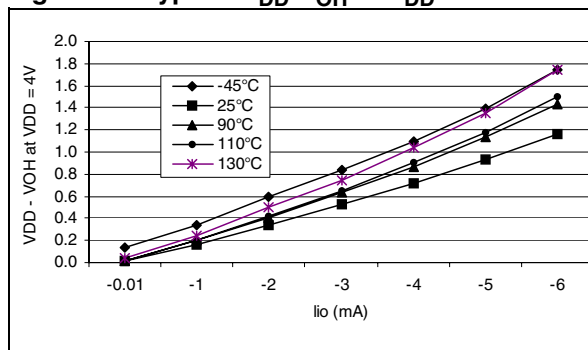
LINSICI™ SERIAL COMMUNICATION INTERFACE (LIN Master/Slave) (Cont'd)

Table 21. LINSICI1 Register Map and Reset Values

Addr. (Hex.)	Register Name	7	6	5	4	3	2	1	0
40	SCISR Reset Value	TDRE 1	TC 1	RDRF 0	IDLE 0	OR/LHE 0	NF 0	FE 0	PE 0
41	SCIDR Reset Value	DR7 -	DR6 -	DR5 -	DR4 -	DR3 -	DR2 -	DR1 -	DR0 -
42	SCIBRR LPR (LIN Slave Mode) Reset Value	SCP1 LPR7 0	SCP0 LPR6 0	SCT2 LPR5 0	SCT1 LPR4 0	SCT0 LPR3 0	SCR2 LPR2 0	SCR1 LPR1 0	SCR0 LPR0 0
43	SCICR1 Reset Value	R8 x	T8 0	SCID 0	M 0	WAKE 0	PCE 0	PS 0	PIE 0
44	SCICR2 Reset Value	TIE 0	TCIE 0	RIE 0	ILIE 0	TE 0	RE 0	RWU 0	SBK 0
45	SCICR3 Reset Value	NP 0	LINE 0	LSLV 0	LASE 0	LHDM 0	LHIE 0	LHDF 0	LSF 0
46	SCIERPR LHLR (LIN Slave Mode) Reset Value	ERPR7 LHL7 0	ERPR6 LHL6 0	ERPR5 LHL5 0	ERPR4 LHL4 0	ERPR3 LHL3 0	ERPR2 LHL2 0	ERPR1 LHL1 0	ERPR0 LHL0 0
47	SCITPR LPFR (LIN Slave Mode) Reset Value	ETPR7 LDUM 0	ETPR6 0 0	ETPR5 0 0	ETPR4 0 0	ETPR3 LPFR3 0	ETPR2 LPFR2 0	ETPR1 LPFR1 0	ETPR0 LPFR0 0

Table 22. ADC Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0034h	ADCCSR Reset Value	EOC 0	SPEED 0	ADON 0	0 0	0 0	CH2 0	CH1 0	CH0 0
0035h	ADCDRH Reset Value	D9 0	D8 0	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0
0036h	ADCRL Reset Value	0 0	0 0	0 0	0 0	SLOW 0	0 0	D1 0	D0 0

Figure 86. Typical V_{OL} at $V_{DD}=3V$ Figure 90. Typical V_{OL} at $V_{DD}=4V$ (high-sink)Figure 87. Typical V_{OL} at $V_{DD}=4V$ Figure 91. Typical V_{OL} at $V_{DD}=5V$ (high-sink)Figure 88. Typical V_{OL} at $V_{DD}=5V$ Figure 92. Typical $V_{DD}-V_{OH}$ at $V_{DD}=3.0V$ Figure 89. Typical V_{OL} at $V_{DD}=3V$ (high-sink)Figure 93. Typical $V_{DD}-V_{OH}$ at $V_{DD}=4.0V$ 

14 PACKAGE CHARACTERISTICS

14.1 PACKAGE MECHANICAL DATA

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.

Figure 105. 20-Lead Very thin Fine pitch Quad Flat No-Lead Package

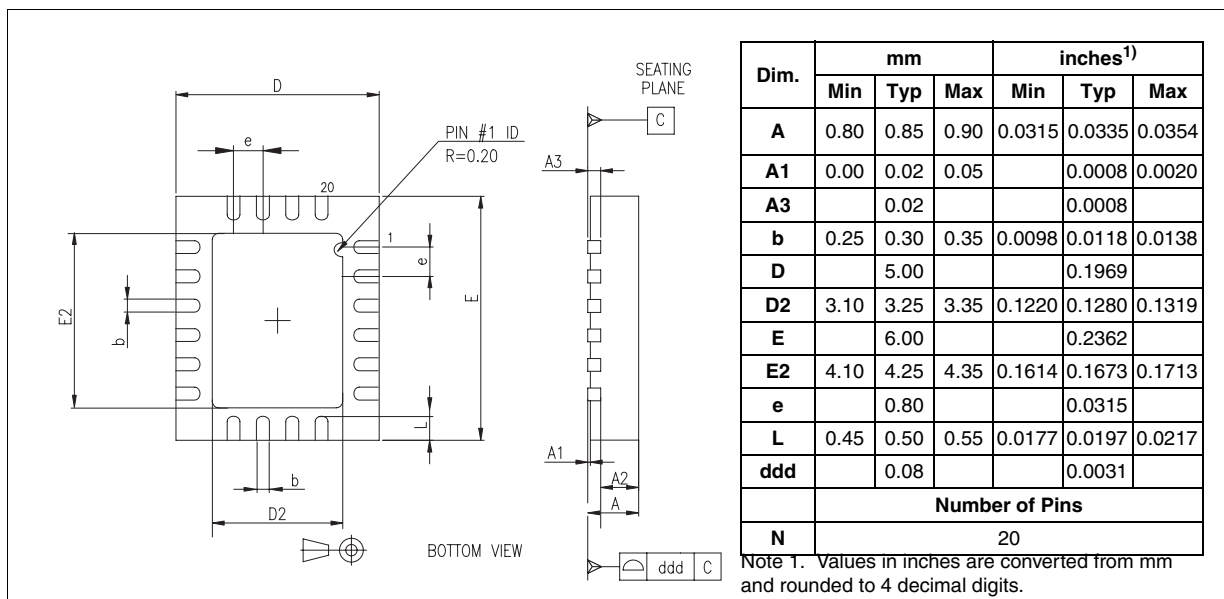
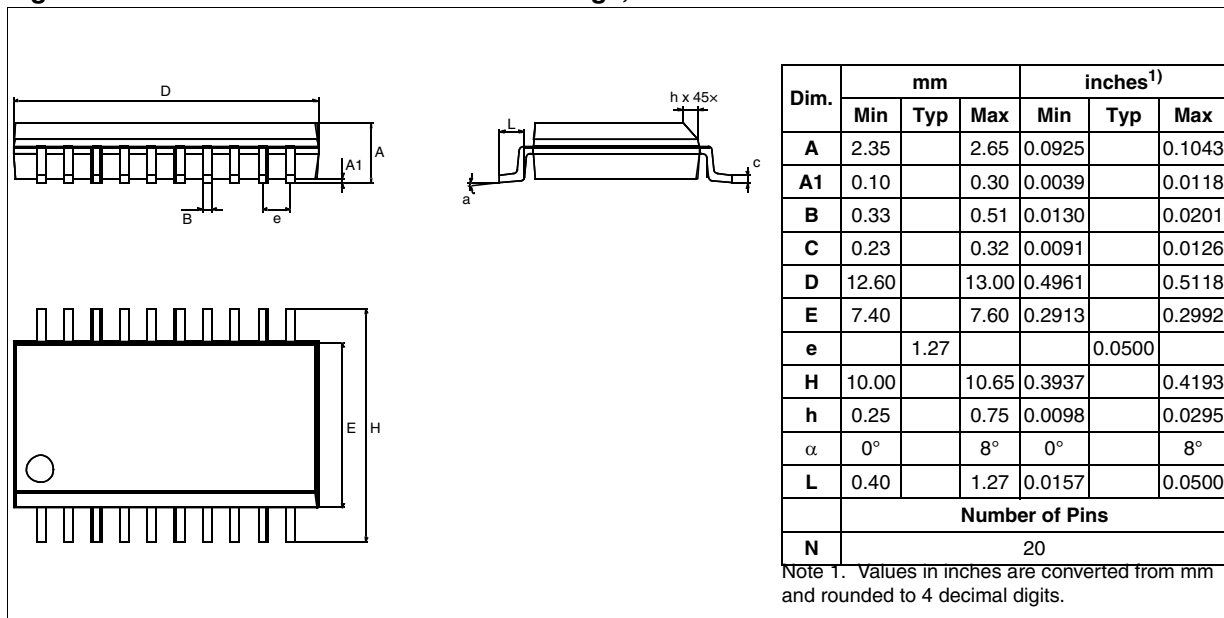


Figure 106. 20-Pin Plastic Small Outline Package, 300-mil Width



15 DEVICE CONFIGURATION

Each device is available for production in user programmable versions (FLASH) as well as in factory coded versions (ROM/FASTROM).

ST7PLITE3 devices are Factory Advanced Service Technique ROM (FASTROM) versions: they are factory programmed FLASH devices.

15.1 FLASH OPTION BYTES

The two option bytes allow the hardware configuration of the microcontroller to be selected.

OPTION BYTE 0

OPT7 = **AWUCK** *Auto Wake Up Clock Selection*

0: 32-KHz Oscillator (VLP) selected as AWU clock

1: AWU RC Oscillator selected as AWU clock.

Note: If this bit is reset, internal RC oscillator must be selected (Option OSC=0).

OPT6:4 = **OSCRANGE**[2:0] *Oscillator Range*

When the internal RC oscillator is not selected (Option OSC=1), these option bits select the range of the resonator oscillator current source or the external clock source.

			OSCRANGE		
			2	1	0
Typ. frequency range with Resonator	LP	1~2MHz	0	0	0
	MP	2~4MHz	0	0	1
	MS	4~8MHz	0	1	0
	HS	8~16MHz	0	1	1
	VLP	32.768kHz	1	0	0
External Clock on OSC1			1	0	1
Reserved			1	1	0
External Clock on PB4			1	1	1

Notes:

1. OSCRANGE[2:0] has no effect when AWUCK option is set to 0. In this case, the VLP oscillator range is automatically selected as AWU clock.

2. When the internal RC oscillator is selected, the OSCRANGE option bits must be kept at their default value to select the 256 clock cycle delay (see [section 7.5 on page 27](#))

ST7FLITE3 devices are shipped to customers with a default program memory content (FFh), while FASTROM factory coded parts contain the code supplied by the customer. This implies that FLASH devices have to be configured by the customer using the Option Bytes.

OPT 3:2 = **SEC**[1:0] *Sector 0 size definition*

These option bits indicate the size of sector 0 according to the following table.

Sector 0 Size	SEC1	SEC0
0.5k	0	0
1k	0	1
2	1	0
4k	1	1

OPT1 = **FMP_R** *Read-out protection*

Readout protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. Erasing the option bytes when the FMP_R option is selected will cause the whole memory to be erased first and the device can be reprogrammed. Refer to the ST7 Flash Programming Reference Manual and [section 4.5 on page 14](#) for more details

0: Read-out protection off

1: Read-out protection on

OPT 0 = **FMP_W** *FLASH write protection*

This option indicates if the FLASH program memory is write protected.

Warning: When this option is selected, the program memory (and the option bit itself) can never be erased or programmed again.

0: Write protection off

1: Write protection on

The option bytes have no address in the memory map and can be accessed only in programming mode (for example using a standard ST7 programming tool). The default content of the FLASH is fixed to FFh.