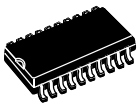
ST7L34 ST7L35 ST7L38 ST7L39

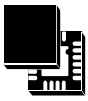
8-bit MCU for automotive with single voltage Flash/ROM, data EEPROM, ADC, timers, SPI, LINSCI™

Features

- Memories
  - 8 Kbytes program memory: Single voltage extended Flash (XFlash) or ROM with readout protection capability. In-application programming and in-circuit programming (IAP and ICP) for XFlash devices
  - 384 bytes RAM
  - 256 bytes data EEPROM (XFlash and ROM devices) with readout protection, 300 K write/erase cycles guaranteed
  - XFlash and EEPROM data retention 20 years at 55°C
- Clock, reset and supply management
  - Enhanced reset system
  - Enhanced low voltage supervisor (LVD) for main supply and an auxiliary voltage detector (AVD) with interrupt capability for implementing safe power-down procedures
  - Clock sources: Internal 1% RC oscillator, crystal/ceramic resonator or external clock
  - Optional x8 PLL for 8 MHz internal clock
  - 5 power saving modes: Halt, active halt, auto wakeup from halt, wait and slow
- I/O ports
  - Up to 15 multifunctional bidirectional I/O lines
  - 7 high sink outputs
- 5 timers
  - Configurable watchdog timer
  - Two 8-bit lite timers with prescaler, 1 real-time base and 1 input capture
  - Two 12-bit autoreload timers with 4 PWM outputs, 1 input capture and 4 output compare functions



SO20
300 mil



QFN20

- 2 communication interfaces
  - Master/slave LINSCI™ asynchronous serial interface
  - SPI synchronous serial interface
- Interrupt management
  - 10 interrupt vectors plus TRAP and reset
  - 12 external interrupt lines (on 4 vectors)
- A/D converter
  - 7 input channels
  - 10-bit resolution
- Instruction set
  - 8-bit data manipulation
  - 63 basic instructions with illegal opcode detection
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instructions
- Development tools
  - Full hardware/software development package
  - DM (debug module)

Contents

List of tables

List of figures

1 Description

The ST7L3x is a member of the ST7 microcontroller family suitable for automotive applications. All ST7 devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

The ST7L3x features Flash memory with byte-by-byte in-circuit programming (ICP) and in-application programming (IAP) capability.

Under software control, the ST7L3x 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.

Table 1. Device summary

Feature	ST7L34	ST7L35	ST7L38	ST7L39
Program memory	8 Kbytes			
RAM (stack)	384 bytes (128 bytes)			
Data EEPROM	-		256 bytes	
Peripherals	Lite timer, autoreload timer, SPI, 10-bit ADC	Lite timer, autoreload timer, SPI, 10-bit ADC, LINSCI	Lite timer, autoreload timer, SPI, 10-bit ADC	Lite timer, autoreload timer, SPI, 10-bit ADC, LINSCI
Operating supply	3.0 V to 5.5 V			
CPU frequency	Up to 8 MHz (with external resonator/clock or internal RC oscillator)			
Operating temperature	Up to -40 to 85°C / -40 to 125°C			
Packages	SO20 300mil, QFN20			

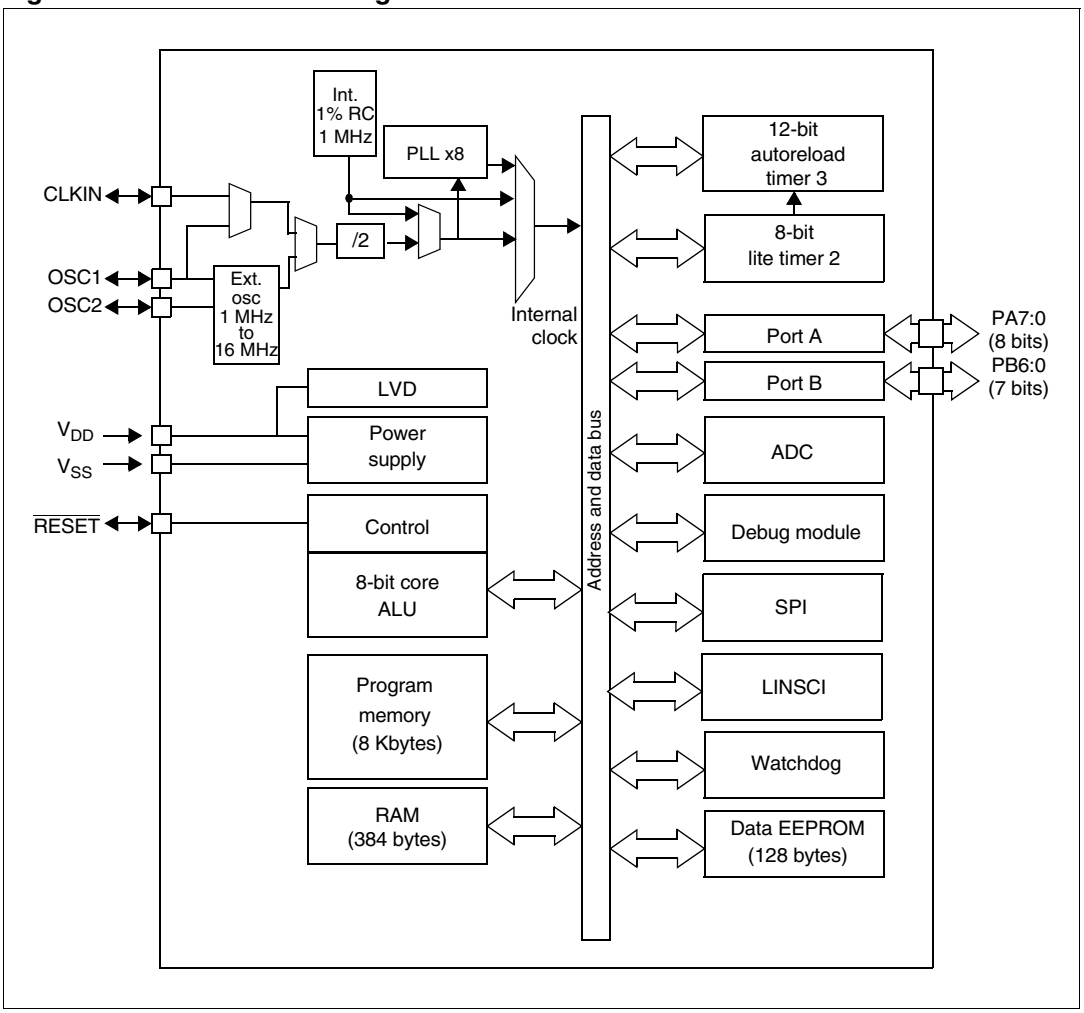
1.1 Parametric data

For easy reference, all parametric data is located in *Section 13: Electrical characteristics*.

1.2 Debug module (DM)

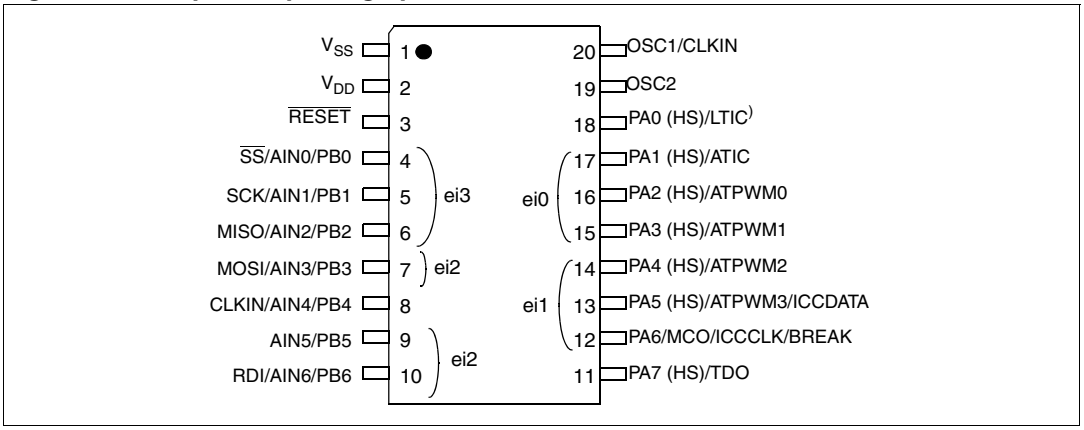
The devices feature an on-chip debug module (DM) to support in-circuit debugging (ICD). For a description of the DM registers, refer to the *ST7 ICC protocol reference manual*.

Figure 1. General block diagram



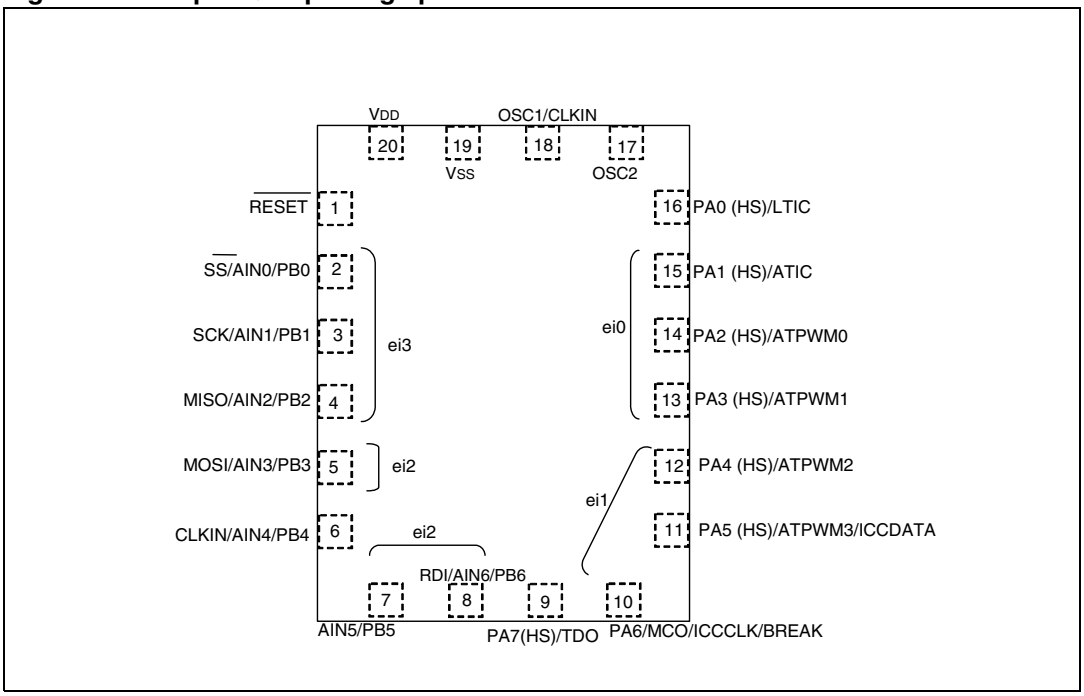
2 Pin description

Figure 2. 20-pin SO package pinout



1. eix: Associated external interrupt vector
2. (HS): 20mA high sink capability

Figure 3. 20-pin QFN package pinout



Legend/abbreviations for *Table 2*:

Type:	I = input, O = output, S = supply
Input/output level:	C_T = CMOS 0.3 V_{DD}/0.7 V_{DD} with input trigger
Output level:	HS = 20mA high sink (on N-buffer only)
Port and control configuration inputs:	float = floating, wpu = weak pull-up, int = interrupt, ana = analog ports
Port and control configuration outputs:	OD = open drain, PP = push-pull

Note: The reset configuration of each pin (shown in bold) is valid as long as the device is in reset state.

Table 2. Device pin description

Pin no.		Pin name	Type	Level		Port/control						Main function (after reset)	Alternate function
						Input(1)				Output			
SO20	QFN20			Input	Output	float	wpu	int	ana	OD	PP		
1	19	V_{SS}	S									Ground	
2	20	V_{DD}	S									Main power supply	
3	1	$\overline{\text{RESET}}$	I/O	C_T			X			X		Top priority non maskable interrupt (active low)	
4	2	PB0/AIN0/$\overline{\text{SS}}$	I/O	C_T		X	ei3		X	X	X	Port B0	ADC analog input 0 or SPI slave select (active low)
5	3	PB1/AIN1/SCK	I/O	C_T		X			X	X	X	Port B1	ADC analog input 1 or SPI serial clock
6	4	PB2/AIN2/MISO	I/O	C_T		X			X	X	X	Port B2	ADC analog input 2 or SPI master in/slave out data
7	5	PB3/AIN3/MOSI	I/O	C_T		X	ei2		X	X	X	Port B3	ADC analog input 3 or SPI master out/slave in data
8	6	PB4/AIN4/CLKIN/	I/O	C_T		X	X		X	X	X	Port B4	ADC analog input 4 or external clock input
9	7	PB5/AIN5	I/O	C_T		X	ei2		X	X	X	Port B5	ADC analog input 5
10	8	PB6/AIN6/RDI	I/O	C_T		X			X	X	X	Port B6	ADC analog input 6 or LINSCI input
11	9	PA7/TDO	I/O	C_T	HS	X	X			X	X	Port A7	LINSCI output

Table 2. Device pin description (continued)

Pin no.		Pin name	Type	Level		Port/control						Main function (after reset)	Alternate function
						Input[1]				Output			
SO20	QFN20			Input	Output	float	wpu	int	ana	OD	PP		
12	10	PA6 /MCO/ICCCLK/ BREAK	I/O	C_T		X	ei1			X	X	Port A6	Main clock output or in-circuit communication clock or external BREAK **Caution:** During normal operation this pin must be pulled- up, internally or externally (external pull-up of 10 k 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 puts it back in input pull-up
13	11	PA5/ICCDATA/ ATPWM3	I/O	C_T	HS	X				X	X	Port A5	Autoreload timer PWM3 or in-circuit communication data
14	12	PA4/ATPWM2	I/O	C_T	HS	X				X	X	Port A4	Autoreload timer PWM2
15	13	PA3/ATPWM1	I/O	C_T	HS	X	ei0			X	X	Port A3	Autoreload timer PWM1
16	14	PA2/ATPWM0	I/O	C_T	HS	X				X	X	Port A2	Autoreload timer PWM0
17	15	PA1/ATIC	I/O	C_T	HS	X				X	X	Port A1	Autoreload timer input capture
18	16	PA0/LTIC	I/O	C_T	HS	X	X			X	X	Port A0	Lite timer input capture
19	17	OSC2	O									Resonator oscillator inverter output	
20	18	OSC1/CLKIN	I			X						Resonator oscillator inverter input or external clock input	

1. For input with interrupt possibility 'ei_x' defines the associated external interrupt vector which can be assigned to one of the I/O pins using the EISR register. Each interrupt can be either weak pull-up or floating defined through option register OR.

3 Register and memory map

As shown in *Figure 4*, the MCU can address 64 Kbytes of memories and I/O registers.

The available memory locations consist of 128 bytes of register locations, 384 bytes of RAM, 256 bytes of data EEPROM and up to 8 Kbytes of user program memory. The RAM space includes up to 128 bytes for the stack from 180h to 1FFh.

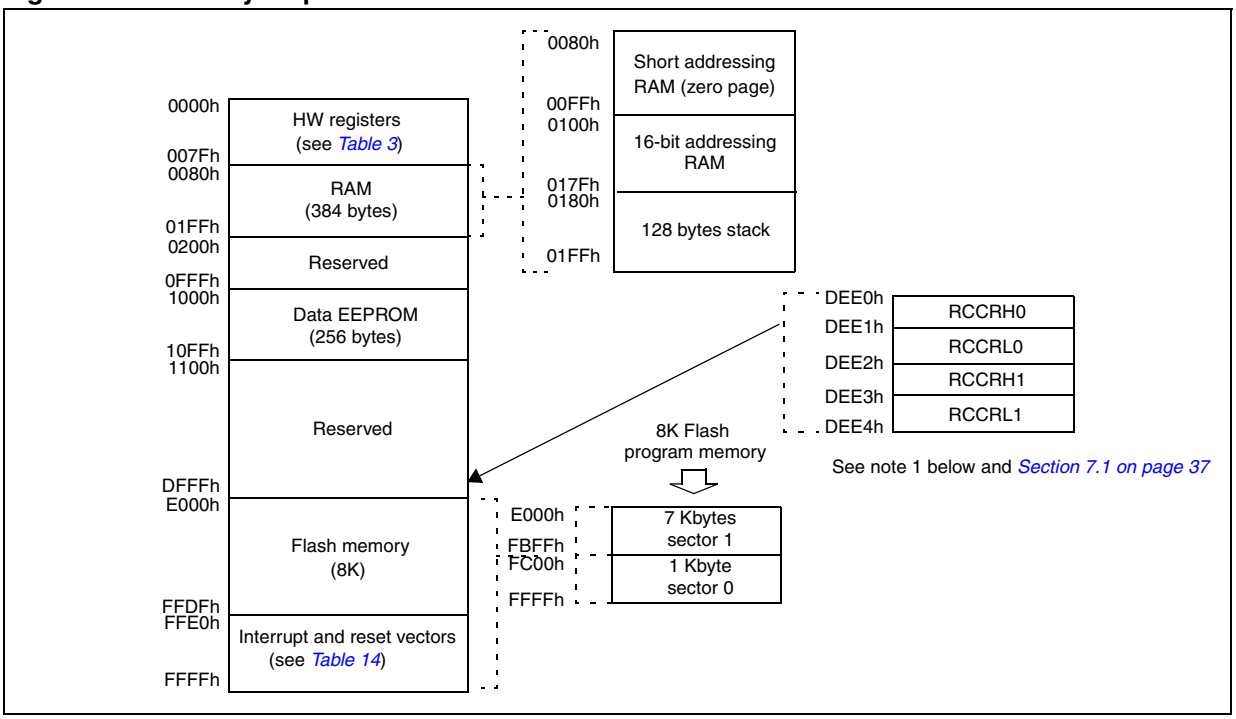
The highest address bytes contain the user reset and interrupt vectors.

The Flash memory contains two sectors (see *Figure 4*) mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

The size of Flash sector 0 and other device options are configurable by option byte (refer to *Section 15.2: Option bytes on page 215*).

Note: Memory locations marked as 'Reserved' must never be accessed. Accessing a reserved area can have unpredictable effects on the device.

Figure 4. Memory map



1. *DEE0h, DEE1h, DEE2h and DEE3h addresses are located in a reserved area but are special bytes containing also the RC calibration values which are read-accessible only in user mode. If all the EEPROM data or Flash space (including the RC calibration values locations) has been erased (after the readout protection removal), then the RC calibration values can still be obtained through these four addresses.*

Legend for *Table 3*: x = undefined, R/W = read/write, RO = read only

Table 3. Hardware register map

Address	Block	Register label	Register name	Reset status	Remarks
0000h	Port A	PADR	Port A data register	FFh(1)	R/W
0001h		PADDR	Port A data direction register	00h	R/W
0002h		PAOR	Port A option register	40h	R/W
0003h	Port B	PBDR	Port B data register	FFh(1)	R/W
0004h		PBDDR	Port B data direction register	00h	R/W
0005h		PBOR	Port B option register	00h	R/W(2)
0006h 0007h	Reserved area (2 bytes)				
0008h	Lite timer 2	LTCSR2	Lite timer control/status register 2	0Fh	R/W
0009h		LTARR	Lite timer autoreload register	00h	R/W
000Ah		LTCNTR	Lite timer counter 2 register	00h	RO
000Bh		LTCSR1	Lite timer control/status register 1	0x00 0000b	R/W
000Ch		LTICR	Lite timer input capture register	xxh	RO
000Dh	Auto-reload timer 3	ATCSR	Timer control/status register	0x00 0000b	R/W
000Eh		CNTR1H	Counter register 1 high	00h	RO
000Fh		CNTR1L	Counter register 1 low	00h	RO
0010h		ATR1H	Autoreload register 1 high	00h	R/W
0011h		ATR1L	Autoreload register 1 low	00h	R/W
0012h		PWMCR	PWM output control register	00h	R/W
0013h		PWM0CSR	PWM 0 control/status register	00h	R/W
0014h		PWM1CSR	PWM 1 control/status register	00h	R/W
0015h		PWM2CSR	PWM 2 control/status register	00h	R/W
0016h		PWM3CSR	PWM 3 control/status register	00h	R/W
0017h		DCR0H	PWM 0 duty cycle register high	00h	R/W
0018h		DCR0L	PWM 0 duty cycle register low	00h	R/W
0019h		DCR1H	PWM 1 duty cycle register high	00h	R/W
001Ah		DCR1L	PWM 1 duty cycle register low	00h	R/W
001Bh		DCR2H	PWM 2 duty cycle register high	00h	R/W
001Ch		DCR2L	PWM 2 duty cycle register low	00h	R/W
001Dh		DCR3H	PWM 3 duty cycle register high	00h	R/W
001Eh		DCR3L	PWM 3 duty cycle register low	00h	R/W
001Fh		ATICRH	Input capture register high	00h	RO
0020h		ATICRL	Input capture register low	00h	RO
0021h		ATCSR2	Timer control/status register 2	03h	R/W
0022h		BREAKCR	Break control register	00h	R/W
0023h		ATR2H	Autoreload register 2 high	00h	R/W
0024h		ATR2L	Autoreload register 2 low	00h	R/W
0025h		DTGR	Dead time generator register	00h	R/W
0026h to 002Dh	Reserved area (8 bytes)				
002Eh	WDG	WDGCR	Watchdog control register	7Fh	R/W
0002Fh	Flash	FCSR	Flash control/status register	00h	R/W
00030h	EEPROM	EECSR	Data EEPROM control/status register	00h	R/W

Table 3. Hardware register map (continued)

Address	Block	Register label	Register name	Reset status	Remarks
0031h 0032h 0033h	SPI	SPIDR SPICR SPICSR	SPI data I/O register SPI control register SPI control/status register	xxh 0xh 00h	R/W R/W R/W
0034h 0035h 0036h	ADC	ADCCSR ADCDRH ADCDRL	A/D control/status register A/D data register high A/D control and data register low	00h xxh x0h	R/W RO R/W
0037h	ITC	EICR	External interrupt control register	00h	R/W
0038h	MCC	MCCSR	Main clock control/status register	00h	R/W
0039h 003Ah	Clock and reset	RCCR SICSR	RC oscillator control register System integrity control/status register	FFh 0110 0xx0b	R/W R/W
003Bh	Reserved area (1 byte)				
003Ch	ITC	EISR	External interrupt selection register	00h	R/W
003Dh to 003Fh	Reserved area (3 bytes)				
0040h 0041h 0042h 0043h 0044h 0045h 0046h 0047h	LINSCI (LIN master/ slave)	SCISR SCIDR SCIBRR SCICR1 SCICR2 SCICR3 SCIERPR SCIETPR	SCI status register SCI data register SCI baud rate register SCI control register 1 SCI control register 2 SCI control register 3 SCI extended receive prescaler register SCI extended transmit prescaler register	C0h xxh 00xx xxxxb xxh 00h 00h 00h 00h	RO R/W R/W R/W R/W R/W R/W R/W
0048h	Reserved area (1 byte)				
0049h 004Ah	AWU	AWUPR AWUCSR	AWU prescaler register AWU control/status register	FFh 00h	R/W R/W
004Bh 004Ch 004Dh 004Eh 004Fh 0050h	DM(3)	DMCR DMSR DMBK1H DMBK1L DMBK2H DMBK2L	DM control register DM status register DM breakpoint register 1 high DM breakpoint register 1 low DM breakpoint register 2 high DM breakpoint register 2 low	00h 00h 00h 00h 00h 00h	R/W R/W R/W R/W R/W R/W
0051h to 007Fh	Reserved area (47 bytes)				

1. The contents of the I/O port DR registers are readable only in output configuration. In input configuration, the values of the I/O pins are returned instead of the DR register contents
2. The bits associated with unavailable pins must always keep their reset value
3. For a description of the debug module registers, see *ST7 ICC protocol reference manual*

4 Flash program memory

4.1 Introduction

The ST7 single voltage extended Flash (XFlash) is a non-volatile memory that can be electrically erased and programmed either on a byte-by-byte basis or up to 32 bytes in parallel.

The XFlash devices can be programmed off-board (plugged in a programming tool) or on-board using in-circuit programming (ICP) or in-application programming (IAP).

The array matrix organization allows each sector to be erased and reprogrammed without affecting other sectors.

4.2 Main features

- In-circuit programming (ICP)
- In-application programming (IAP)
- In-circuit testing (ICT) for downloading and executing user application test patterns in RAM
- Sector 0 size configurable by option byte
- Readout and write protection

4.3 Programming modes

The ST7 can be programmed in three different ways:

– Insertion in a programming tool. In this mode, Flash sectors 0 and 1, option byte row and data EEPROM (if present) can be programmed or erased.

– In-circuit programming. In this mode, Flash sectors 0 and 1, option byte row and data EEPROM (if present) can be programmed or erased without removing the device from the application board.

– In-application programming. In this mode, sector 1 and data EEPROM (if present) can be programmed or erased without removing the device from the application board and while the application is running.

4.3.1 In-circuit programming (ICP)

ICP uses a protocol called ICC (in-circuit communication) which allows an ST7 plugged on a printed circuit board (PCB) to communicate with an external programming device connected via a cable. ICP is performed in three steps:

– Switch the ST7 to ICC mode (in-circuit communications). This is done by driving a specific signal sequence on the ICCCLK/DATA pins while the $\overline{\text{RESET}}$ pin is pulled low. When the ST7 enters ICC mode, it fetches a specific reset vector which points to the ST7 system memory containing the ICC protocol routine. This routine enables the ST7 to receive bytes from the ICC interface.

– Download ICP driver code in RAM from the ICCDATA pin

– Execute ICP driver code in RAM to program the Flash memory

Depending on the ICP driver code downloaded in RAM, Flash memory programming can be fully customized (number of bytes to program, program locations, or selection of the serial communication interface for downloading).

4.3.2 In-application programming (IAP)

This mode uses an IAP driver program previously programmed in sector 0 by the user (in ICP mode).

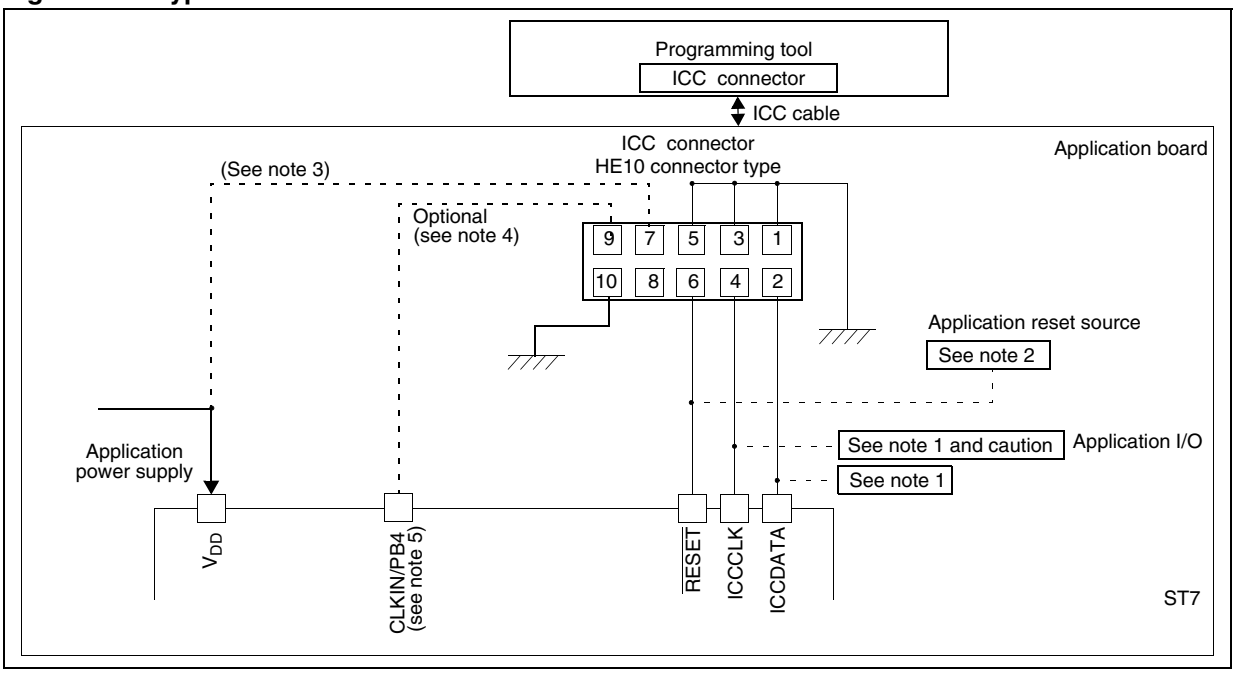
IAP mode is fully controlled by user software, allowing it to be adapted to the user application (such as a user-defined strategy for entering programming mode or a choice of communications protocol used to fetch the data to be stored). This mode can be used to program any memory areas except sector 0, which is write/erase protected to allow recovery in case errors occur during the programming operation.

4.4 ICC interface

ICP needs a minimum of four and up to six 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/PB4: Main clock input for external source
- V_{DD}: Application board power supply (optional, see note 3, *Figure 5: Typical ICC interface on page 24*)

Figure 5. Typical ICC interface



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 must 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 5 mA at high level (push-pull output or pull-up resistor < 1K). A schottky diode can be used to isolate the application 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 must be connected to the PB4 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. ST7 devices with multi-oscillator capability must have OSC2 grounded in this case.
5. With any programming tool, while the ICP option is disabled, the external clock must be provided on PB4.
6. In ICC mode, the internal RC oscillator is forced as a clock source, regardless of the selection in the option byte.

Caution: During normal operation the 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 puts it back in input pull-up.

4.5 Memory protection

There are two different types of memory protection: Readout protection and write/erase protection, which can be applied individually.

4.5.1 Readout protection

Readout protection, when selected, protects against program memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller. Both program and data EE memory are protected.

In Flash devices, this protection is removed by reprogramming the option. In this case, both program and data EE memory are automatically erased and the device can be reprogrammed.

Readout protection selection depends on the device type:

- In Flash devices it is enabled and removed through the FMP_R bit in the option byte.
- In ROM devices it is enabled by the mask option specified in the option list.

4.5.2 Flash write/erase protection

Write/erase protection, when set, makes it impossible to both overwrite and erase program memory. It does not apply to EE data. Its purpose is to provide advanced security to applications and prevent any change being made to the memory content.

Warning: Once set, write/erase protection can never be removed. A write-protected Flash device is no longer reprogrammable.

Write/erase protection is enabled through the FMP_W bit in the option byte.

4.6 Related documentation

For details on Flash programming and ICC protocol, refer to the *ST7 Flash programming reference manual* and to the *ST7 ICC protocol reference manual*.

4.7 Register description

Flash control/status register (FCSR)

FCSR

Reset value: 0000 0000 (00h)
1st RASS key: 0101 0110 (56h)
2nd RASS key: 10101110 (AEh)

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	OPT	LAT	PGM
-	-	-	-	-	R/W	R/W	R/W

Note: This register is reserved for programming using ICP, IAP or other programming methods. It controls the XFlash programming and erasing operations.

When an EPB or another programming tool is used (in socket or ICP mode), the RASS keys are sent automatically.

5 Data EEPROM

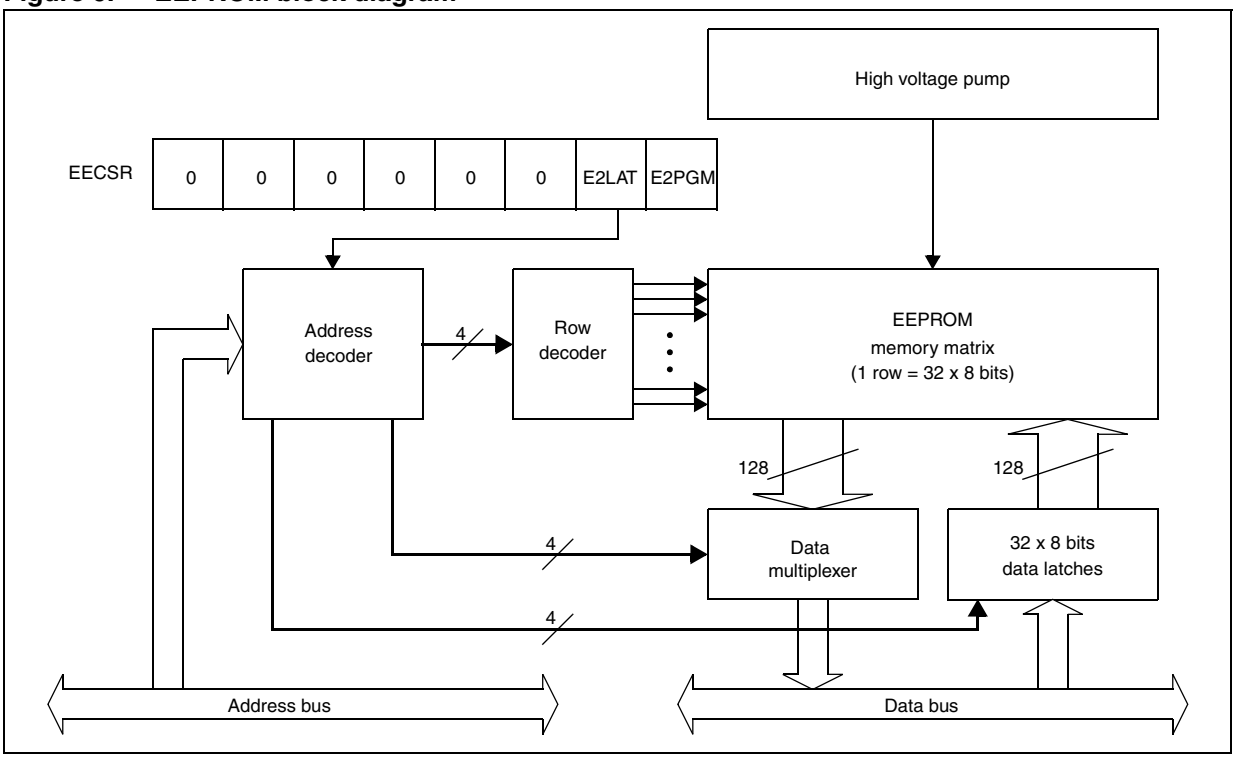
5.1 Introduction

The electrically erasable programmable read only memory can be used as a non volatile back-up for storing data. Using the EEPROM requires a basic access protocol described in this chapter.

5.2 Main features

- Up to 32 bytes programmed in the same cycle
- EEPROM mono-voltage (charge pump)
- Chained erase and programming cycles
- Internal control of the global programming cycle duration
- Wait mode management
- Readout protection

Figure 6. EEPROM block diagram



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: Data EEPROM programming flowchart on page 29* 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. Do not 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 must 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 ensure 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 over-programs the memory (logical AND between the two write access data results) 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 Figure 9: Data EEPROM programming cycle on page 31.

Figure 7. Data EEPROM programming flowchart

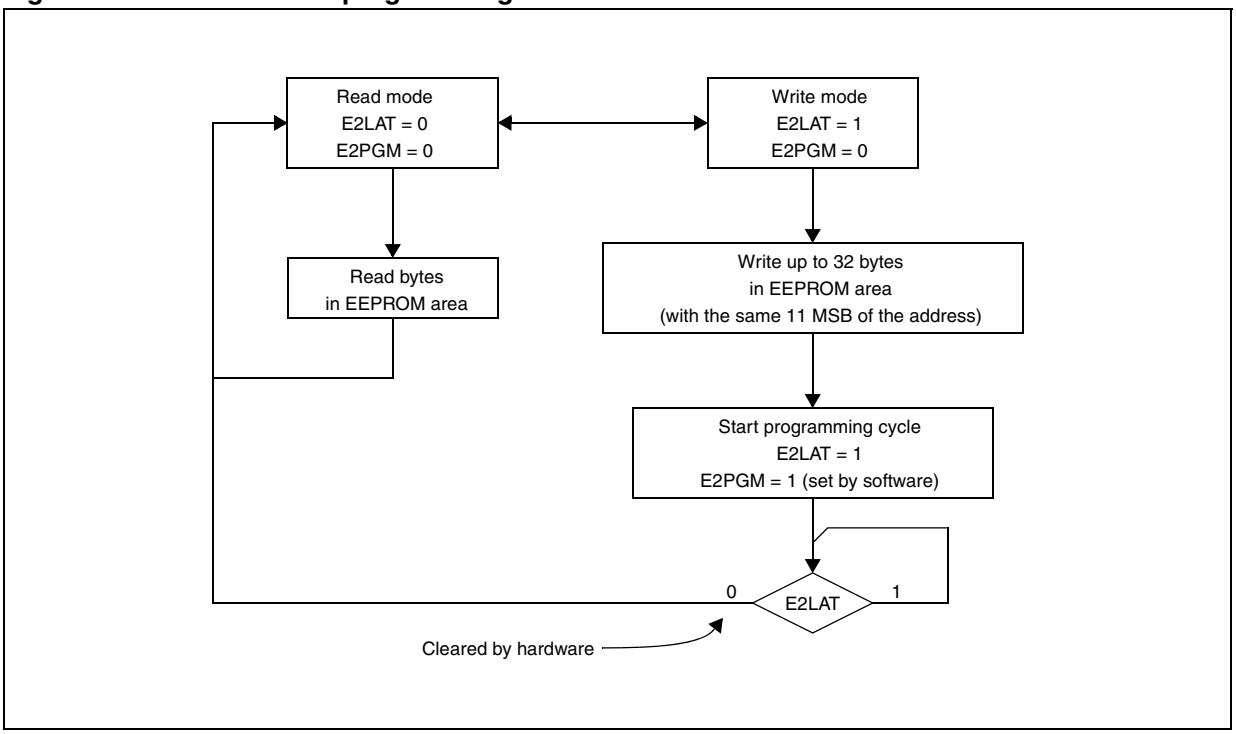
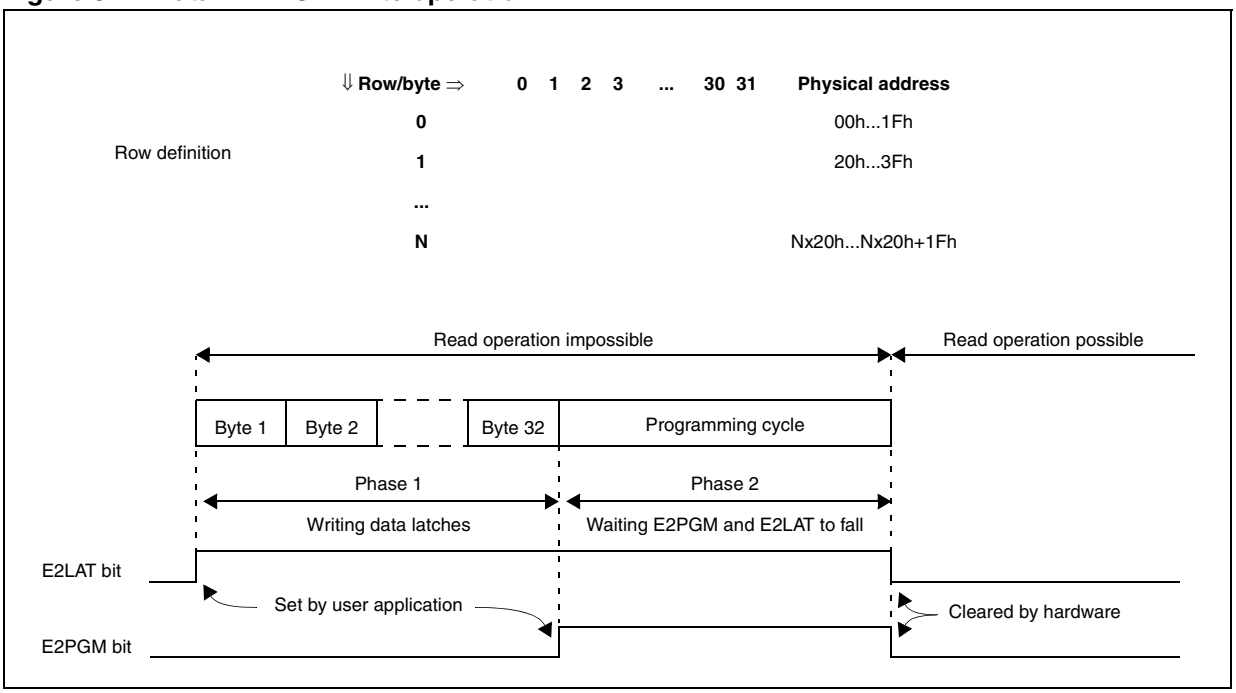


Figure 8. Data EEPROM write operation



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

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 immediately enters this mode if there is no programming in progress, otherwise the data EEPROM finishes the cycle and then enters 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 stops 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 is not driven.

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

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

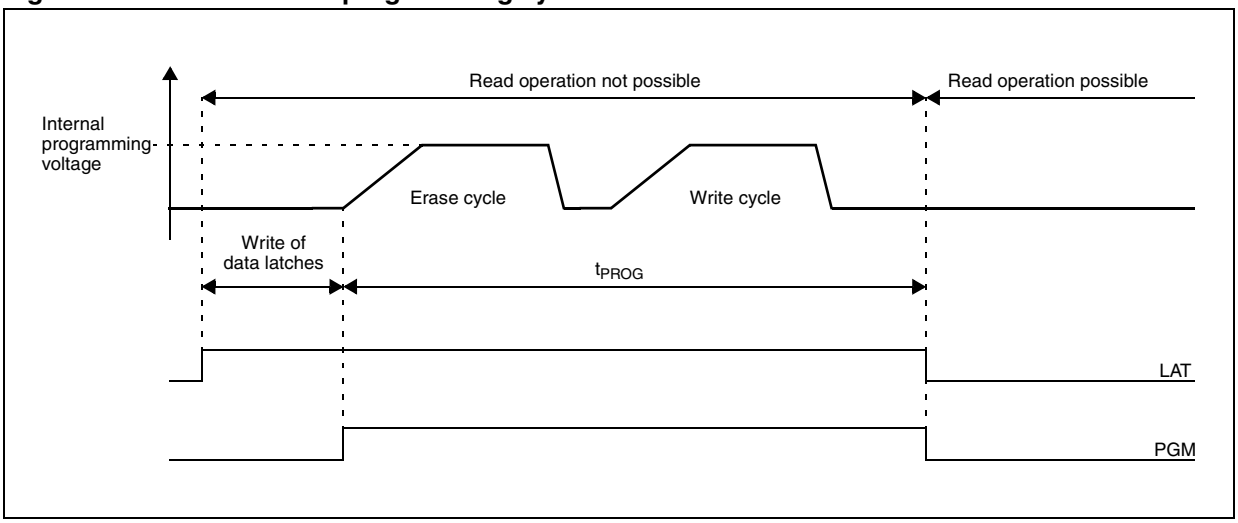
5.6 Data EEPROM readout protection

The readout protection is enabled through an option bit (see *Section 15.2: Option bytes on page 215*).

When this option is selected, the programs and data stored in the EEPROM memory are protected against readout (including a rewrite 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 9. Data EEPROM programming cycle



5.7 Register description

EEPROM control/status register (EECSR)

EECSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	E2LAT	E2PGM
-	-	-	-	-	-	R/W	R/W

Table 4. EECSR register description

Bit	Bit name	Function
7:2	-	Reserved, forced by hardware to 0
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
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 5. Data EEPROM register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0030h	EECSR Reset value	0	0	0	0	0	0	E2LAT 0	E2PGM 0

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.

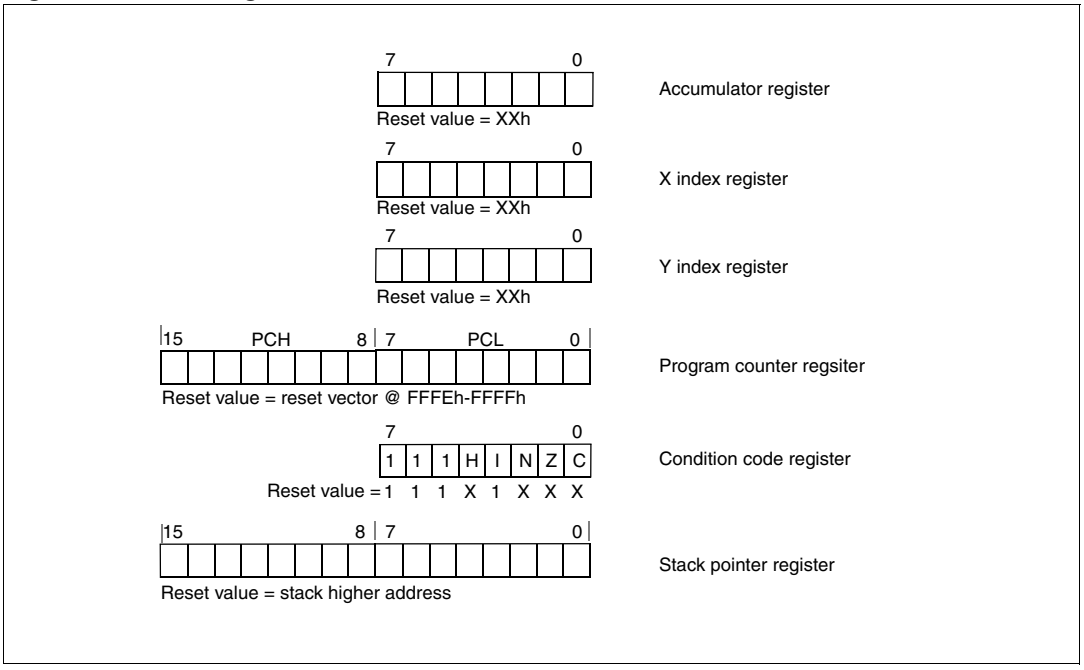
6.2 Main features

- 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.

Figure 10. CPU registers



1. X = undefined value

Accumulator register (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 register (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).

Condition code register (CC)

CC Reset value: 111x 1xxx

7	6	5	4	3	2	1	0
1	1	1	H	I	N	Z	C
			R/W	R/W	R/W	R/W	R/W

The 8-bit condition code register contains the interrupt masks 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.

Table 6. CC register description

Bit	Bit name	Function
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 instructions. 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.

Table 6. CC register description (continued)

Bit	Bit name	Function
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.*
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 7^{th} bit of the result. 0: The result of the last operation is positive or null 1: The result of the last operation is negative (in other words, the most significant bit is a logic 1) This bit is accessed by the JRMI and JRPL test instructions.
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.
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.

Stack pointer register (SP)

SP — Reset value: 01 FFh

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	1
-	-	-	-	-	-	-	R/W

7	6	5	4	3	2	1	0
1	SP[6:0]						
R/W	R/W						

The stack pointer is a 16-bit register which always points 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: Stack manipulation example on page 36*).

Since the stack is 128 bytes deep, the 9 most significant bits are forced by hardware. Following an MCU reset, or after a reset stack pointer instruction (RSP), the stack pointer contains its reset value (the SP6 to SP0 bits are set) which is the stack higher address.

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

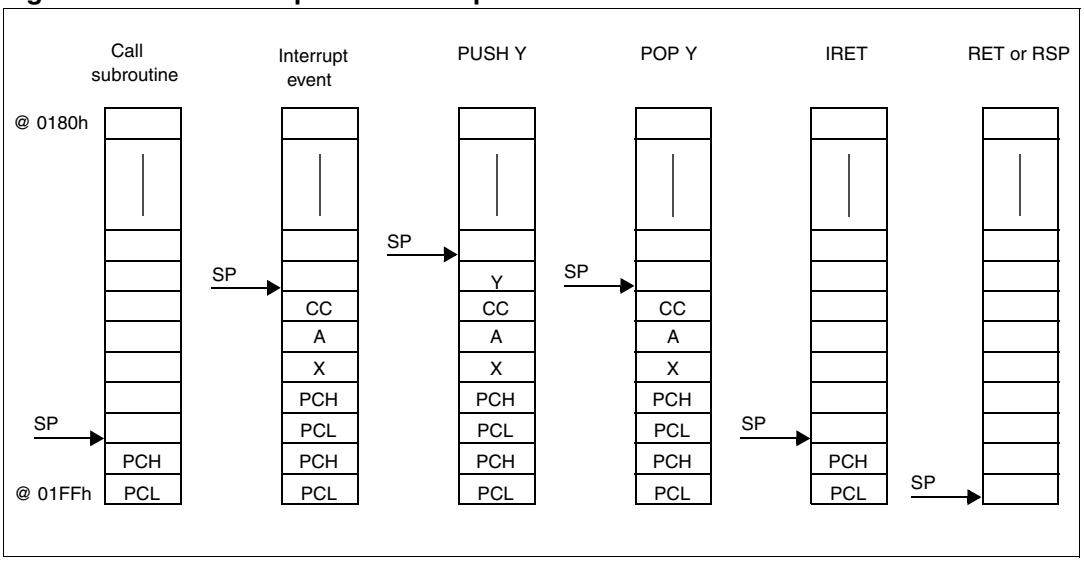
Note: When the lower limit is exceeded, the stack pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in *Figure 11: Stack manipulation example on page 36*.

- When an interrupt is received, the SP is decremented and the context is pushed on the stack
- On return from interrupt, the SP is incremented and the context is popped from the stack

A subroutine call occupies two locations and an interrupt five locations in the stack area.

Figure 11. Stack manipulation example



1. Legend: stack higher address = 01FFh; stack lower address = 0180h

7 Supply, reset and clock management

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

Main features

- Clock management
  - 1 MHz internal RC oscillator (enabled by option byte)
  - 1 to 16 MHz or 32 kHz external crystal/ceramic resonator (selected by option byte)
  - External clock input (enabled by option byte)
  - PLL for multiplying the frequency by 8 (enabled by option byte)
- Reset sequence manager (RSM)
- System integrity management (SI)
  - Main supply low voltage detection (LVD) with reset generation (enabled by option byte)
  - Auxiliary voltage detector (AVD) with interrupt capability for monitoring the main supply (enabled by option byte)

7.1 Internal RC oscillator adjustment

The device contains an internal RC oscillator with high accuracy for a given device, temperature and voltage. It must be calibrated to obtain the frequency required in the application. This is done by the software writing an 8-bit calibration value in the RCCR (RC control register) and in the bits [6:5] in the SICSR (SI control status register).

Whenever the microcontroller is reset, the RCCR returns to its default value (FFh), that is, each time the device is reset, the calibration value must be loaded in the RCCR. Predefined calibration values are stored in EEPROM for 3.3 V and 5 V V_{DD} supply voltages at 25°C, as shown in *Table 7*.

Table 7. RCCR calibration registers

RCCR	Conditions	ST7L3 addresses
RCCRH0	V_{DD} = 5 V T_A = 25°C f_{RC} = 1 MHz(1)	DEE0h(2) (CR[9:2] bits)
RCCRL0		DEE1h(2) (CR[1:0] bits)
RCCRH1	V_{DD} = 3.3 V T_A = 25°C f_{RC} = 1 MHz(1)	DEE2h(2)(CR[9:2] bits)
RCCRL1		DEE3h(2) (CR[1:0] bits)

1. RCCR0 and RCCR1 calibrated within these conditions in order to reach RC accuracy as mentioned in *Table 101: Operating conditions (tested for TA = -40 to +125 °C) @ VDD = 4.5 to 5.5 V on page 183* and *Table 103: Operating conditions (tested for TA = -40 to +125 °C) @ VDD = 3.0 to 3.6 V on page 184*

2. DEE0h, DEE1h, DEE2h and DEE3h addresses are located in a reserved area but are special bytes containing also the RC calibration values which are read-accessible only in user mode. If all the EEPROM data or Flash space (including the RC calibration values locations) has been erased (after the readout protection removal), then the RC calibration values can still be obtained through these four addresses. For compatibility reasons with the SICSR register, CR[1:0] bits are stored in the fifth and sixth positions of DEE1 and DEE3 addresses.

Note: 1 In ICC mode, the internal RC oscillator is forced as a clock source, regardless of the selection in the option byte.

2 For more information on the frequency and accuracy of the RC oscillator see Section 13: Electrical characteristics.

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

4 These bytes are systematically programmed by ST, including on FASTROM devices. Consequently, customers intending to use FASTROM service must not use these bytes.

5 RCCR0 and RCCR1 calibration values are not erased if the readout protection bit is reset after it has been set (see Section 4.5.1: Readout protection on page 25).

Caution: If the voltage or temperature conditions change in the application, the frequency may need to be recalibrated.

Refer to application note AN1324 for information on how to calibrate the RC frequency using an external reference signal.

7.2 Phase locked loop

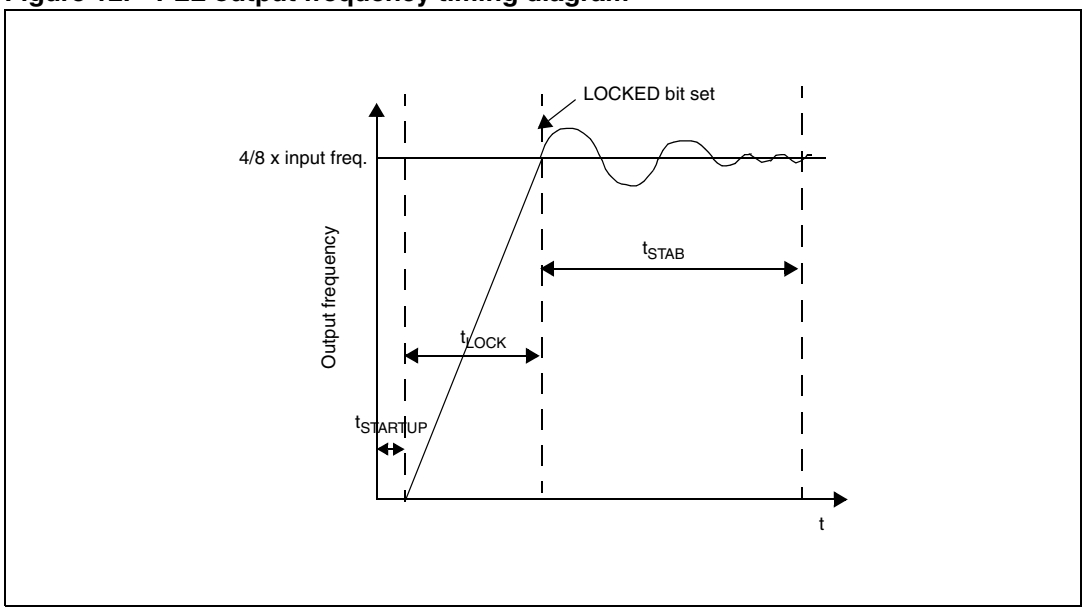
The PLL can be used to multiply a 1 MHz frequency from the RC oscillator or the external clock by 8 to obtain an f_{OSC} of 8 MHz. The PLL is enabled (by 1 option bit) and the multiplication factor is 8.

- The x8 PLL is intended for operation with V_{DD} in the 3.6 V to 5.5 V range.

If the PLL is disabled and the RC oscillator is enabled, then f_{OSC} = 1 MHz.

If both the RC oscillator and the PLL are disabled, f_{OSC} is driven by the external clock.

Figure 12. PLL output frequency timing diagram



When the PLL is started, after reset or wakeup from halt mode or AWUFH mode, it outputs the clock after a delay of $t_{STARTUP}$.

When the PLL output signal reaches the operating frequency, the locked bit in the SICSCR register is set. Full PLL accuracy (ACC_{PLL}) is reached after a stabilization time of t_{STAB} (see *Figure 12* and *Section 13.3.4: Internal RC oscillator and PLL on page 188*).

Refer to *Section 7.6.4: Register description on page 48* for a description of the locked bit in the SICSR register.

7.3 Register description

Main clock control/status register (MCCSR)

MCCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	MCO	SMS
-	-	-	-	-	-	R/W	R/W

Table 8. MCCSR register description

Bit	Bit name	Function
7:2	-	Reserved, must be kept cleared
1	MCO	Main clock out enable This bit is read/write by software and cleared by hardware after a reset. This bit enables the MCO output clock. 0: MCO clock disabled, I/O port free for general purpose I/O 1: MCO clock enabled
0	SMS	Slow mode select This bit is read/write by software and cleared by hardware after a reset. This bit selects the input clock f_{OSC} or $f_{OSC}/32$. 0: Normal mode ($f_{CPU} = f_{OSC}$) 1: Slow mode ($f_{CPU} = f_{OSC}/32$)

RC control register (RCCR)

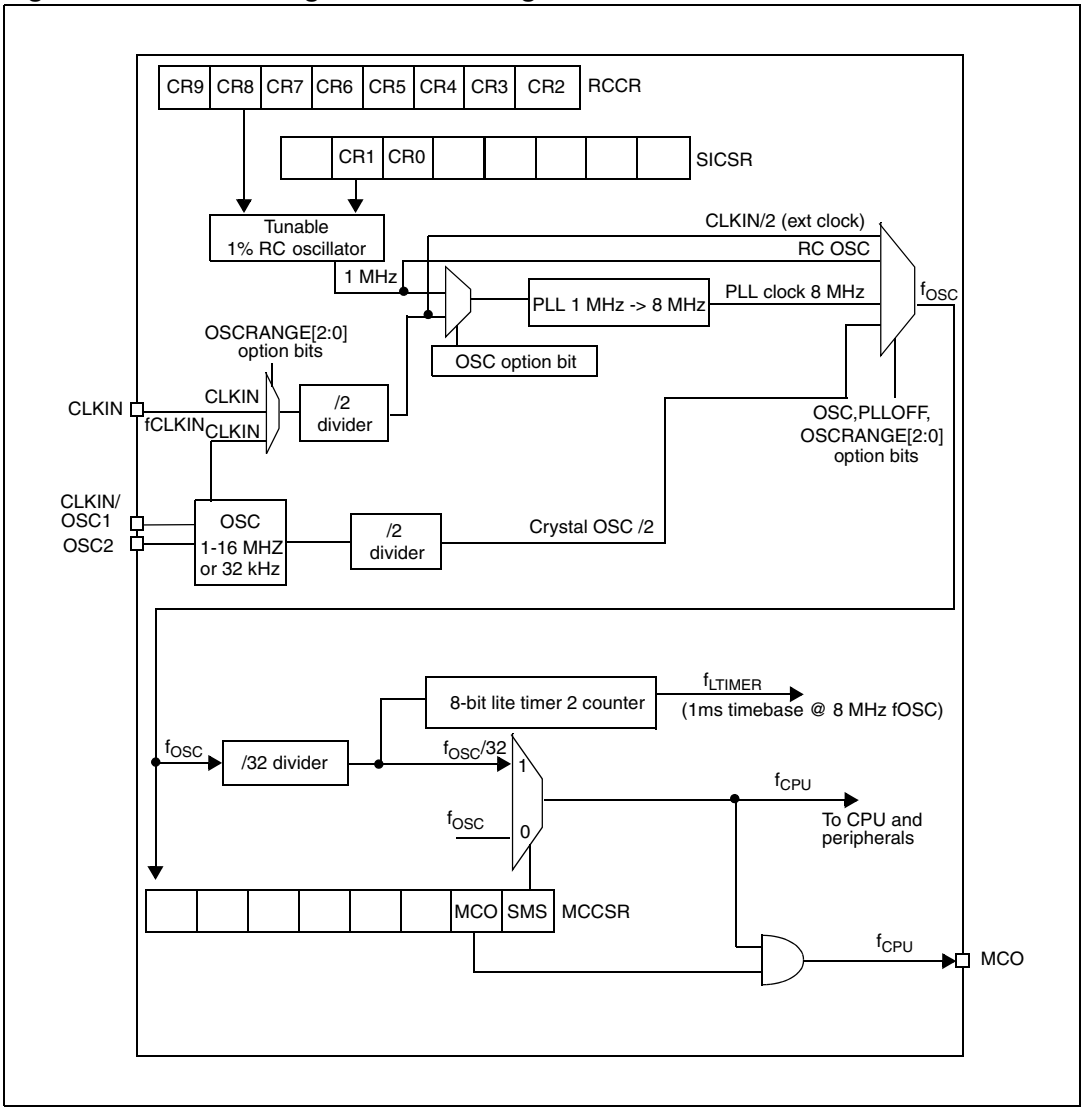
RCCR Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
CR[9:2]							
R/W							

Table 9. RCCR register description

Bit	Bit name	Function
7:0	CR[9:2]	RC oscillator frequency adjustment bits These bits must be written immediately after reset to adjust the RC oscillator frequency and to obtain an accuracy of 1%. The application can store the correct value for each voltage range in EEPROM and write it to this register at startup. 00h = Maximum available frequency FFh = Lowest available frequency These bits are used with the CR[1:0] bits in the SICSR register. Refer to *Section 7.6.4: Register description on page 48*. *Note: To tune the oscillator, write a series of different values in the register until the correct frequency is reached. The fastest method is to use a dichotomy starting with 80h.*

Figure 13. Clock management block diagram



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 16 MHz or 32 kHz):

- 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 14: ST7 clock sources on page 42*. Refer to *Section 13: Electrical characteristics* for more details.

7.4.1 External clock source

In external clock mode, a clock signal (square, sinus or triangle) with ~50% duty cycle must 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 the external clock.

7.4.2 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 four oscillators with different frequency ranges has to be done by option byte in order to reduce consumption (refer to *Section 15.2 on page 215* for more details on the frequency ranges). In this mode of the multi-oscillator, the resonator and the load capacitors must be placed as close as possible to the oscillator pins to minimize output distortion and startup stabilization time. The loading capacitance values must be adjusted according to the selected oscillator.

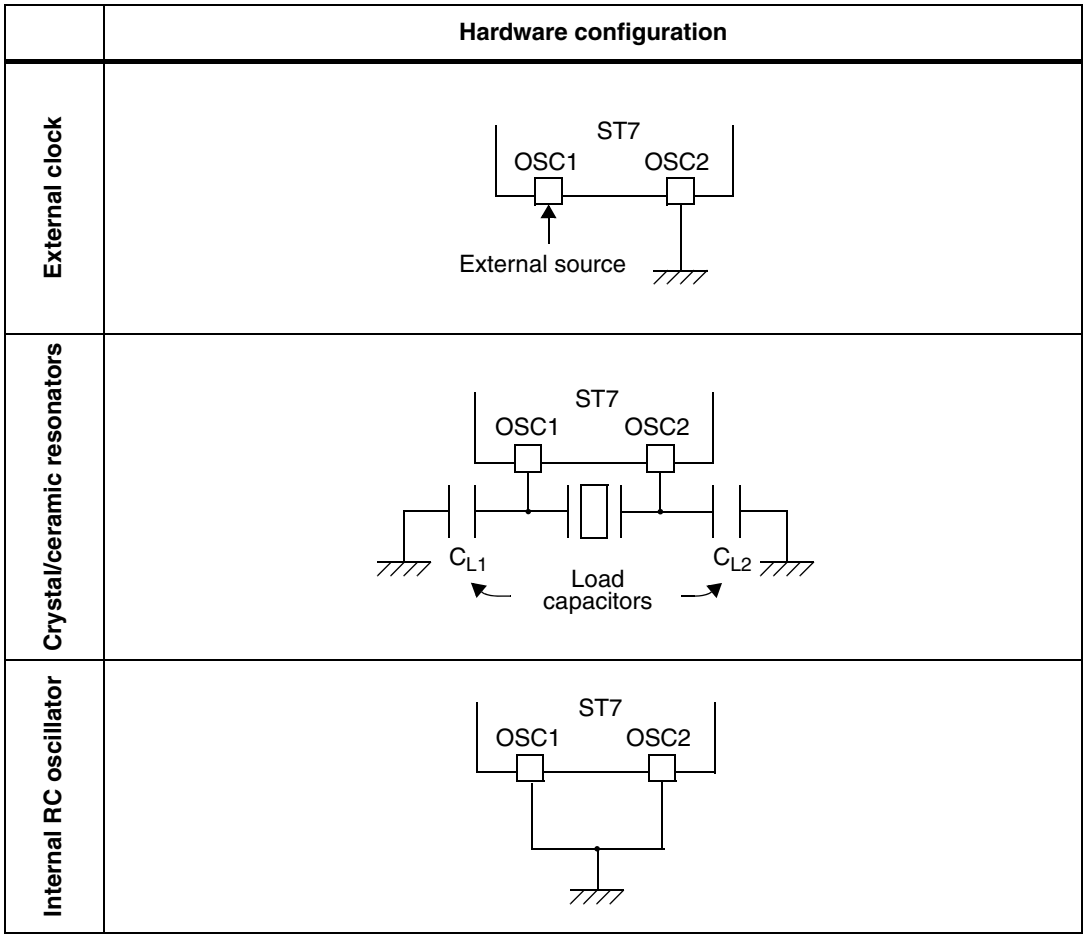
These oscillators are not stopped during the reset phase to avoid losing time in the oscillator startup phase.

7.4.3 Internal RC oscillator

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

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

Figure 14. ST7 clock sources



7.5 Reset sequence manager (RSM)

7.5.1 Introduction

The reset sequence manager includes three reset sources as shown in *Figure 16: Reset block diagram on page 44*:

- 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 12.2.2: Illegal opcode reset on page 175 for further details.

These sources act on the $\overline{\text{RESET}}$ pin which 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 three phases as shown in *Figure 15*:

- Active phase depending on the reset source
- 256 or 4096 CPU clock cycle delay (see *Table 10*)
- 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 $\overline{\text{RESET}}$ pin in low state until programming mode is entered, in order to avoid unwanted behavior.

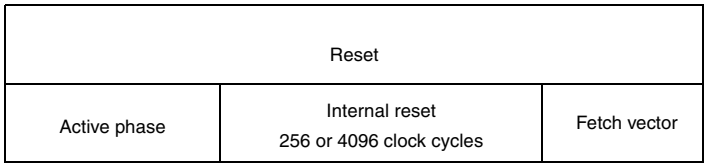
The 256 or 4096 CPU clock cycle delay allows the oscillator to stabilize and ensures that recovery has taken place from the reset state. The shorter or longer clock cycle delay is automatically selected depending on the clock source chosen by option byte: The reset vector fetch phase duration is two clock cycles.

Table 10. Clock cycle delays

Clock source	CPU clock cycle delay
Internal RC oscillator	256
External clock (connected to CLKIN pin)	
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: PLL output frequency timing diagram on page 38*).

Figure 15. Reset sequence phases

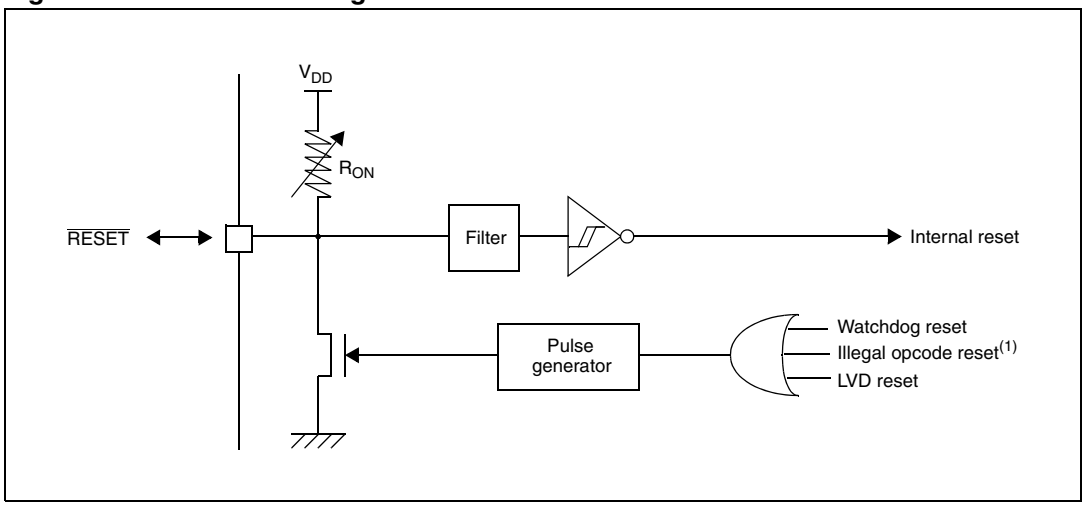


7.5.2 Asynchronous external $\overline{\text{RESET}}$ pin

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 *Section 13: Electrical characteristics* 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: Reset sequences on page 45*). This detection is asynchronous and therefore the MCU can enter the reset state even in halt mode.

Figure 16. Reset block diagram



1. See *Section 12.2.2: Illegal opcode reset on page 175* for more details on illegal opcode reset conditions

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 *Section 13: Electrical characteristics*.

7.5.3 External power-on reset

If the LVD is disabled by the option byte, to start up the microcontroller correctly, the user must use an external reset circuit to ensure 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.

7.5.4 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: Reset sequences on page 45*.

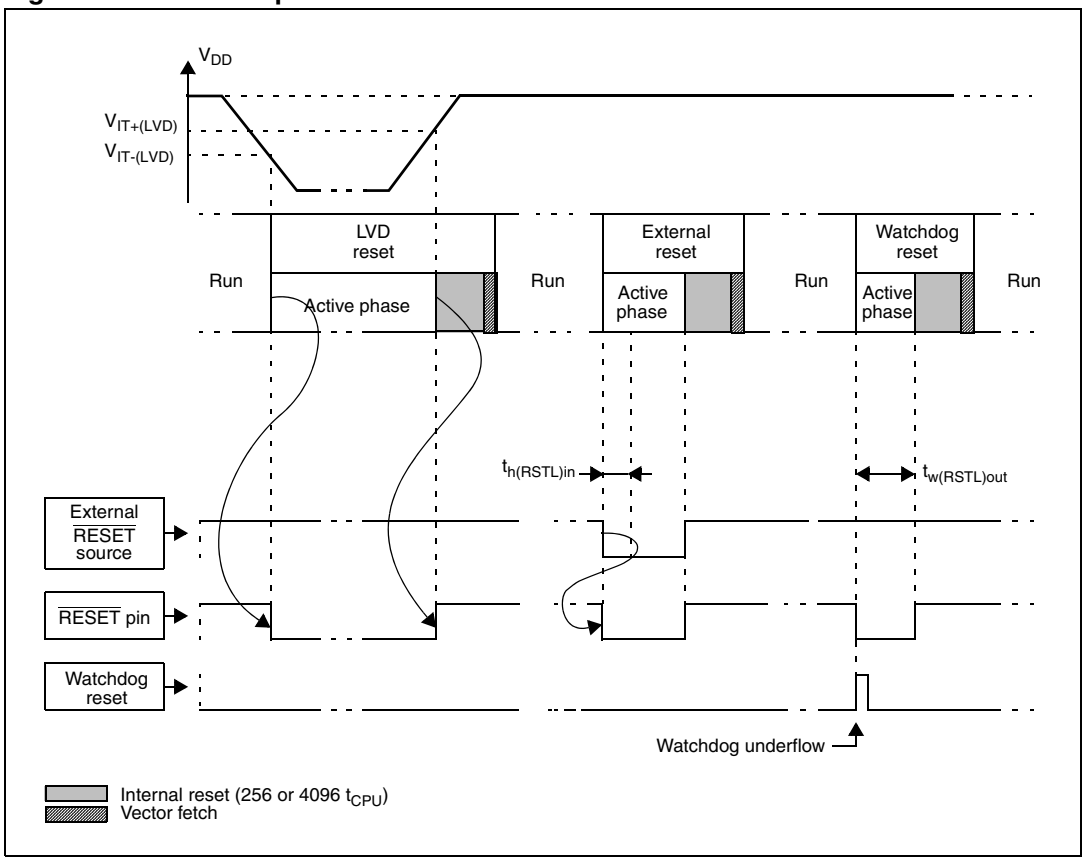
The LVD filter spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets.

7.5.5 Internal watchdog reset

The reset sequence generated by an internal Watchdog counter overflow is shown in *Figure 17: Reset sequences on page 45*.

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



7.6 System integrity management (SI)

The system integrity management block contains the low voltage detector (LVD) and auxiliary voltage detector (AVD) functions. It is managed by the SICSR register.

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to Section 12.2.2: Illegal opcode reset on page 175 for further details.

7.6.1 Low voltage detector (LVD)

The low voltage detector (LVD) function generates a static reset when the V_{DD} supply voltage is below a $V_{IT-(LVD)}$ reference value. This means that it secures the power-up as well as the power-down, keeping the ST7 in reset.

The $V_{IT-(LVD)}$ reference value for a voltage drop is lower than the $V_{IT+(LVD)}$ reference value for power-on to avoid a parasitic reset when the MCU starts running and sinks current on the supply (hysteresis).

The LVD reset circuitry generates a reset when V_{DD} is below:

- $V_{IT+(LVD)}$ when V_{DD} is rising
- $V_{IT-(LVD)}$ when V_{DD} is falling

The LVD function is illustrated in *Figure 18: Low voltage detector vs. reset on page 46*.

The LVD can be enabled by option byte with highest voltage threshold.

Provided the minimum V_{DD} value (guaranteed for the oscillator frequency) is above $V_{IT-(LVD)}$, the MCU can only be in two modes:

- Under full software control
- In static safe reset

In these conditions, secure operation is always ensured for the application without the need for external reset hardware.

During a low voltage detector reset, the $\overline{\text{RESET}}$ pin is held low, thus permitting the MCU to reset other devices.

Note: 1 The LVD allows the device to be used without any external reset circuitry.

2 The LVD is an optional function which can be selected by the option byte.

3 Use of LVD with capacitive power supply: With this type of power supply, if power cuts occur in the application, it is recommended to pull V_{DD} down to 0 V to ensure optimum restart conditions. Refer to the circuit example in Figure 98: RESET pin protection when LVD Is enabled on page 206 and Note on the same page.

4 For the application to function correctly, it is recommended to make sure that the V_{DD} supply voltage rises monotonously when the device is exiting from reset, to ensure the application functions properly.

Figure 18. Low voltage detector vs. reset

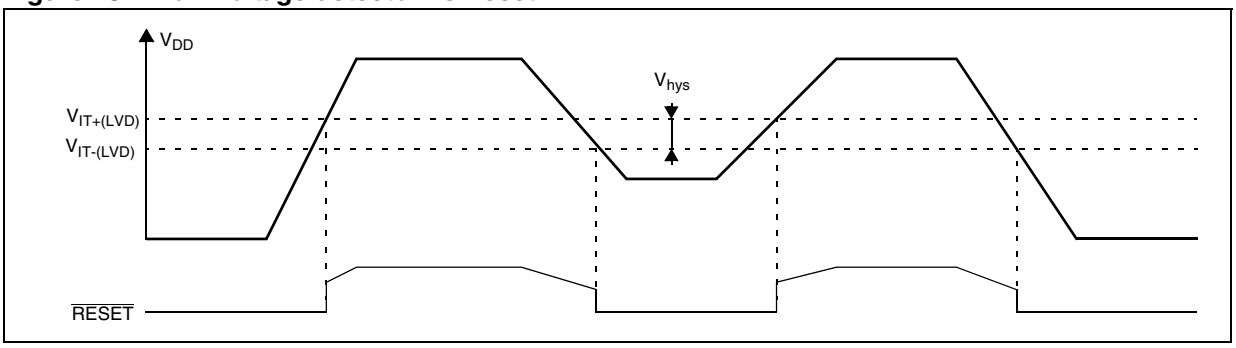
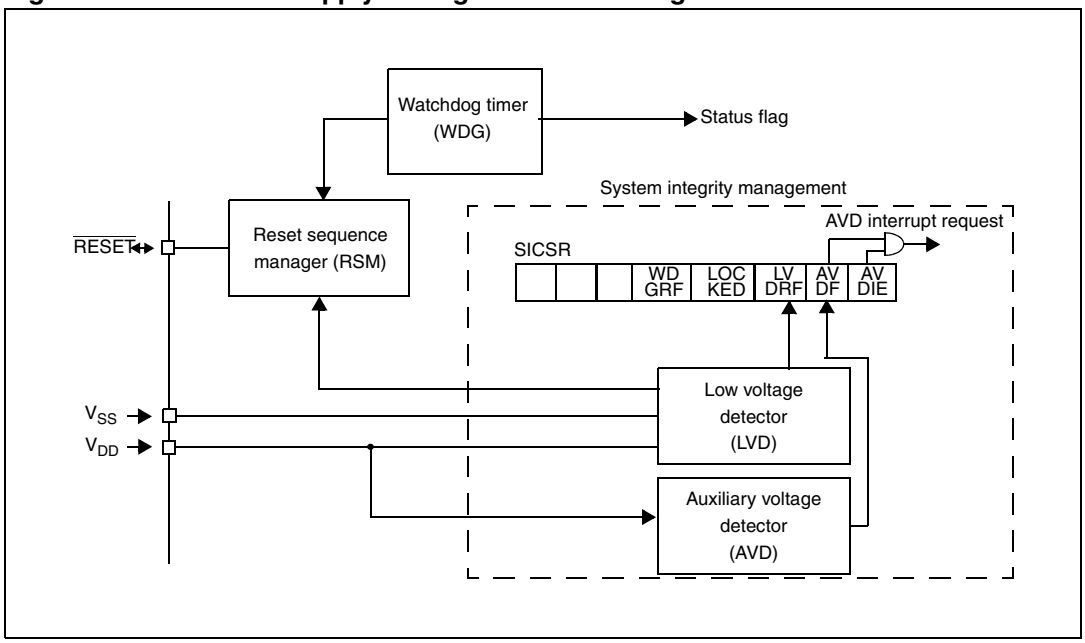


Figure 19. Reset and supply management block diagram



7.6.2 Low power modes

Table 11. Effect of low power modes on system integrity

Mode	Description
Wait	No effect on SI. AVD interrupts cause the device to exit from wait mode.
Halt	The SICSR register is frozen. The AVD becomes inactive and the AVD interrupt cannot be used to exit from halt mode.

7.6.3 Interrupts

The AVD interrupt event generates an interrupt if the corresponding enable control bit (AVDIE) is set and the interrupt mask in the CC register is reset (RIM instruction).

Table 12. Supply, reset and clock management interrupt control/wake-up capability

Interrupt event	Event flag	Enable control bit	Exit from wait	Exit from halt
AVD event	AVDF	AVDIE	Yes	No

Auxiliary voltage detector (AVD)

The voltage detector function (AVD) is based on an analog comparison between a $V_{IT-(AVD)}$ and $V_{IT+(AVD)}$ reference value and the V_{DD} main supply voltage (V_{AVD}). The $V_{IT-(AVD)}$ reference value for falling voltage is lower than the $V_{IT+(AVD)}$ reference value for rising voltage in order to avoid parasitic detection (hysteresis).

The output of the AVD comparator is directly readable by the application software through a real time status bit (AVDF) in the SICSR register. This bit is read only.

Caution: The AVD functions only if the LVD is enabled through the option byte.

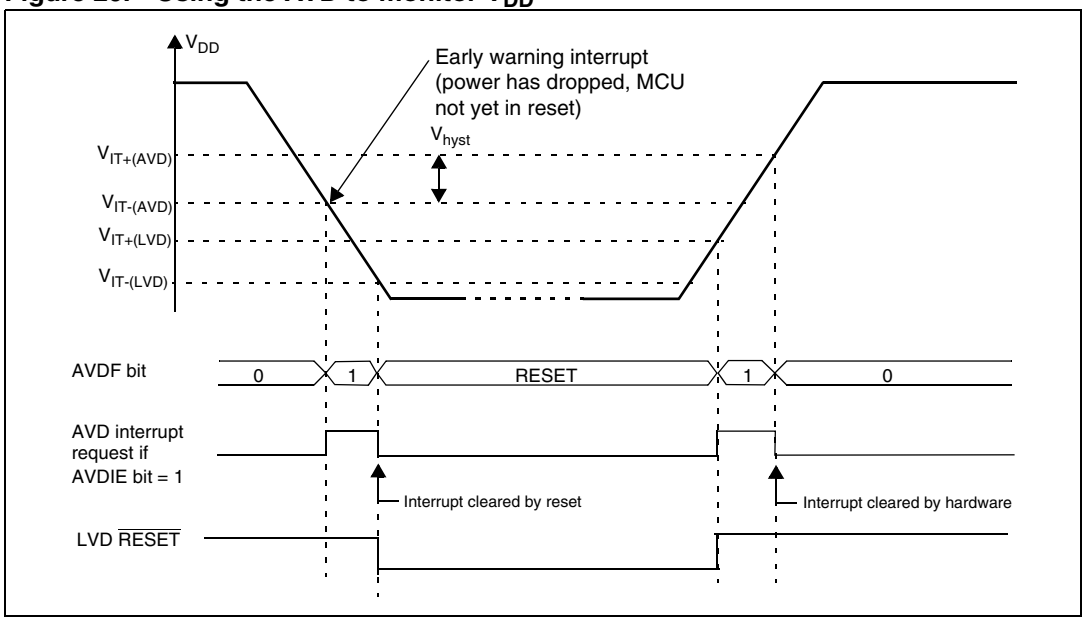
Monitoring the V_{DD} main supply

The AVD voltage threshold value is relative to the selected LVD threshold configured by option byte (see *Section 15.2 on page 215*).

If the AVD interrupt is enabled, an interrupt is generated when the voltage crosses the $V_{IT+(LVD)}$ or $V_{IT-(AVD)}$ threshold (AVDF bit is set).

In the case of a drop in voltage, the AVD interrupt acts as an early warning, allowing software to shut down safely before the LVD resets the microcontroller. See *Figure 20*.

Figure 20. Using the AVD to monitor V_{DD}



7.6.4 Register description

System integrity (SI) control/status register (SICSR)

SICSR Reset value: 0110 0xx0 (6xh)

7	6	5	4	3	2	1	0
Reserved	CR[1:0]		WDGRF	LOCKED	LVDRF	AVDF	AVDIE
-	R/W		R/W	R/W	R/W	R/W	R/W

Table 13. SICSR register description

Bit	Bit name	Function
7	-	Reserved, must be kept cleared
6:5	CR[1:0]	RC oscillator frequency adjustment bits These bits, as well as CR[9:2] bits in the RCCR register must be written immediately after reset to adjust the RC oscillator frequency and to obtain an accuracy of 1%. Refer to *Section 7.3: Register description on page 39*

Table 13. SICSR register description (continued)

Bit	Bit name	Function
4	WDGRF	Watchdog reset flag This bit indicates that the last reset was generated by the watchdog peripheral. It is set by hardware (watchdog reset) and cleared by software (by reading SICSR register) or an LVD reset (to ensure a stable cleared state of the WDGRF flag when CPU starts). Combined with the LVDRF flag information, the flag description is given as follows: 00 (LVDRF, WDGRF): Reset sources = External $\overline{\text{RESET}}$ pin 01 (LVDRF, WDGRF): Reset sources = Watchdog 1X (LVDRF, WDGRF): Reset sources = LVD
3	LOCKED	PLL locked flag This bit is set and cleared by hardware. It is set automatically when the PLL reaches its operating frequency. 0: PLL not locked 1: PLL locked
2	LVDRF	LVD reset flag This bit indicates that the last reset was generated by the LVD block. It is set by hardware (LVD reset) and cleared by software (by reading). When the LVD is disabled by option byte, the LVDRF bit value is undefined. *Note: The LVDRF flag is not cleared when another reset type occurs (external or watchdog), the LVDRF flag remains set to keep trace of the original failure. In this case, a watchdog reset can be detected by software while an external reset can not.*
1	AVDF	Voltage detector flag This read-only bit is set and cleared by hardware. If the AVDIE bit is set, an interrupt request is generated when the AVDF bit is set. Refer to *Figure 20* and to *Monitoring the VDD main supply on page 48* for additional details. 0: V_{DD} over AVD threshold 1: V_{DD} under AVD threshold
0	AVDIE	Voltage detector interrupt enable This bit is set and cleared by software. It enables an interrupt to be generated when the AVDF flag is set. The pending interrupt information is automatically cleared when software enters the AVD interrupt routine. 0: AVD interrupt disabled 1: AVD interrupt enabled

8 Interrupts

The ST7 core may be interrupted by one of two different methods: Maskable hardware interrupts as listed in *Table 14: Interrupt mapping on page 53* and a non-maskable software interrupt (TRAP). The interrupt processing flowchart is shown in *Figure 21: Interrupt processing flowchart on page 52*.

The maskable interrupts must be enabled by clearing the I bit in order to be serviced. However, disabled interrupts may be latched and processed when they are enabled (see external interrupts subsection).

Note: After reset, all interrupts are disabled.

When an interrupt has to be serviced:

- Normal processing is suspended at the end of the current instruction execution.
- The PC, X, A and CC registers are saved onto the stack.
- The I bit of the CC register is set to prevent additional interrupts.
- The PC is then loaded with the interrupt vector of the interrupt to service and the first instruction of the interrupt service routine is fetched (refer to *Table 14: Interrupt mapping* for vector addresses).

The interrupt service routine should finish with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

Note: As a consequence of the IRET instruction, the I bit is cleared and the main program resumes.

Priority management

By default, a servicing interrupt cannot be interrupted because the I bit is set by hardware entering in interrupt routine.

In the case when several interrupts are simultaneously pending, an hardware priority defines which one will be serviced first (see *Table 14: Interrupt mapping*).

Interrupts and low power mode

All interrupts allow the processor to leave the wait low power mode. Only external and specifically mentioned interrupts allow the processor to leave the halt low power mode (refer to the ‘Exit from halt’ column in *Table 14: Interrupt mapping*).

8.1 Non maskable software interrupt

This interrupt is entered when the TRAP instruction is executed regardless of the state of the I bit. It is serviced according to the flowchart in *Figure 21: Interrupt processing flowchart on page 52*.

8.2 External interrupts

External interrupt vectors can be loaded into the PC register if the corresponding external interrupt occurred and if the I bit is cleared. These interrupts allow the processor to leave the halt low power mode.

The external interrupt polarity is selected through the miscellaneous register or interrupt register (if available).

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

Caution: The type of sensitivity defined in the miscellaneous or interrupt register (if available) applies to the *ei* source. In case of a NANDed source (as described in *Section 10: I/O ports*), a low level on an I/O pin, configured as input with interrupt, masks the interrupt request even in case of rising-edge sensitivity.

8.3 Peripheral interrupts

Different peripheral interrupt flags in the status register are able to cause an interrupt when they are active if both the following conditions are met:

- The I bit of the CC register is cleared
- The corresponding enable bit is set in the control register

If either of these two conditions is false, the interrupt is latched and thus remains pending.

Clearing an interrupt request is done by:

- Writing ‘0’ to the corresponding bit in the status register or
- Access to the status register while the flag is set followed by a read or write of an associated register.

Note: The clearing sequence resets the internal latch. A pending interrupt (that is, waiting for being enabled) will therefore be lost if the clear sequence is executed.

Figure 21. Interrupt processing flowchart

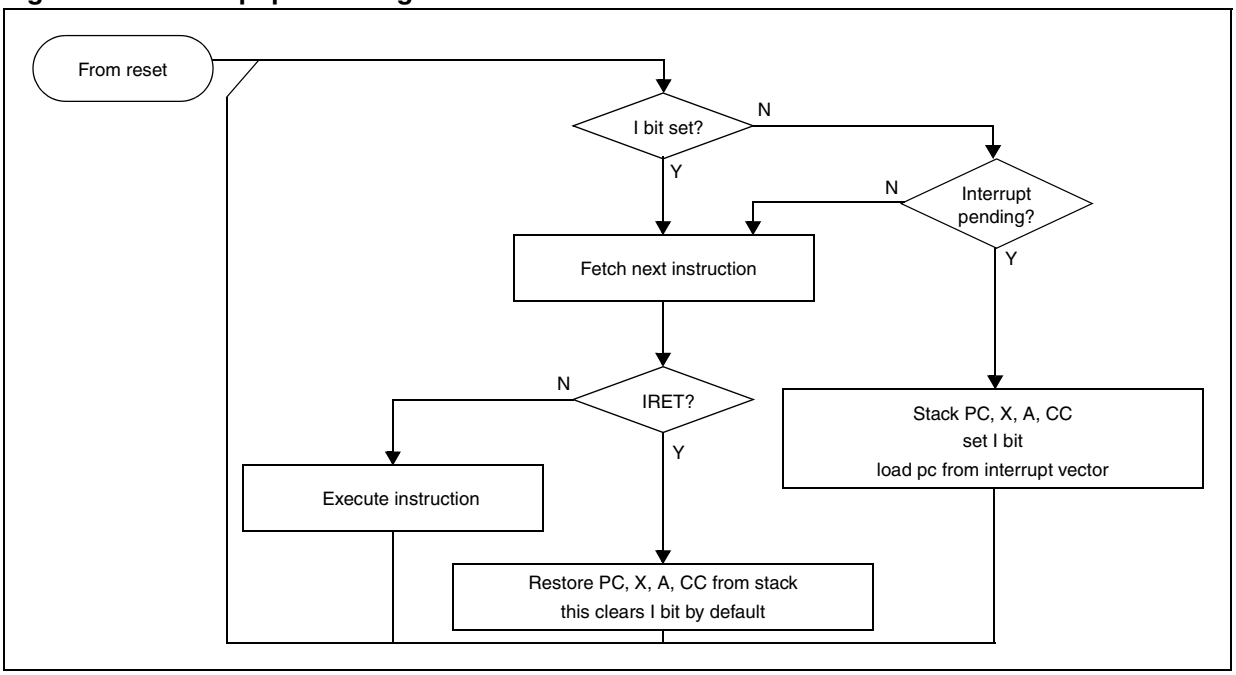


Table 14. Interrupt mapping

No.	Source block	Description	Register label	Priority order	Exit from halt or AWUFH	Address vector
	Reset	Reset	-	Highest priority	Yes	FFFEh-FFFFh
	TRAP	Software interrupt			No	FFFCh-FFFDh
0	AWU	Auto wakeup interrupt	AWUCSR		Yes(1)	FFFAh-FFFBh
1	ei0	External interrupt 0	-		Yes	FFF8h-FFF9h
2	ei1	External interrupt 1				FFF6h-FFF7h
3	ei2	External interrupt 2				FFF4h-FFF5h
4	ei3	External interrupt 3				FFF2h-FFF3h
5	Lite timer	Lite timer RTC2 interrupt	LTCSR2		No	FFF0h-FFF1h
6	LINSCI	LINSCI interrupt	SCICR1/ SCICR2		No	FFEEh-FFEFh
7	SI	AVD interrupt	SICSR		No	FFECh-FFEDh
8	At timer	At timer output compare Interrupt or input capture interrupt	PWMxCSR or ATCSR		No	FFEAh-FFEBh
9		At timer overflow interrupt	ATCSR		Yes(2)	FFE8h-FFE9h
10	Lite timer	Lite timer input capture interrupt	LTCSR		No	FFE6h-FFE7h
11		Lite timer RTC1 interrupt	LTCSR		Yes(2)	FFE4h-FFE5h
12	SPI	SPI peripheral interrupts	SPICSR		Yes	FFE2h-FFE3h
13	At timer	At timer overflow interrupt 2	ATCSR2	lowest priority	No	FFE0h-FFE1h

1. This interrupt exits the MCU from ‘auto wakeup from halt’ mode only
2. These interrupts exit the MCU from ‘active halt’ mode only

External interrupt control register (EICR)

EICR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
IS3[1:0]		IS2[1:0]		IS1[1:0]		IS0[1:0]	
R/W		R/W		R/W		R/W	

Table 15. EICR register description

Bit	Bit name	Function
7:6	IS3[1:0]	ei3 sensitivity These bits define the interrupt sensitivity for ei3 (port B0) according to *Table 16*
5:4	IS2[1:0]	ei2 sensitivity These bits define the interrupt sensitivity for ei2 (port B3) according to *Table 16*
3:2	IS1[1:0]	ei1 sensitivity These bits define the interrupt sensitivity for ei1 (port A7) according to *Table 16*
1:0	IS0[1:0]	ei0 sensitivity These bits define the interrupt sensitivity for ei0 (port A0) according to *Table 16*

Note: 1 These 8 bits can be written only when the I bit in the CC register is set.

Table 16. Interrupt sensitivity bits

ISx1	ISx0	External interrupt sensitivity
0	0	Falling edge and low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

External interrupt selection register (EISR)

EISR

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ei3[1:0]		ei2[1:0]		ei1[1:0]		ei0[1:0]	
R/W		R/W		R/W		R/W	

Table 17. EISR register description

Bit	Bit name	Function
7:6	ei3[1:0]	ei3 pin selection These bits are written by software. They select the port B I/O pin used for the ei3 external interrupt as follows: 00: I/O pin = No interrupt (reset state) 01: I/O pin = PB0 10: I/O pin = PB1 11: I/O pin = PB2
5:4	ei2[1:0]	ei2 pin selection These bits are written by software. They select the port B I/O pin used for the ei2 external interrupt as follows: 00: I/O pin = No interrupt (reset state) 01: I/O pin = PB3 10: I/O pin = PB5 11: I/O pin = PB6
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 as follows: 00: I/O pin = No interrupt (reset state) 01: I/O pin = PA4 10: I/O pin = PA5 11: I/O pin = PA6
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 as follows: 00: I/O pin = No interrupt (reset state)PA0 (reset state) 01: I/O pin = PA1 10: I/O pin = PA2 11: I/O pin = PA3

9 Power saving modes

9.1 Introduction

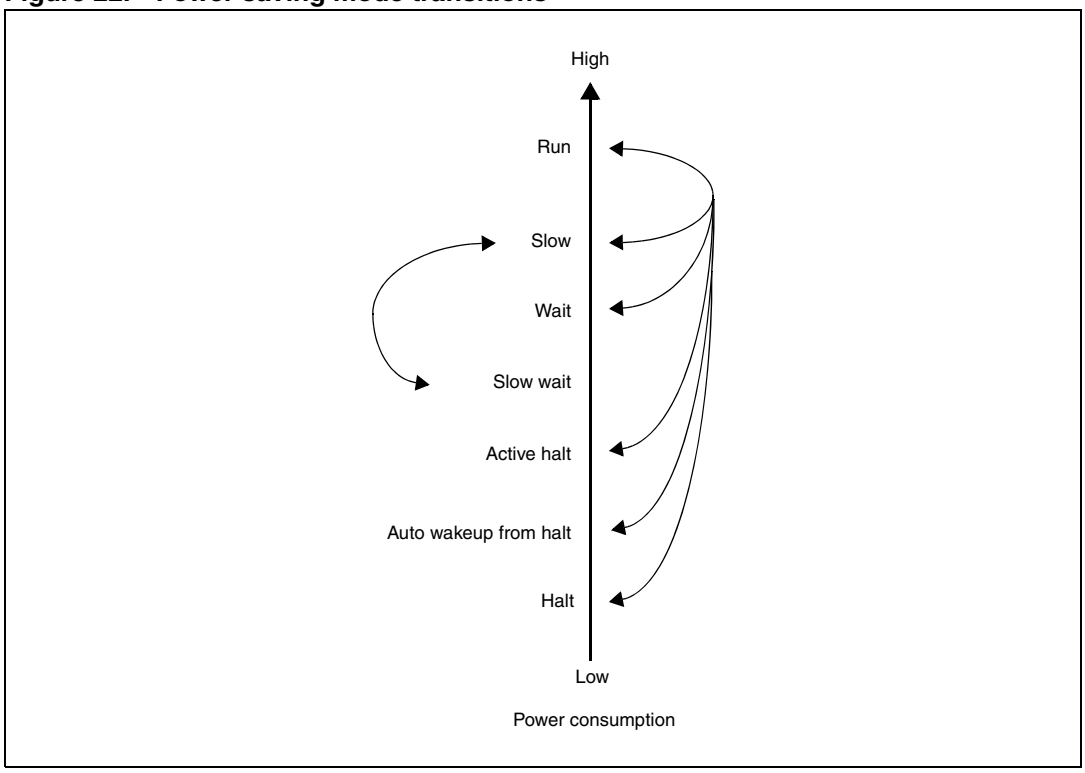
To give a large measure of flexibility to the application in terms of power consumption, five main power saving modes are implemented in the ST7 (see *Figure 22*):

- Slow
- Wait (and Slow-Wait)
- Active halt
- Auto wakeup from halt (AWUFH)
- Halt

After a reset, the normal operating mode is selected by default (run mode). This mode drives the device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency divided or multiplied by 2 (f_{OSC2}).

From run mode, the different power saving modes can be selected by setting the relevant register bits or by calling the specific ST7 software instruction whose action depends on the oscillator status.

Figure 22. Power saving mode transitions



9.2 Slow mode

This mode has two targets:

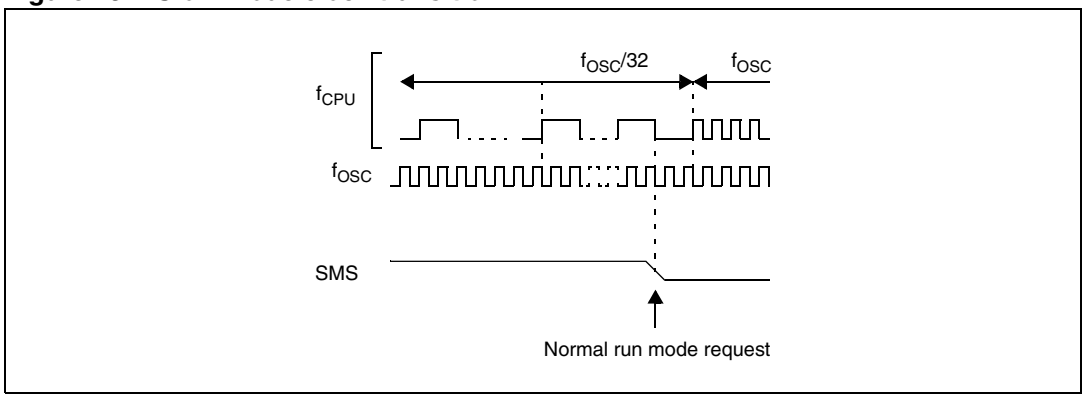
- To reduce power consumption by decreasing the internal clock in the device
- To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

Slow mode is controlled by the SMS bit in the MCCSR register which enables or disables slow mode.

In this mode, the oscillator frequency is divided by 32. The CPU and peripherals are clocked at this lower frequency.

Note: Slow-wait mode is activated when entering wait mode while the device is already in slow mode.

Figure 23. Slow mode clock transition



9.3 Wait mode

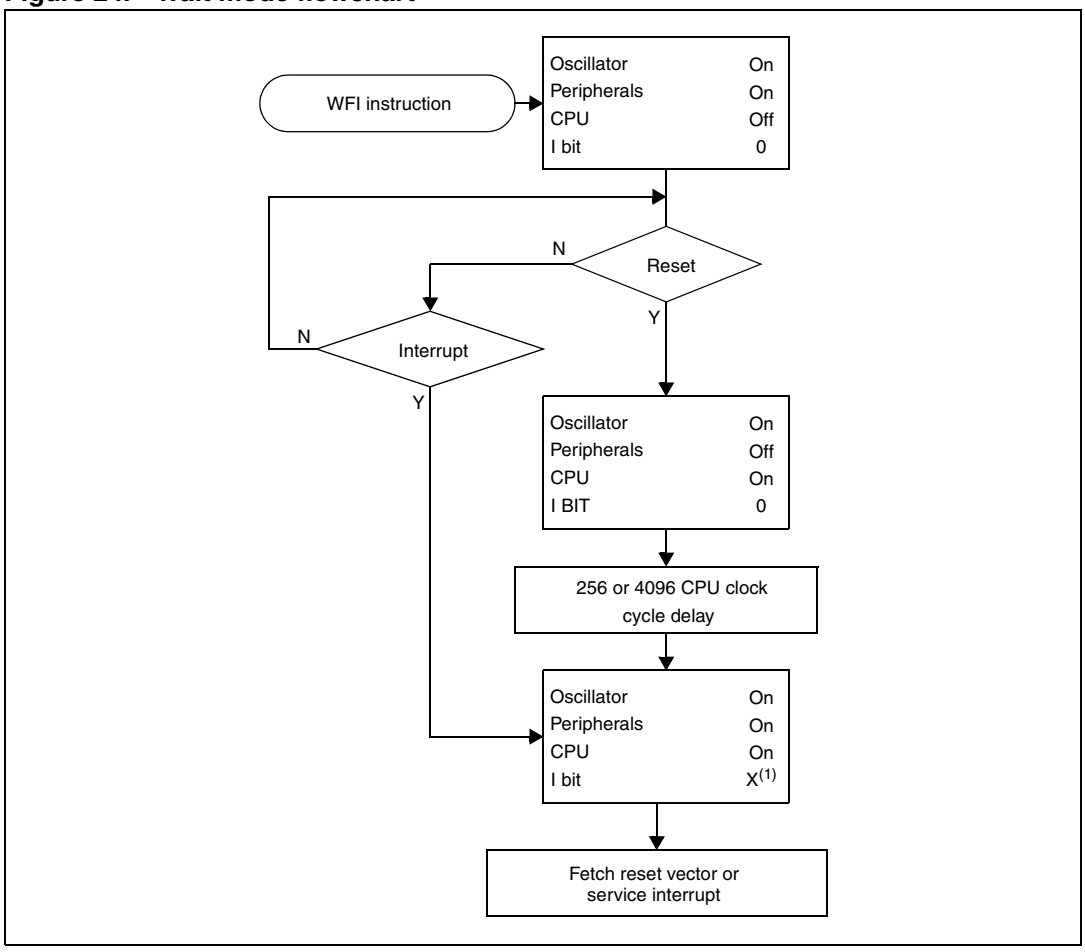
Wait mode places the MCU in a low power consumption mode by stopping the CPU.

This power saving mode is selected by calling the 'WFI' instruction.

All peripherals remain active. During wait mode, the I bit of the CC register is cleared to enable all interrupts. All other registers and memory remain unchanged. The MCU remains in wait mode until a reset or an interrupt occurs, causing it to wake up. Then the program counter branches to the starting address of the interrupt or reset service routine.

Refer to *Figure 24: Wait mode flowchart*.

Figure 24. Wait mode flowchart



1. Before servicing an interrupt, the CC register is pushed on the stack. The I bit of the CC register is set during the interrupt routine and cleared when the CC register is popped.

9.4 Halt mode

The halt mode is the lowest power consumption mode of the MCU. It is entered by executing the 'HALT' instruction when active halt is disabled (see *Section 9.5: Active halt mode on page 61* for more details) and when the AWUEN bit in the AWUCSR register is cleared.

The MCU can exit halt mode on reception of either a specific interrupt (see *Table 14: Interrupt mapping on page 53*) or a reset. When exiting halt mode by means of a reset or an interrupt, the oscillator is immediately turned on and the 256 or 4096 CPU cycle delay is used to stabilize the oscillator. After the startup delay, the CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see *Figure 26: Halt mode flowchart on page 60*).

When entering halt 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 halt mode, the main oscillator is turned off, stopping all internal processing, including the operation of the on-chip peripherals. All peripherals are not clocked except those which receive their clock supply from another clock generator (such as an external or auxiliary oscillator).

The compatibility of watchdog operation with halt mode is configured by the 'WDGHALT' option bit of the option byte. The HALT instruction, when executed while the watchdog system is enabled, can generate a watchdog reset (see *Section 15.2: Option bytes on page 215* for more details).

Figure 25. Halt timing overview

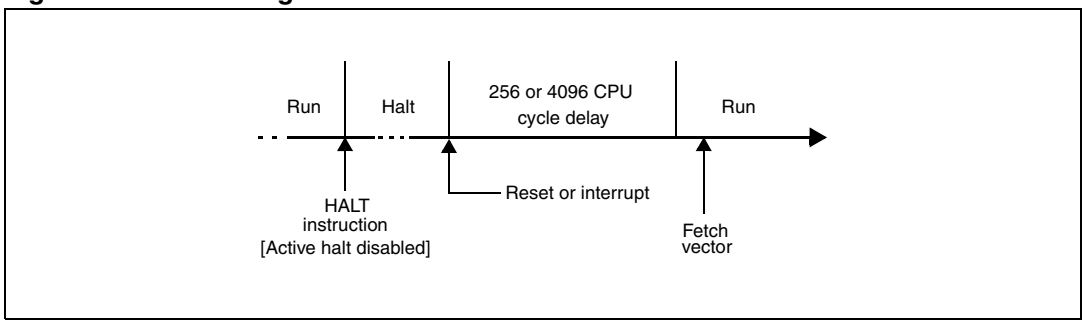
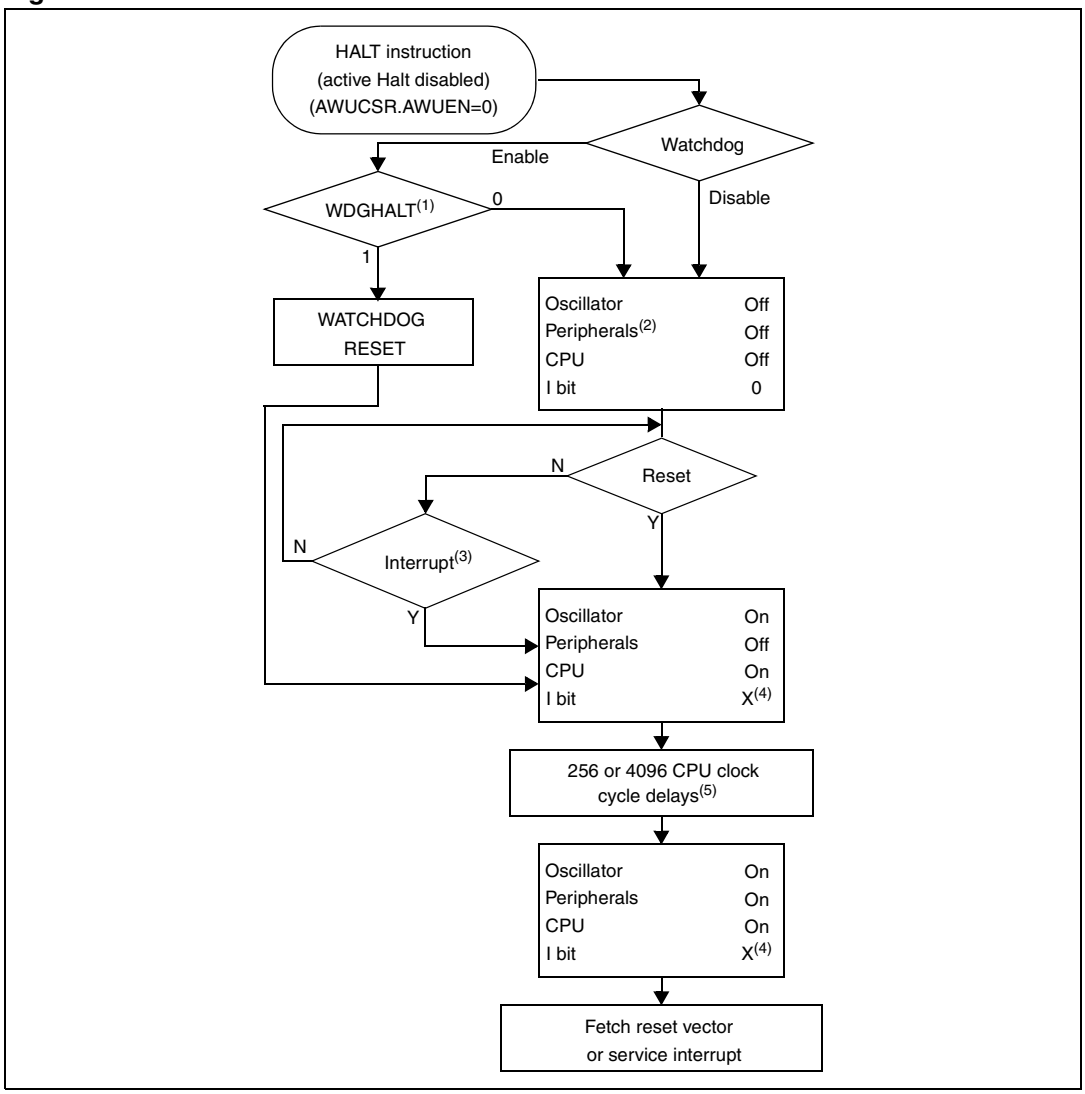


Figure 26. Halt mode flowchart



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 some specific interrupts can exit the MCU from HALT mode (such as external interrupt). Refer to *Table 14: Interrupt mapping on page 53* for more details.
4. Before servicing an interrupt, the CC register is pushed on the stack. The I bit of the CC register is set during the interrupt routine and cleared when the CC register is popped.
5. If the PLL is enabled by option byte, it outputs the clock after a delay of $t_{STARTUP}$ (see *Figure 12 on page 38*).

9.4.1 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, re-initialize the corresponding I/O as 'input pull-up with interrupt' or 'floating interrupt' before executing the HALT instruction. The main reason for this is that the I/O may be incorrectly configured due to external interference or by an unforeseen logical condition.
- For the same reason, re-initialize 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 program memory with the value 0x8E.
- As the HALT instruction clears the interrupt mask 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 wakeup event (reset or external interrupt).

9.5 Active halt mode

Active halt mode is the lowest power consumption mode of the MCU with a real-time clock (RTC) available. It is entered by executing the 'HALT' instruction. The decision to enter either in active halt or halt mode is given by the LTCSR/ATCSR register status as shown in the following table:

Table 18. LTCSR/ATCSR register status

LTCSR1 TB1IE bit	ATCSR OVFIE bit	ATCSRCK1 bit	ATCSRCK0 bit	Meaning
0	x	x	0	Active halt mode disabled
0	0	x	x	
1	x	x	x	Active halt mode enabled
x	1	0	1	

The MCU exits in active halt mode on reception of a specific interrupt (see *Table 14: Interrupt mapping on page 53*) or a reset.

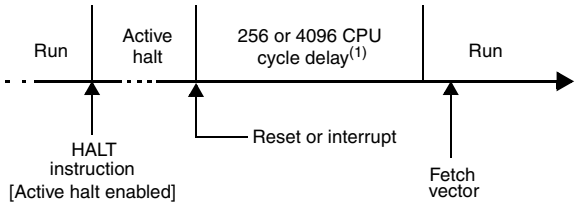
- When exiting active halt mode by means of a reset, a 256 CPU cycle delay occurs. After the startup delay, the CPU resumes operation by fetching the reset vector which woke it up (see *Figure 28: Active halt mode flowchart on page 62*).
- When exiting active halt mode by means of an interrupt, the CPU immediately resumes operation by servicing the interrupt vector which woke it up (see *Figure 28: Active halt mode flowchart on page 62*).

When entering active halt mode, the I bit in the CC register is cleared to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately (see *Figure 28*, *Note 2*).

In active halt mode, only the main oscillator and the selected timer counter (LT/AT) are running to keep a wakeup time base. All other peripherals are not clocked except those which receive their clock supply from another clock generator (such as external or auxiliary oscillator).

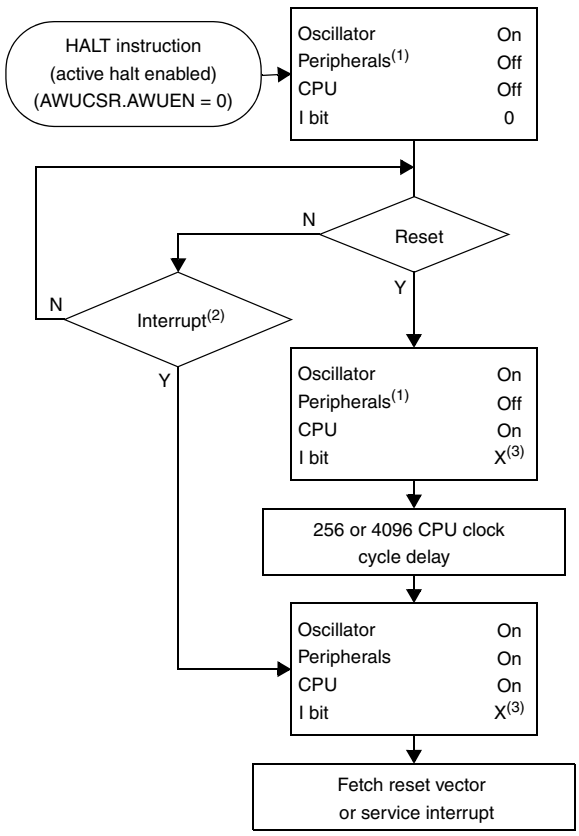
Note: As soon as active halt is enabled, executing a HALT instruction while the watchdog is active does not generate a reset. This means that the device cannot exceed a defined delay in this power saving mode.

Figure 27. Active halt timing overview



1. This delay occurs only if the MCU exits active halt mode by means of a reset.

Figure 28. Active halt mode flowchart



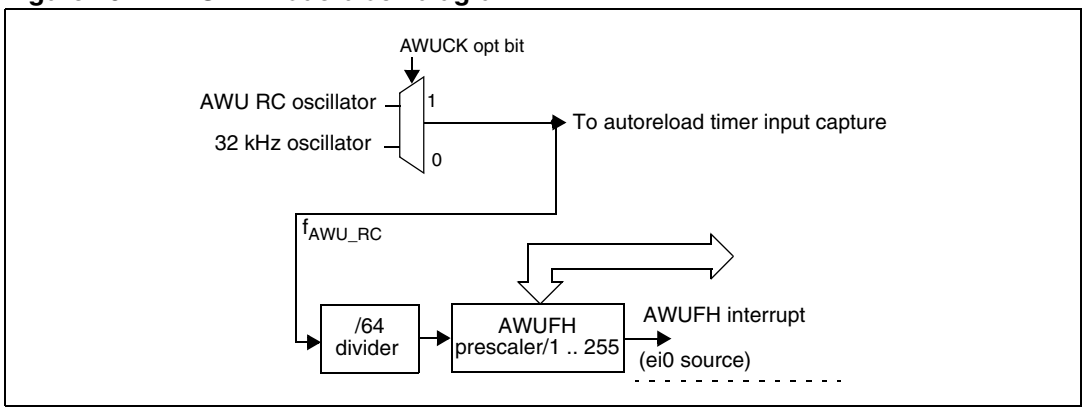
1. Peripherals clocked with an external clock source can still be active.
2. Only the RTC1 interrupt and some specific interrupts can exit the MCU from active halt mode. Refer to *Table 14: Interrupt mapping* for more details.
3. Before servicing an interrupt, the CC register is pushed on the stack. The I bit of the CC register is set during the interrupt routine and cleared when the CC register is popped.

9.6 Auto wakeup from halt mode

Auto wakeup from halt (AWUFH) mode is similar to halt mode with the addition of a specific internal RC oscillator for wakeup (auto wakeup from halt oscillator). Compared to active halt mode, AWUFH has lower power consumption (the main clock is not kept running but there is no accurate real-time clock available).

It is entered by executing the HALT instruction when the AWUEN bit in the AWUCSR register has been set.

Figure 29. AWUFH mode block diagram



As soon as halt mode is entered and if the AWUEN bit has been set in the AWUCSR register, the AWU RC oscillator provides a clock signal (f_{AWU_RC}). Its frequency is divided by a fixed divider and a programmable prescaler controlled by the AWUPR register. The output of this prescaler provides the delay time. When the delay has elapsed, the AWUF flag is set by hardware and an interrupt wakes up the MCU from halt mode. At the same time, the main oscillator is immediately turned on and a 256-cycle delay is used to stabilize it. After this startup delay, the CPU resumes operation by servicing the AWUFH interrupt. The AWU flag and its associated interrupt are cleared by software reading the AWUCSR register.

To compensate for any frequency dispersion of the AWU RC oscillator, it can be calibrated by measuring the clock frequency f_{AWU_RC} and then calculating the right prescaler value. Measurement mode is enabled by setting the AWUM bit in the AWUCSR register in run mode. This connects f_{AWU_RC} to the input capture of the 12-bit autoreload timer, allowing the f_{AWU_RC} to be measured using the main oscillator clock as a reference timebase.

Similarities with halt mode

The following AWUFH mode behavior 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 on page 59*).
- 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, stopping all internal processing, including the operation of the on-chip peripherals. None of the peripherals are clocked except those which receive 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 30. AWUF halt timing diagram

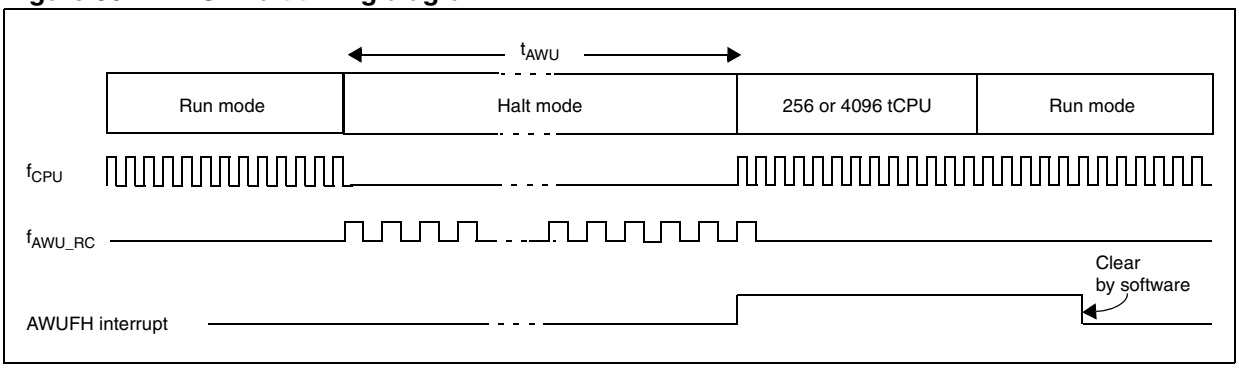
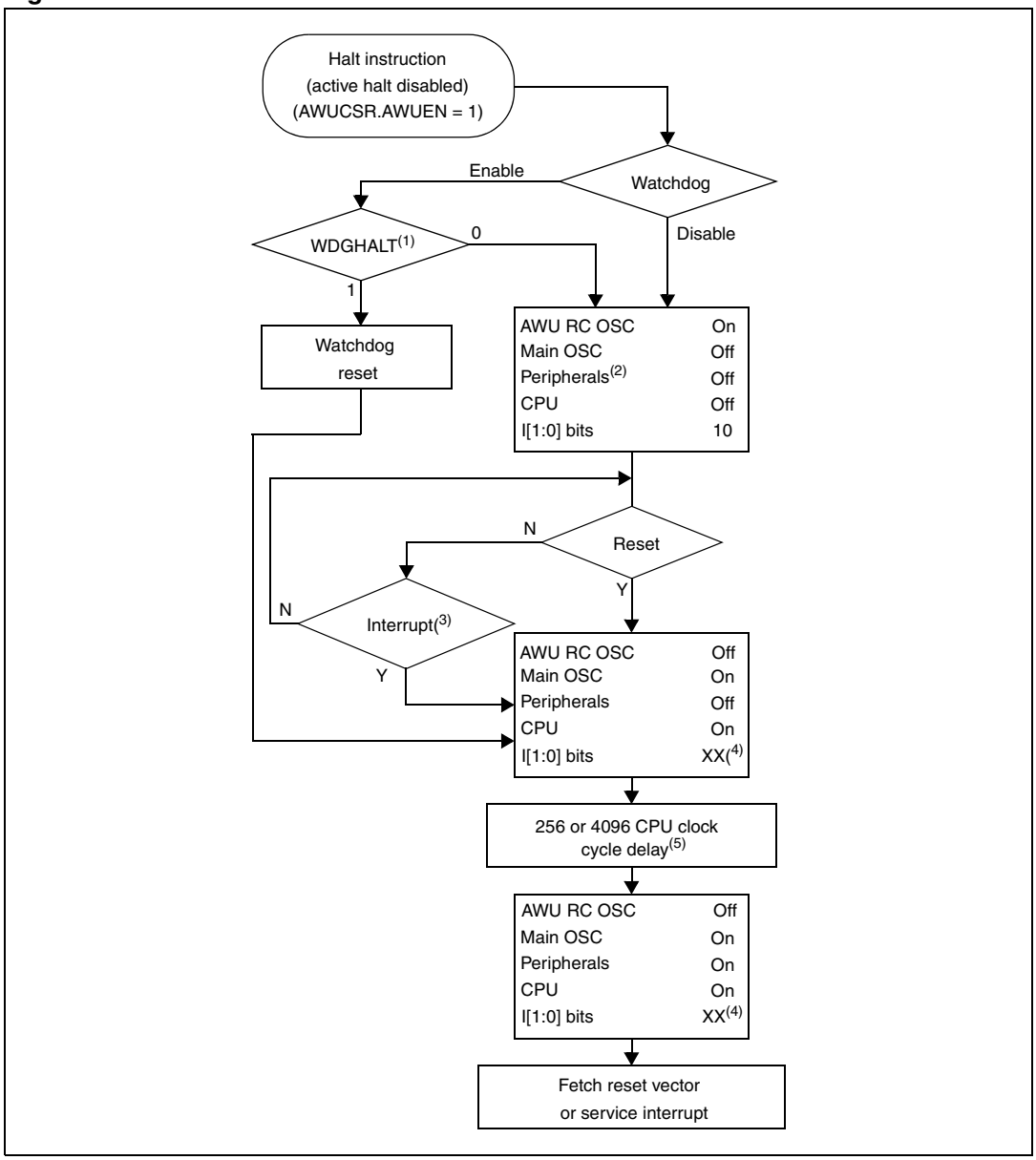


Figure 31. AWUFH mode flowchart



1. WDGHALT is an option bit. See *Section 15.2: Option bytes on page 215* 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 14: Interrupt mapping on page 53* 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.
5. If the PLL is enabled by the option byte, it outputs the clock after an additional delay of $t_{STARTUP}$ (see *Figure 12: PLL output frequency timing diagram on page 38*).

Register description

AWUFH control/status register (AWUCSR)

AWUCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	AWUF	AWUM	AWUEN
-	-	-	-	-	R/W	R/W	R/W

Table 19. AWUCSR register description

Bit	Bit name	Function
7:3	-	Reserved, must be kept cleared
2	AWUF	Auto wakeup flag This bit is set by hardware when the AWU module generates an interrupt and cleared by software on reading AWUCSR. Writing to this bit does not change its value. 0: No AWU interrupt occurred 1: AWU interrupt occurred
1	AWUM	Auto wakeup measurement This bit enables the AWU RC oscillator and connects its output to the input capture of the 12-bit autoreload timer. This allows the timer to measure the AWU RC oscillator dispersion and then compensate this dispersion by providing the right value in the AWUPR register. 0: Measurement disabled 1: Measurement enabled
0	AWUEN	Auto wakeup from halt enabled This bit enables the auto wakeup from halt feature: Once halt mode is entered, the AWUFH wakes up the microcontroller after a time delay dependent on the AWU prescaler value. It is set and cleared by software. 0: AWUFH (auto wakeup from halt) mode disabled 1: AWUFH (auto wakeup from halt) mode enabled

AWUFH prescaler register (AWUPR)

AWUPR — Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
AWUPR[7:0]							
R/W							

Table 20. AWUPR register description

Bit	Bit name	Function
7:0	AWUPR[7:0]	Auto wakeup prescaler These 8 bits define the AWUPR dividing factor as explained in *Table 21*

Table 21. AWUPR dividing factor

AWUPR[7:0]	Dividing factor
00h	Forbidden
01h	1
...	...
FEh	254
FFh	255

In AWU mode, the period that the MCU stays in halt mode (t_{AWU} in *Figure 30: AWUF halt timing diagram on page 64*) is defined by

$$t_{AWU} = 64 \times AWUPR \times \frac{1}{f_{AWURC}} + t_{RCSTRT}$$

This prescaler register can be programmed to modify the time that the MCU stays in halt mode before waking up automatically.

Note: If 00h is written to AWUPR, depending on the product, an interrupt is generated immediately after a HALT instruction or the AWUPR remains unchanged.

Table 22. AWU register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0049h	**AWUPR** Reset value	AWUPR7 1	AWUPR6 1	AWUPR5 1	AWUPR4 1	AWUPR3 1	AWUPR2 1	AWUPR1 1	AWUPR0 1
004Ah	**AWUCSR** Reset value	0	0	0	0	0	AWUF	AWUM	AWUEN

10 I/O ports

10.1 Introduction

The I/O ports allow data transfer. An I/O port contains up to eight 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*Section 10.7: Device-specific I/O port configuration on page 73* 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 32: I/O port general block diagram on page 70 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 *Section 10.3: I/O port implementation on page 72* for configuration.

Note: 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.

Each external interrupt vector is linked to a dedicated group of I/O port pins (see pinout description in *Section 2: Pin description on page 16* and interrupt section).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. Changing the sensitivity bits clears 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 *Section 10.3: I/O port implementation on page 72* for configuration.

Table 23. 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 ST7 I/Os have one or more alternate functions. These may include output signals from, or input signals to, on-chip peripherals. *Table 2: Device pin description on page 17* 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 increases 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.

Figure 32. I/O port general block diagram

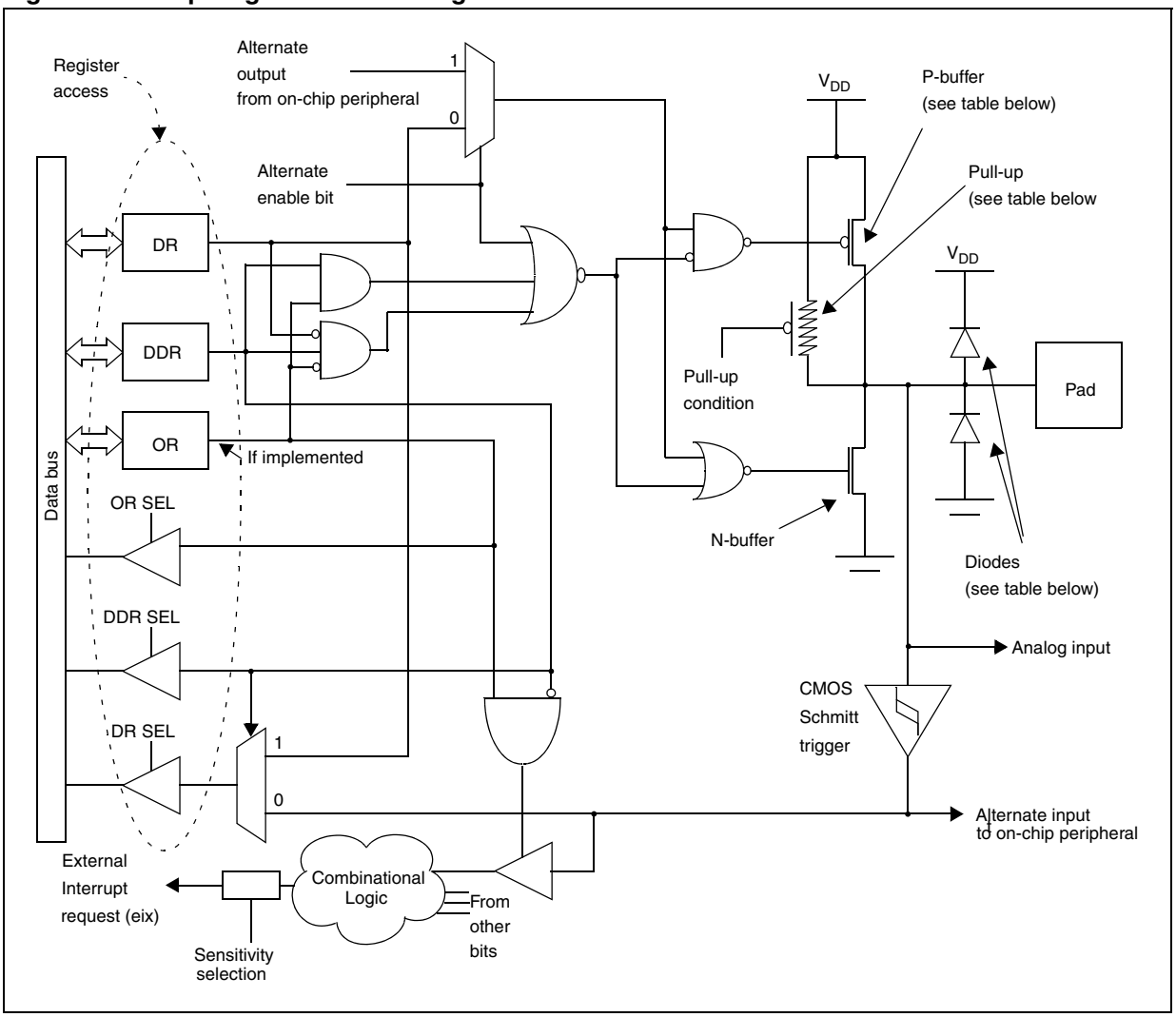
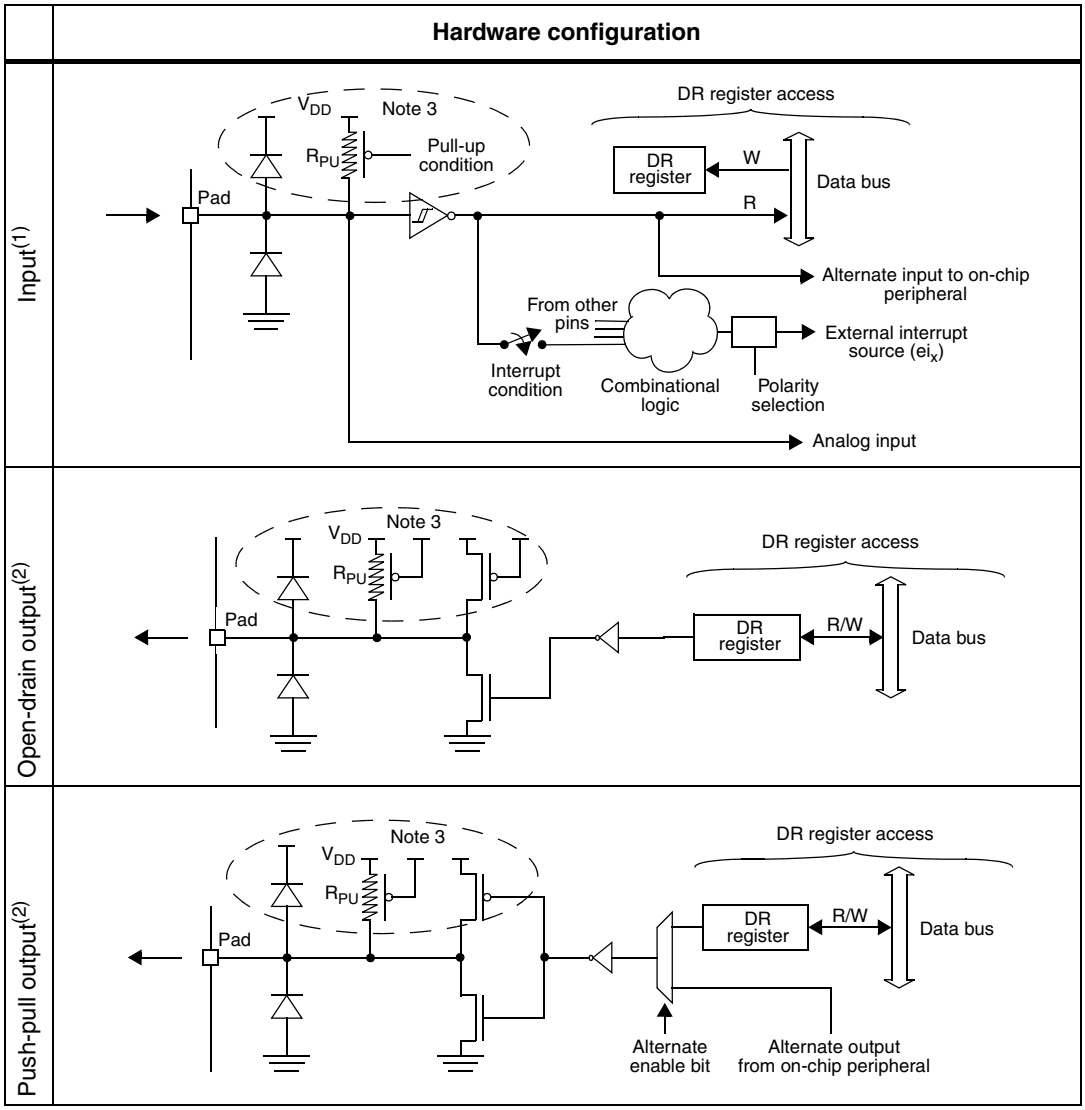


Table 24. I/O port mode options(1)

Configuration mode		Pull-up	P-buffer	Diodes	
				to V_{DD}(2)	to V_{SS}
Input	Floating with/without interrupt	Off	Off	On	On
	Pull-up with/without interrupt	On			
Output	Push-pull	Off	On		
	Open drain (logic level)		Off		
	True open drain	NI	NI	NI(3)	

1. Legend: Off = implemented not activated; On = implemented and activated
2. The diode to V_{DD} is not implemented in the true open drain pads. A local protection between the pad and V_{OL} is implemented to protect the device against positive stress.
3. For further details on port configuration, please refer to *Table 28: Port configuration (standard ports)* and *Table 29: Port configuration (external interrupts) on page 73*.

Table 25. I/O configuration



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

Analog alternate function

Configure the I/O as floating input to use an ADC input. The analog multiplexer (controlled by the ADC registers) switches the analog voltage present on the selected pin to the common analog rail, connected to the ADC input.

Analog recommendations

Do not change the voltage level or loading on any I/O while conversion is in progress. Do not have clocking pins located close to a selected analog pin.

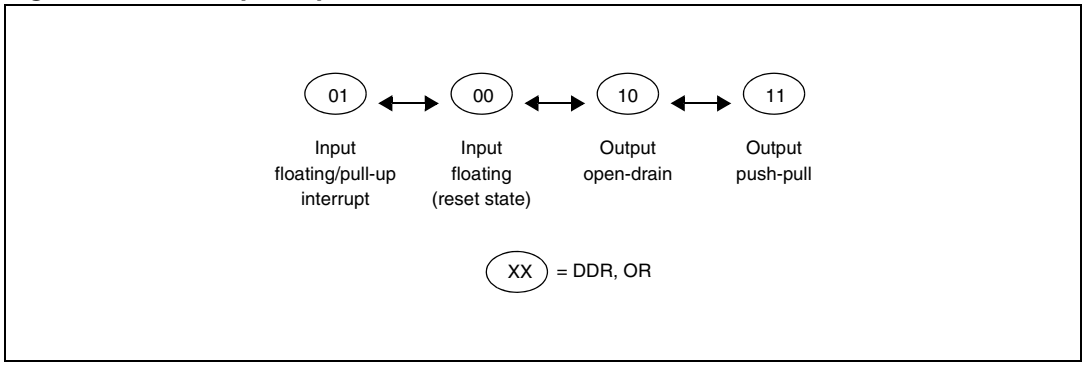
Warning: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

10.3 I/O port implementation

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific I/O port features such as ADC input or open drain.

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in *Figure 33*. Other transitions are potentially risky and should be avoided, since they may present unwanted side-effects such as spurious interrupt generation.

Figure 33. Interrupt I/O port state transitions



10.4 Unused I/O pins

Unused I/O pins must be connected to fixed voltage levels. Refer to *Section 13.8: I/O port pin characteristics on page 199*.

10.5 Low-power modes

Table 26. Effect of low power modes on I/O ports

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.

10.6 Interrupts

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and OR registers and if the I bit in the CC register is cleared (RIM instruction).

Table 27. I/O interrupt control/wake-up capability

Interrupt event	Event flag	Enable control bit	Exit from wait	Exit from halt
External interrupt on selected external event	-	DDRx ORx	Yes	Yes

Related documentation

- SPI communication between ST7 and EEPROM (AN970)
- S/W implementation of I^2C bus master (AN1045)
- Software LCD driver (AN1048)

10.7 Device-specific I/O port configuration

The I/O port register configurations are summarized as follows:

Table 28. Port configuration (standard ports)

Port	Pin name	Input (DDR = 0)		Output (DDR = 1)	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7:0	Floating	Pull-up	Open drain	Push-pull
Port B	PB6:0				

On ports where the external interrupt capability is selected using the EISR register, the configuration is as follows:

Table 29. Port configuration (external interrupts)

Port	Pin name	Input with interrupt (DDR = 0; EISR ≠ 00)	
		OR = 0	OR = 1
Port A	PA6:1	Floating	Pull-up
Port B	PB5:0		

Table 30. I/O port register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0000h	PADR Reset value	MSB 1	1	1	1	1	1	1	LSB 1
0001h	PADDR Reset value	MSB 0	0	0	0	0	0	0	LSB 0

Table 30. I/O port register map and reset values (continued)

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0002h	PAOR	MSB							LSB
	Reset value	0	1	0	0	0	0	0	0
0003h	PBDR	MSB							LSB
	Reset value	1	1	1	1	1	1	1	1
0004h	PBDDR	MSB							LSB
	Reset value	0	0	0	0	0	0	0	0
0005h	PBOR	MSB							LSB
	Reset value	0	0	0	0	0	0	0	0

11 On-chip peripherals

11.1 Watchdog timer (WDG)

11.1.1 Introduction

The watchdog timer is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The watchdog circuit generates an MCU reset upon expiration of a programmed time period, unless the program refreshes the counter's contents before the T6 bit is cleared.

11.1.2 Main features

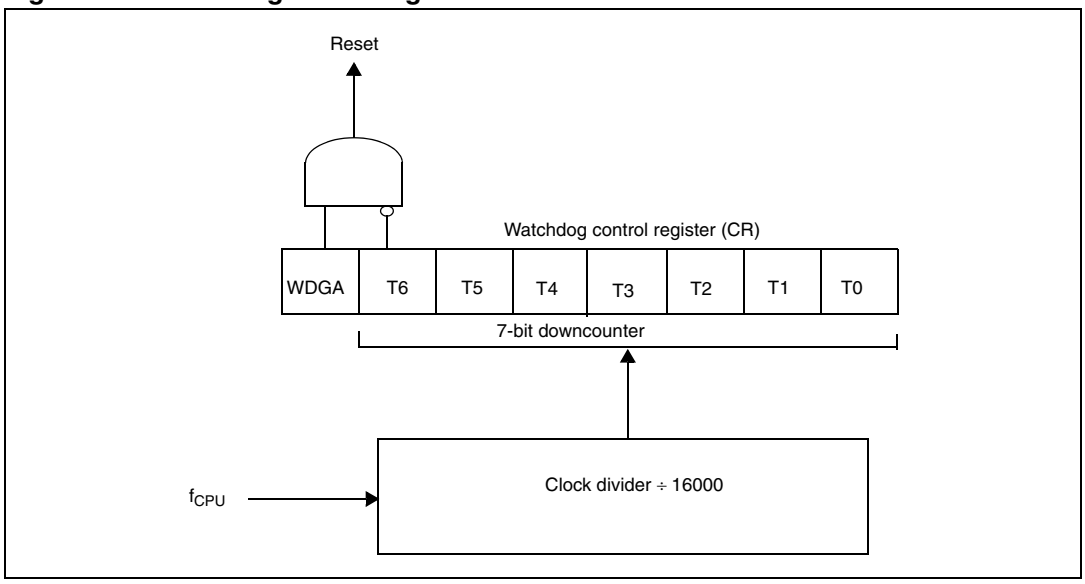
- Programmable free-running downcounter (64 increments of 16000 CPU cycles)
- Programmable reset
- Reset (if watchdog activated) when the T6 bit reaches zero
- Optional reset on HALT instruction (configurable by option byte)
- Hardware watchdog selectable by option byte

11.1.3 Functional description

The counter value stored in the CR register (bits T[6:0]) is decremented every 16000 machine cycles and the length of the timeout period can be programmed by the user in 64 increments.

If the watchdog is activated (the WDGA bit is set) and when the 7-bit timer (bits T[6:0]) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling low the reset pin for typically 30 µs.

Figure 34. Watchdog block diagram



The application program must write in the CR register at regular intervals during normal operation to prevent an MCU reset. This downcounter is free-running: It counts down, even if the watchdog is disabled. The value to be stored in the CR register must be between FFh and C0h (see *Table 31*):

- The WDGA bit is set (watchdog enabled)
- The T6 bit is set to prevent generating an immediate reset
- The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset.

Following a reset, the watchdog is disabled. Once activated, it can be disabled only by a reset.

The T6 bit can generate a software reset (the WDGA bit is set and the T6 bit is cleared).

If the watchdog is activated, the HALT instruction generates a reset.

Table 31. Watchdog timing(1)

f_{CPU} = 8 MHz		
WDG counter code	**min. (ms)**	**max (ms)**
C0h	1	2
FFh	127	128

1. The timing variation shown in *Table 31* is due to the unknown status of the prescaler when writing to the CR register.

11.1.4 Hardware watchdog option

If hardware watchdog is selected by the option byte, the watchdog is always active and the WDGA bit in the CR is not used.

Refer to the option byte description in *Section 15.2: Option bytes on page 215*.

Using halt mode with the WDG (WDGHALT option)

If halt mode with watchdog is enabled by the option byte (no watchdog reset on HALT instruction), it is recommended before executing the HALT instruction to refresh the WDG counter, to avoid an unexpected WDG reset immediately after waking up the microcontroller.

11.1.5 Interrupts

None.

11.1.6 Register description

Watchdog control register (WDGCR)

WDGCR Reset value: 0111 1111 (7Fh)

7	6	5	4	3	2	1	0
WDGA	T[6:0]						
R/W	R/W						

Table 32. WDGCR register description

Bit	Bit name	Function
7	WDGA	Activation bit(1) This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset. 0: Watchdog disabled 1: Watchdog enabled
6:0	T[6:0]	7-bit counter (MSB to LSB) These bits contain the decremented value. A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

1. The WDGA bit is not used if the hardware watchdog option is enabled by option byte.

Table 33. Watchdog timer register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
002Eh	WDGCR Reset value	WDGA 0	T6 1	T5 1	T4 1	T3 1	T2 1	T1 1	T0 1

11.2 Dual 12-bit autoreload timer 3 (AT3)

11.2.1 Introduction

The 12-bit autoreload timer can be used for general-purpose timing functions. It is based on one or two free-running 12-bit upcounters with an input capture register and four PWM output channels. There are six external pins:

- 4 PWM outputs
- ATIC/LTIC pin for the input capture function
- BREAK pin for forcing a break condition on the PWM outputs

11.2.2 Main features

- Single timer or dual timer mode with two 12-bit upcounters (CNTR1/CNTR2) and two 12-bit autoreload registers (ATR1/ATR2)
- Maskable overflow interrupts
- PWM mode
  - Generation of four independent PWMx signals
  - Dead time generation for half-bridge driving mode with programmable dead time
  - Frequency 2 kHz to 4 MHz (@ 8 MHz f_{CPU})
  - Programmable duty-cycles
  - Polarity control
  - Programmable output modes
- Output compare mode
- Input capture mode
  - 12-bit input capture register (ATICR)
  - Triggered by rising and falling edges
  - Maskable IC interrupt
  - Long range input capture
- Break control
- Flexible clock control

Figure 35. Single timer mode (ENCNTR2 = 0)

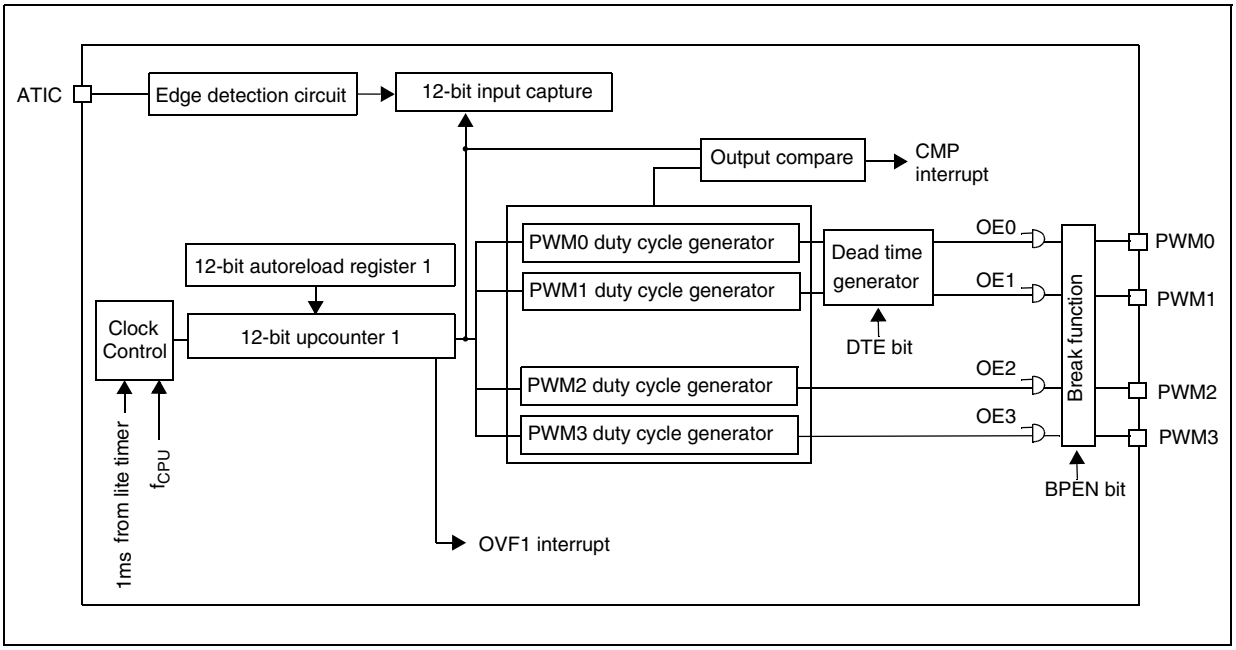
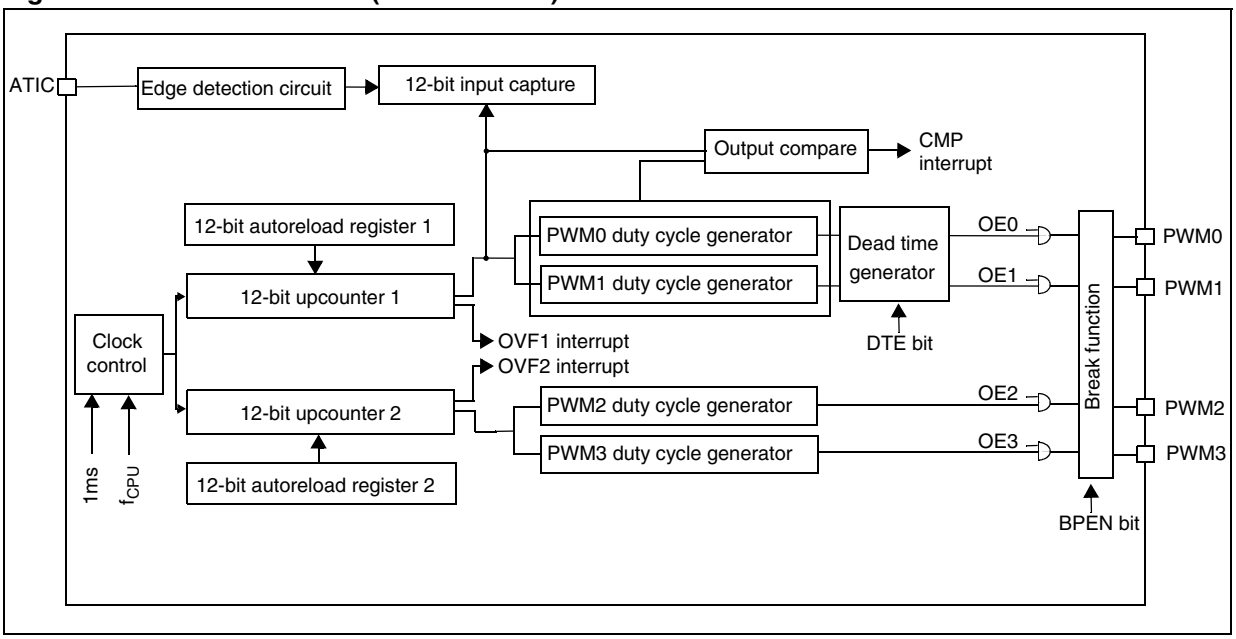


Figure 36. Dual timer mode (ENCNTR2 = 1)



11.2.3 Functional description

PWM mode

This mode allows up to four pulse width modulated signals to be generated on the PWMx output pins.

PWM frequency

The four PWM signals can have the same frequency (f_{PWM}) or can have two different frequencies. This is selected by the ENCNTR2 bit which enables single timer or dual timer mode (see *Figure 35: Single timer mode (ENCNTR2 = 0) on page 79* and *Figure 36: Dual timer mode (ENCNTR2 = 1) on page 79*).

The frequency is controlled by the counter period and the ATR register value. In dual timer mode, PWM2 and PWM3 can be generated with a different frequency controlled by CNTR2 and ATR2.

$$f_{PWM} = f_{COUNTER}/(4096 - ATR)$$

Following the above formula, if $f_{COUNTER}$ is 4 MHz, the maximum value of f_{PWM} is 2 MHz (ATR register value = 4094), the minimum value is 1 kHz (ATR register value = 0).

Duty cycle

The duty cycle is selected by programming the DCRx registers. These are preload registers. The DCRx values are transferred in active duty cycle registers after an overflow event if the corresponding transfer bit (TRANx bit) is set.

The TRAN1 bit controls the PWMx outputs driven by counter 1 and the TRAN2 bit controls the PWMx outputs driven by counter 2.

PWM generation and output compare are done by comparing these active DCRx values with the counter.

The maximum available resolution for the PWMx duty cycle is:

$$Resolution = 1/(4096 - ATR)$$

Where ATR is equal to 0. With this maximum resolution, 0% and 100% duty cycle can be obtained by changing the polarity.

At reset, the counter starts counting from 0.

When an upcounter overflow occurs (OVF event), the preloaded duty cycle values are transferred to the active duty cycle registers and the PWMx signals are set to a high level. When the upcounter matches the active DCRx value, the PWMx signals are set to a low level. To obtain a signal on a PWMx pin, the contents of the corresponding active DCRx register must be greater than the contents of the ATR register.

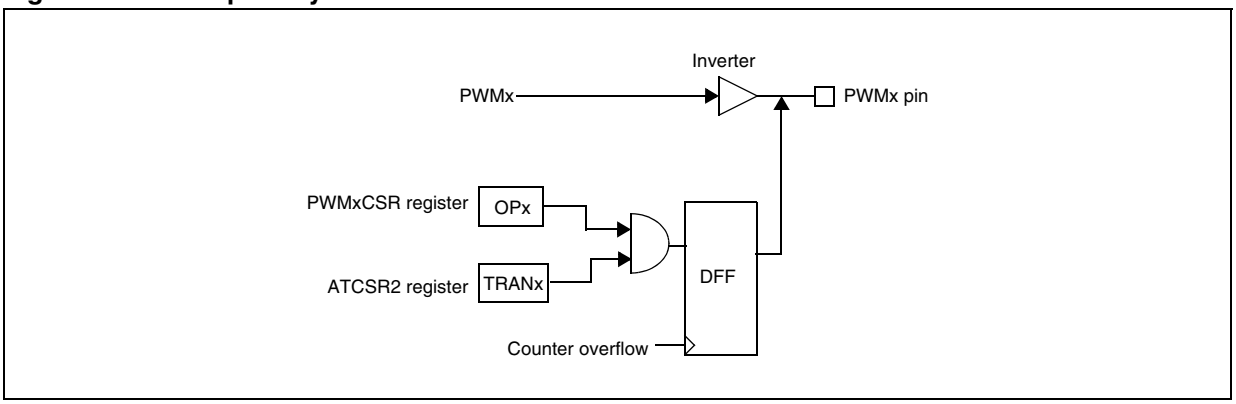
Note: ***For ROM devices only:*** *The PWM can be enabled/disabled only in overflow ISR, otherwise the first pulse of PWM can be different from expected one because no force overflow function is present.*

The maximum value of ATR is 4094 because it must be lower than the DCR value, which in this case must be 4095.

Polarity inversion

The polarity bits can be used to invert any of the four output signals. The inversion is synchronized with the counter overflow if the corresponding transfer bit in the ATCSR2 register is set (reset value). See *Figure 37*.

Figure 37. PWM polarity inversion



The data flip flop (DFF) applies the polarity inversion when triggered by the counter overflow input.

Output control

The PWMx output signals can be enabled or disabled using the OEx bits in the PWMCR register.

Figure 38. PWM function

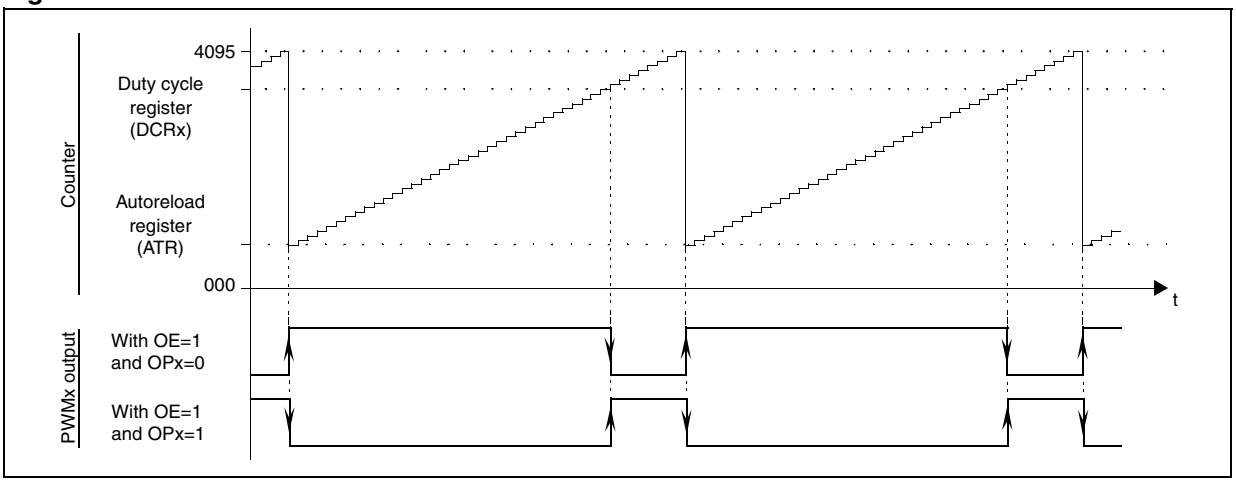
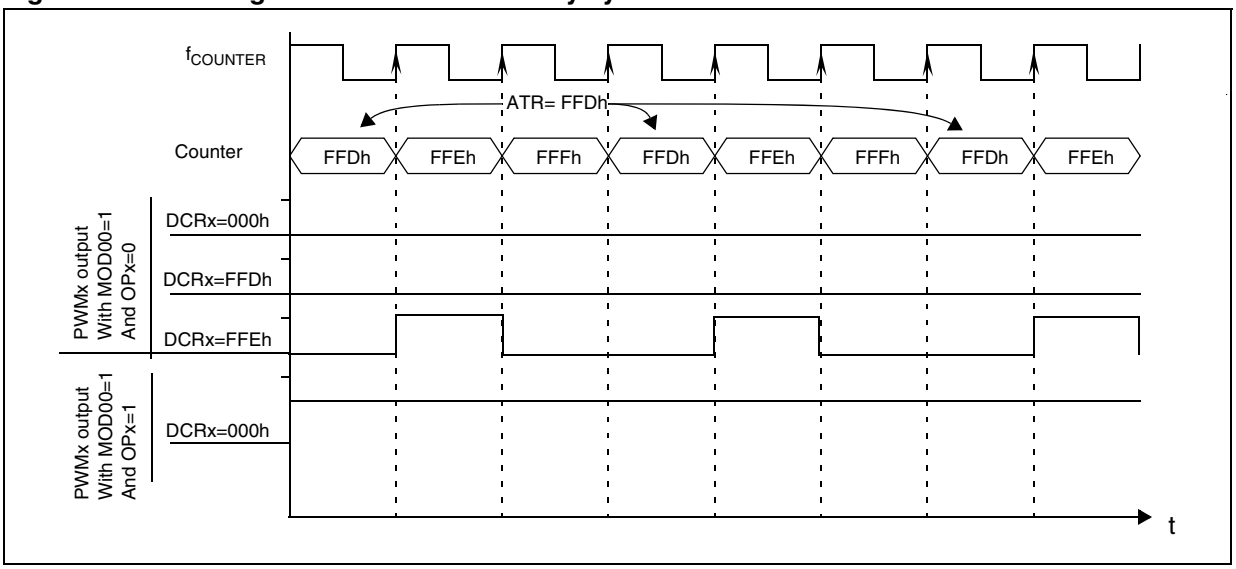


Figure 39. PWM signal from 0% to 100%duty cycle



Dead time generation

A dead time can be inserted between PWM0 and PWM1 using the DTGR register. This is required for half-bridge driving where PWM signals must not be overlapped. The non-overlapping PWM0/PWM1 signals are generated through a programmable dead time by setting the DTE bit.

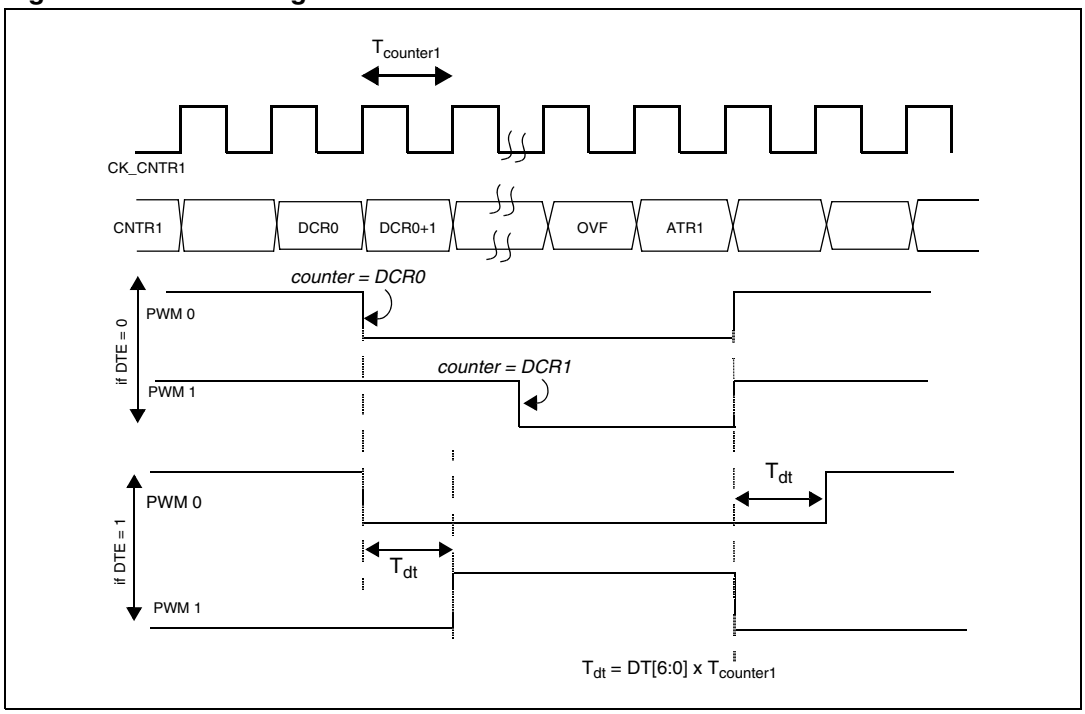
Dead time value = DT[6:0] x Tcounter1

DTGR[7:0] is buffered inside so as to avoid deforming the current PWM cycle. The DTGR effect will take place only after an overflow.

Note: 1 Dead time is generated only when DTE = 1 and DT[6:0] ≠ 0. If DTE is set and DT[6:0] = 0, PWM output signals will be at their reset state.

2 Half-bridge driving is possible only if polarities of PWM0 and PWM1 are not inverted, that is, if OP0 and OP1 are not set. If polarity is inverted, overlapping PWM0/PWM1 signals will be generated.

Figure 40. Dead time generation



In the above example, when the DTE bit is set:

- PWM goes low at DCR0 match and goes high at ATR1 + T_{dt}
- PWM1 goes high at DCR0 + T_{dt} and goes low at ATR match.

With this programmable delay (T_{dt}), the PWM0 and PWM1 signals which are generated are not overlapped.

Break function

The break function can be used to perform an emergency shutdown of the application being driven by the PWM signals.

The break function is activated by the external BREAK pin (active low). In order to use the break pin it must be previously enabled by software setting the BPEN bit in the BREAKCR register.

When a low level is detected on the break pin, the BA bit is set and the break function is activated. In this case, the four PWM signals are stopped.

Software can set the BA bit to activate the break function without using the break pin.

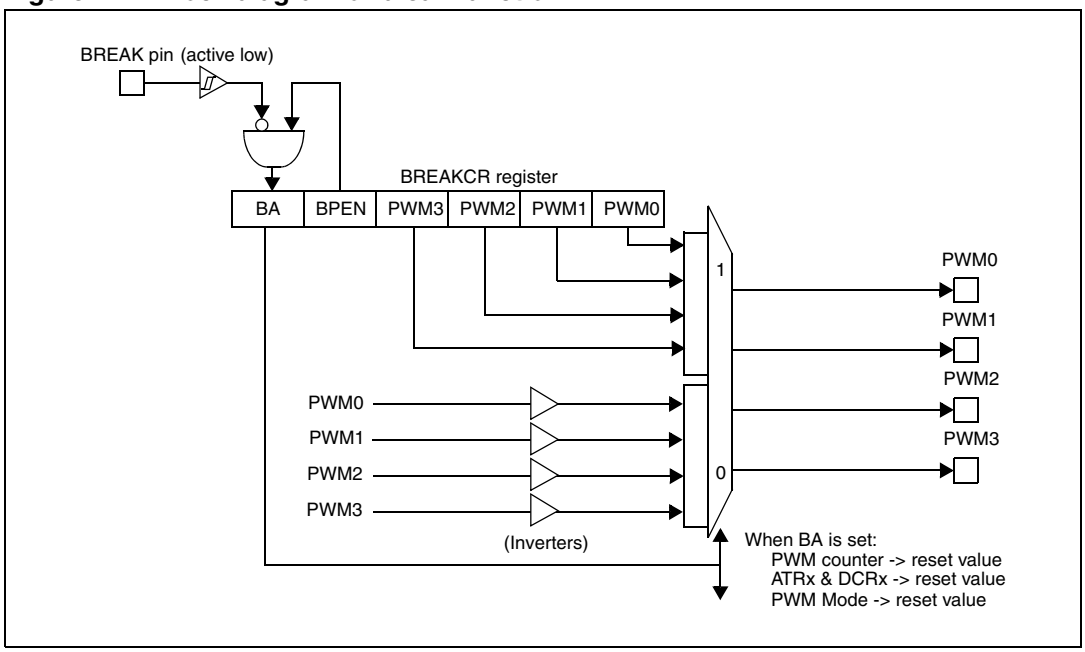
When a break function is activated (BA bit = 1 and BREN1/BREN2 = 1):

- The break pattern (PWM[3:0] bits in the BREAKCR is forced directly on the PWMx output pins (after the inverter)
- The 12-bit PWM counter CNTR1 is put to its reset value, that is, 00h
- The 12-bit PWM counter CNTR2 is put to its reset value, that is 00h
- ATR1, ATR2, preload and active DCRx are put to their reset values
- The PWMCR register is reset
- Counters stop counting

When the break function is deactivated after applying the break (BA bit goes from 1 to 0 by software):

- The control of the four PWM outputs is transferred to the port registers.

Figure 41. Block diagram of break function



1. The BREAK pin value is latched by the BA bit

Output compare mode

To use this function, load a 12-bit value in the preload DCRxH and DCRxL registers.

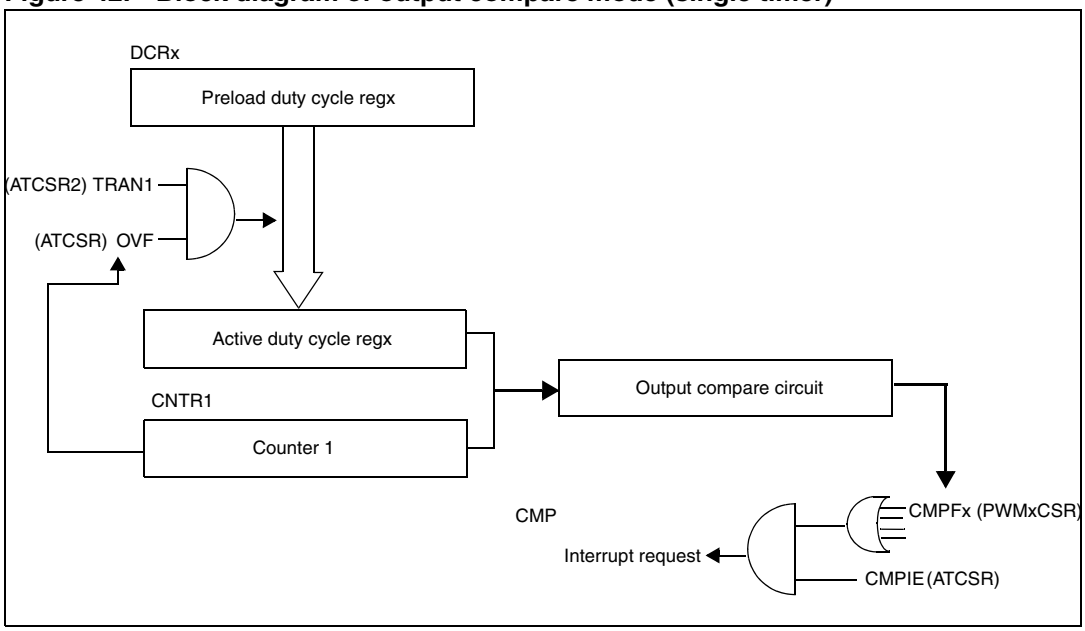
When the 12-bit upcounter CNTR1 reaches the value stored in the active DCRxH and DCRxL registers, the CMPFx bit in the PWMxCSR register is set and an interrupt request is generated if the CMPIE bit is set.

The output compare function is always performed on CNTR1 in both single timer mode and dual timer mode and never on CNTR2. The difference is that in single timer mode the counter 1 can be compared with any of the four DCR registers and in dual timer mode, the counter 1 is compared with DCR0 or DCR1.

Note: 1 The output compare function is only available for DCRx values other than 0 (reset value).

2 Duty cycle registers are buffered internally. The CPU writes in preload duty cycle registers and these values are transferred to active duty cycle registers after an overflow event if the corresponding transfer bit (TRAN1 bit) is set. Output compare is done by comparing these active DCRx values with the counter.

Figure 42. Block diagram of output compare mode (single timer)



Input capture mode

The 12-bit ATICR register is used to latch the value of the 12-bit free running upcounter CNTR1 after a rising or falling edge is detected on the ATIC pin. When an input capture occurs, the ICF bit is set and the ATICR register contains the value of the free running upcounter. An IC interrupt is generated if the ICIE bit is set. The ICF bit is reset by reading the ATICRH/ATICRL register when the ICF bit is set. The ATICR is a read only register and always contains the free running upcounter value which corresponds to the most recent input capture. Any further input capture is inhibited while the ICF bit is set.

Figure 43. Block diagram of input capture mode

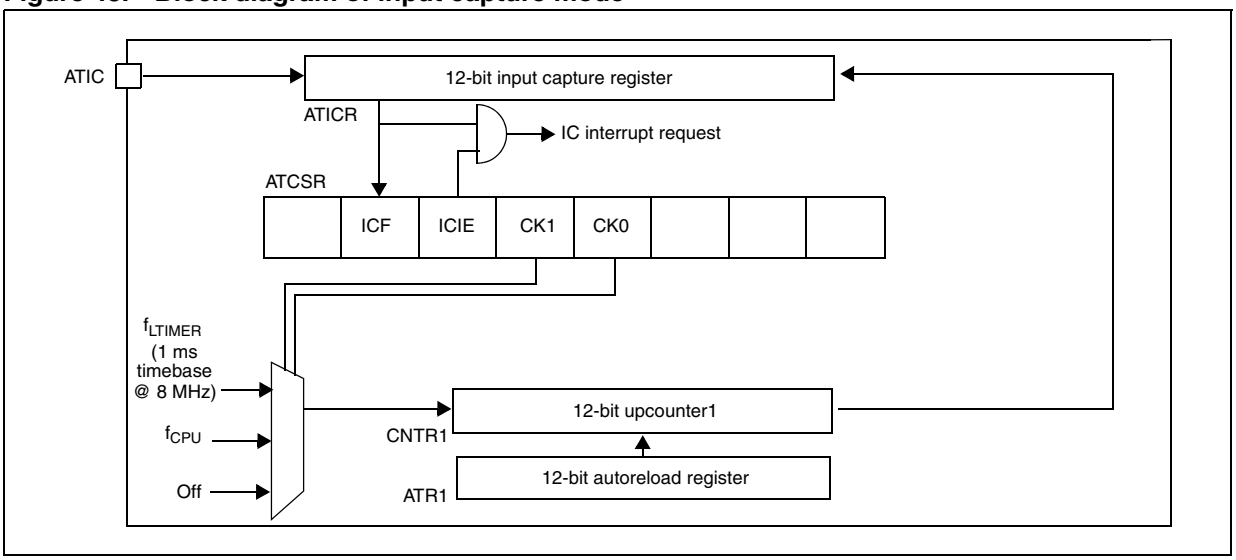
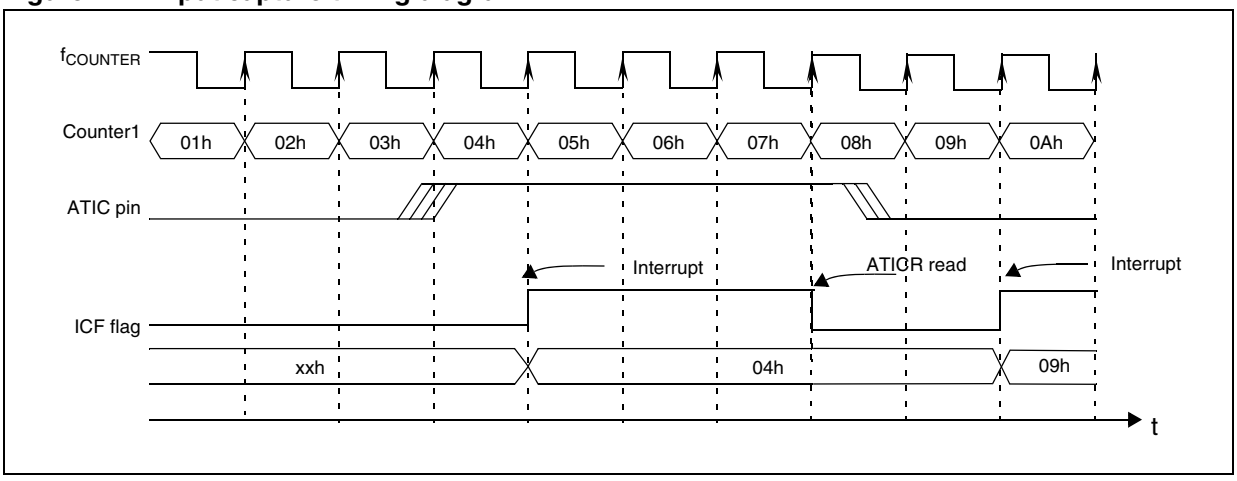


Figure 44. Input capture timing diagram



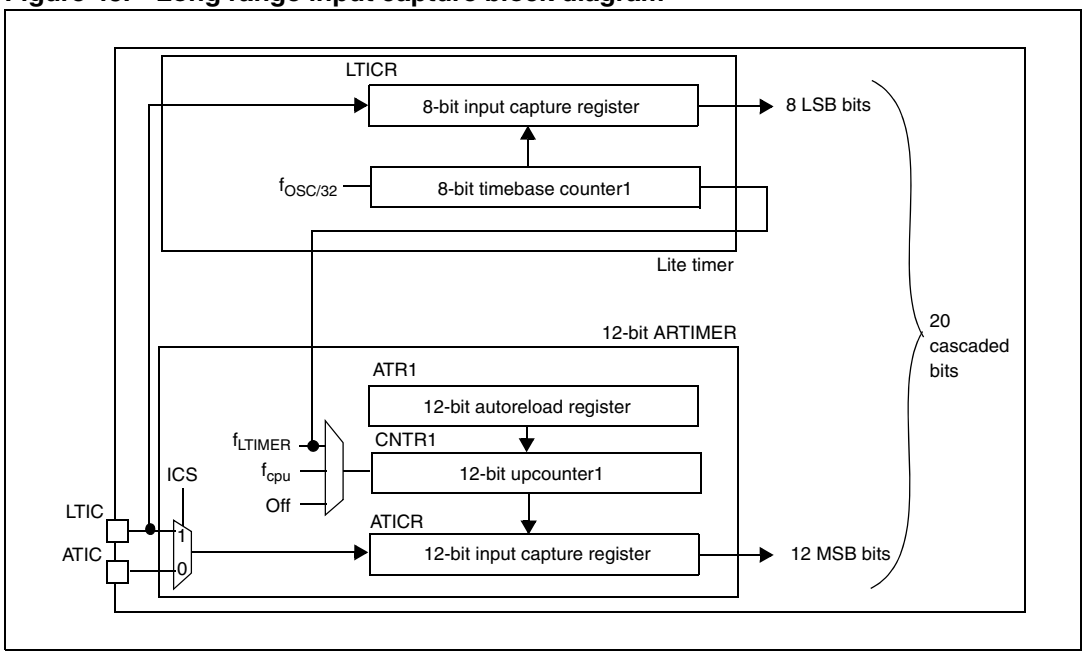
Long input capture

Pulses that last between 8 µs and 2 s can be measured with an accuracy of 4µs if f_{OSC} = 8 MHz under 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 45*.

Figure 45. Long range input capture block diagram



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.

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 the following:

- First, reset both ICIE bits
- Then set the ICS bit
- Reset both ICF bits
- Then set the ICIE bit of desired interrupt

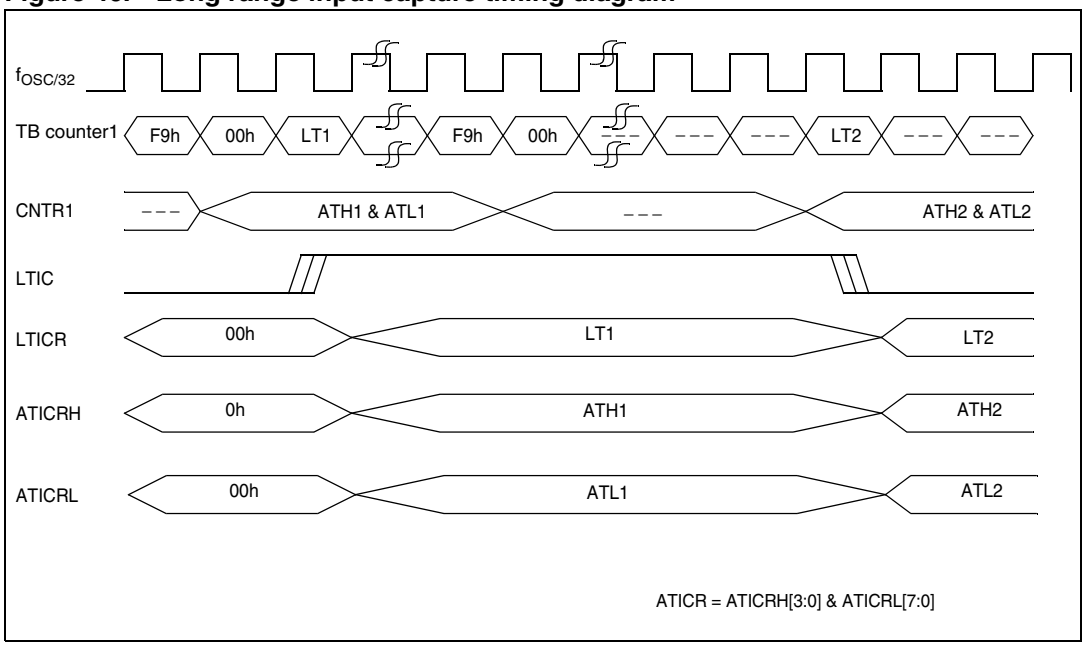
Both timers are used to compute a pulse length with long input capture feature. The procedure is not straight-forward and is as follows:

- 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 46*.
- At the second input capture on the falling edge of the pulse, we assume that the values in the registers are as follows:
  - LTICR = LT2
  - ATICRH = ATH2
  - ATICRL = ATL2
  - Hence ATICR2 [11:0] = ATH2 & ATL2

Now pulse width P between first capture and second capture is:

P = decimal (F9 – LT1 + LT2 + 1) * 0.004ms + decimal (ATICR2 - ATICR1 – 1) * 1ms

Figure 46. Long range input capture timing diagram



11.2.4 Low power modes

Table 34. Effect of low power modes on AT3 timer

Mode	Description
Slow	The input frequency is divided by 32
Wait	No effect on AT timer

Table 34. Effect of low power modes on AT3 timer (continued)

Active halt	AT timer halted except if CK0 = 1, CK1 = 0 and OVFIE = 1
Halt	AT timer halted

11.2.5 Interrupts

Table 35. AT3 interrupt control/wake-up capability

Interrupt event(1)	Event flag	Enable control bit	Exit from wait	Exit from Halt	Exit from active halt
Overflow event	OVF1	OVFIE1	Yes	No	Yes(2)
AT3 IC event	ICF	ICIE			No
CMP event	CMPFx	CMPIE			

1. The CMP and AT3 IC events are connected to the same interrupt vector. The OVF event is mapped on a separate vector (see *Section 8: Interrupts*). They generate an interrupt if the enable bit is set in the ATCSR register and the interrupt mask in the CC register is reset (RIM instruction).
2. Only if CK0 = 1 and CK1 = 0 ($f_{COUNTER} = f_{LTIMER}$)

11.2.6 Register description

Timer control status register (ATCSR)

ATCSR Reset value: 0x00 0000 (x0h)

7	6	5	4	3	2	1	0
Reserved	ICF	ICIE	CK[1:0]		OVF1	OVFIE1	CMPIE
-	R/W	R/W	R/W		R/W	R/W	R/W

Table 36. ATCSR register description

Bit	Bit name	Function
7	-	Reserved, must be kept cleared
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 clears this flag). Writing to this bit does not change the bit value. 0: No input capture 1: An input capture has occurred
5	ICIE	IC interrupt enable This bit is set and cleared by software. 0: Input capture interrupt disabled 1: Input capture interrupt enabled

Table 36. ATCSR register description (continued)

Bit	Bit name	Function
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 as follows/ 00: Counter clock selection = off 01: Counter clock selection = f_{LTIMER} (1ms timebase @ 8 MHz) 10: Counter clock selection = f_{CPU} 11: Counter clock selection = off
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
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
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)

CNTR1H — Reset value: 0000 0000 (00h)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	CNTR1[11:8]			
-	-	-	-	R			

Counter register 1 low (CNTR1L)

CNTR1L — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CNTR1[7:0]							
R							

Table 37. CNTR1H and CNTR1L register descriptions

Bit	Bit name	Function
15:12	-	Reserved, must be kept cleared
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 increments 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.

Autoreload register high (ATR1H)

ATR1H — Reset value: 0000 0000 (00h)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	ATR1[11:8]			
-	-	-	-	R/W			

Autoreload register low (ATR1L)

ATR1L — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ATR1[7:0]							
R/W							

Table 38. ATR1H and ATR1L register descriptions

Bit	Bit name	Function
15:12	-	Reserved, must be kept cleared
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)

PWMCR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	OE3	Reserved	OE2	Reserved	OE1	Reserved	OE0
-	R/W	-	R/W	-	R/W	-	R/W

Table 39. PWMCR register description

Bit	Bit name	Function
7, 5, 3, 1	-	Reserved, must be kept cleared
6, 4, 2, 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)

PWMxCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	OPx	CMPFx
-	-	-	-	-	-	R/W	R/W

Table 40. PWMxCSR register description

Bit	Bit name	Function
7:2	-	Reserved, must be kept cleared
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
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)

BREAKCR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	Reserved	BA	BPEN	PWM[3:0]			
-	-	R/W	R/W	R/W			

Table 41. BREAKCR register description

Bit	Bit name	Function
7:6	-	Reserved, must be kept cleared
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
4	BPEN	Break pin enable This bit is read/write by software and cleared by hardware after reset. 0: Break pin disabled 1: Break pin enabled
3:0	PWM[3:0]	Break pattern These bits are read/write by software and cleared by hardware after a reset. They are used to force the four PWMx output signals into a stable state when the break function is active and corresponding OEx bit is set.

PWMx duty cycle register high (DCRxH)

DCRxH Reset value: 0000 0000 (00h)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	DCRx[11:8]			
-	-	-	-	R/W			

PWMx duty cycle register low (DCRxL)

DCRxL Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
DCRx[7:0]							
R/W							

Table 42. DCRxH and DCRxL register descriptions

Bit	Bit name	Function
15:12	-	Reserved, must be kept cleared
11:0	DCRx[11:0]	PWMx duty cycle value This 12-bit value is written by software. It defines the duty cycle of the corresponding PWM output signal (see *Figure 38: PWM function on page 81*). In PWM mode (OEx = 1 in the PWMCR register) the DCRx[11:0] bits define the duty cycle of the PWMx output signal (see *Figure 38*). In output compare mode, they define the value to be compared with the 12-bit upcounter value.

Input capture register high (ATICRH)

ATICRH — Reset value: 0000 0000 (00h)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	ICR[11:8]			
-	-	-	-	R			

Input capture register low (ATICRL)

ATICRL — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ICR[7:0]							
R							

Table 43. ATICRH and ATICRL register descriptions

Bit	Bit name	Function
15:12	-	Reserved, must be kept cleared
11:0	ICR[11:0]	Input capture data This is a 12-bit register which is readable by software and cleared by hardware after a reset. The ATICR register contains the captured value of the 12-bit CNTR1 register when a rising or falling edge occurs on the ATIC or LTIC pin (depending on ICS). Capture will only be performed when the ICF flag is cleared.

Timer control register2 (ATCSR2)

ATCSR2 Reset value: 0000 0011 (03h)

7	6	5	4	3	2	1	0
Reserved	Reserved	ICS	OVFIE2	OVF2	ENCNTR2	TRAN2	TRAN1
-	-	R/W	R/W	R/W	R/W	R/W	R/W

Table 44. ATCSR2 register description

Bit	Bit name	Function
7:6	-	Reserved, must be kept cleared
5	ICS	Input capture shorted This bit is read/write by software. It allows the AT timer CNTR1 to use the LTIC pin for long input capture. 0: ATIC for CNTR1 input capture 1: LTIC for CNTR1 input capture
4	OVFIE2	Overflow interrupt 2 enable This bit is read/write by software and controls the overflow interrupt of counter 2. 0: Overflow interrupt disabled 1: Overflow interrupt enabled
3	OVF2	Overflow flag This bit is set by hardware and cleared by software by reading the ATCSR2 register. It indicates the transition of the counter 2 from FFFh to ATR2 value. 0: No counter overflow occurred 1: Counter overflow occurred
2	ENCNTR2	Enable counter 2 This bit is read/write by software and switches the second counter CNTR2. If this bit is set, PWM2/3 is generated using CNTR2 0: CNTR2 stopped 1: CNTR2 starts running
1	TRAN2	Transfer enable 2 This bit is read/write by software, cleared by hardware after each completed transfer and set by hardware after reset. It controls the transfers on CNTR2. It allows the value of the preload DCRx registers to be transferred to the active DCRx registers after the next overflow event. The OPx bits are transferred to the shadow OPx bits in the same way. *Note: Only DCR2/3 can be controlled using this bit*

Table 44. ATCSR2 register description (continued)

Bit	Bit name	Function
0	TRAN1	Transfer enable 1 This bit is read/write by software, cleared by hardware after each completed transfer and set by hardware after reset. It controls the transfers on CNTR1. It allows the value of the preload DCRx registers to be transferred to the active DCRx registers after the next overflow event. The OPx bits are transferred to the shadow OPx bits in the same way.

Autoreload register2 high (ATR2H)

ATR2H — Reset value: 0000 0000 (00h)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	ATR2[11:8]			
-	-	-	-	R/W			

Autoreload register2 low (ATR2L)

ATR2L — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ATR2[7:0]							
R/W							

Table 45. ATR2H and ATR2L register descriptions

Bit	Bit name	Function
15:12	-	Reserved, must be kept cleared
11:0	ATR2[11:0]	Autoreload register 2 This is a 12-bit register which is written by software. The ATR2 register value is automatically loaded into the upcounter CNTR2 when an overflow of CNTR2 occurs. The register value is used to set the PWM2/PWM3 frequency when ENCNTR2 is set.

Dead time generator register (DTGR)

DTGR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
DTE	DT[6:0]						
R/W	R/W						

Table 46. DTGR register description

Bit	Bit name	Function
7	DTE	Dead time enable This bit is read/write by software. It enables a dead time generation on PWM0/PWM1. 0: No dead time insertion 1: Dead time insertion enabled
6:0	DT[6:0]	Dead time value These bits are read/write by software. They define the dead time inserted between PWM0/PWM1. Dead time is calculated as follows: Dead time = DT[6:0] x Tcounter1

Table 47. Register map and reset values

Add (Hex.)	Register label	7	6	5	4	3	2	1	0
0D	ATCSR Reset value	0	ICF 0	ICIE 0	CK1 0	CK0 0	OVF1 0	OVFIE1 0	CMPIE 0
0E	CNTR1H Reset value	0	0	0	0	CNTR1_11 0	CNTR1_10 0	CNTR1_9 0	CNTR1_8 0
0F	CNTR1L Reset value	CNTR1_7 0	CNTR1_6 0	CNTR1_5 0	CNTR1_4 0	CNTR1_3 0	CNTR1_2 0	CNTR1_1 0	CNTR1_0 0
10	ATR1H Reset value	0	0	0	0	ATR11 0	ATR10 0	ATR9 0	ATR8 0
11	ATR1L Reset value	ATR7 0	ATR6 0	ATR5 0	ATR4 0	ATR3 0	ATR2 0	ATR1 0	ATR0 0
12	PWMCR Reset value	0	OE3 0	0	OE2 0	0	OE1 0	0	OE0 0
13	PWM0CSR Reset value	0	0	0	0	0	0	OP0 0	CMPF0 0
14	PWM1CSR Reset value	0	0	0	0	0	0	OP1 0	CMPF1 0
15	PWM2CSR Reset value	0	0	0	0	0	0	OP2 0	CMPF2 0
16	PWM3CSR Reset value	0	0	0	0	0	0	OP3 0	CMPF3 0
17	DCR0H Reset value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
18	DCR0L Reset value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
19	DCR1H Reset value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
1A	DCR1L Reset value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
1B	DCR2H Reset value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
1C	DCR2L Reset value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
1D	DCR3H Reset value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
1E	DCR3L Reset value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
1F	ATICRH Reset value	0	0	0	0	ICR11 0	ICR10 0	ICR9 0	ICR8 0
20	ATICRL Reset value	ICR7 0	ICR6 0	ICR5 0	ICR4 0	ICR3 0	ICR2 0	ICR1 0	ICR0 0
21	ATCSR2 Reset value	0	0	ICS 0	OVFIE2 0	OVF2 0	ENCNTR2 0	TRAN2 1	TRAN1 1

Table 47. Register map and reset values (continued)

Add (Hex.)	Register label	7	6	5	4	3	2	1	0
22	BREAKCR Reset value	0	0	BA 0	BPEN 0	PWM3 0	PWM2 0	PWM1 0	PWM0 0
23	ATR2H Reset value	0	0	0	0	ATR11 0	ATR10 0	ATR9 0	ATR8 0
24	ATR2L Reset value	ATR7 0	ATR6 0	ATR5 0	ATR4 0	ATR3 0	ATR2 0	ATR1 0	ATR0 0
25	DTGR Reset value	DTE 0	DT6 0	DT5 0	DT4 0	DT3 0	DT2 0	DT1 0	DT0 0

11.3 Lite timer 2 (LT2)

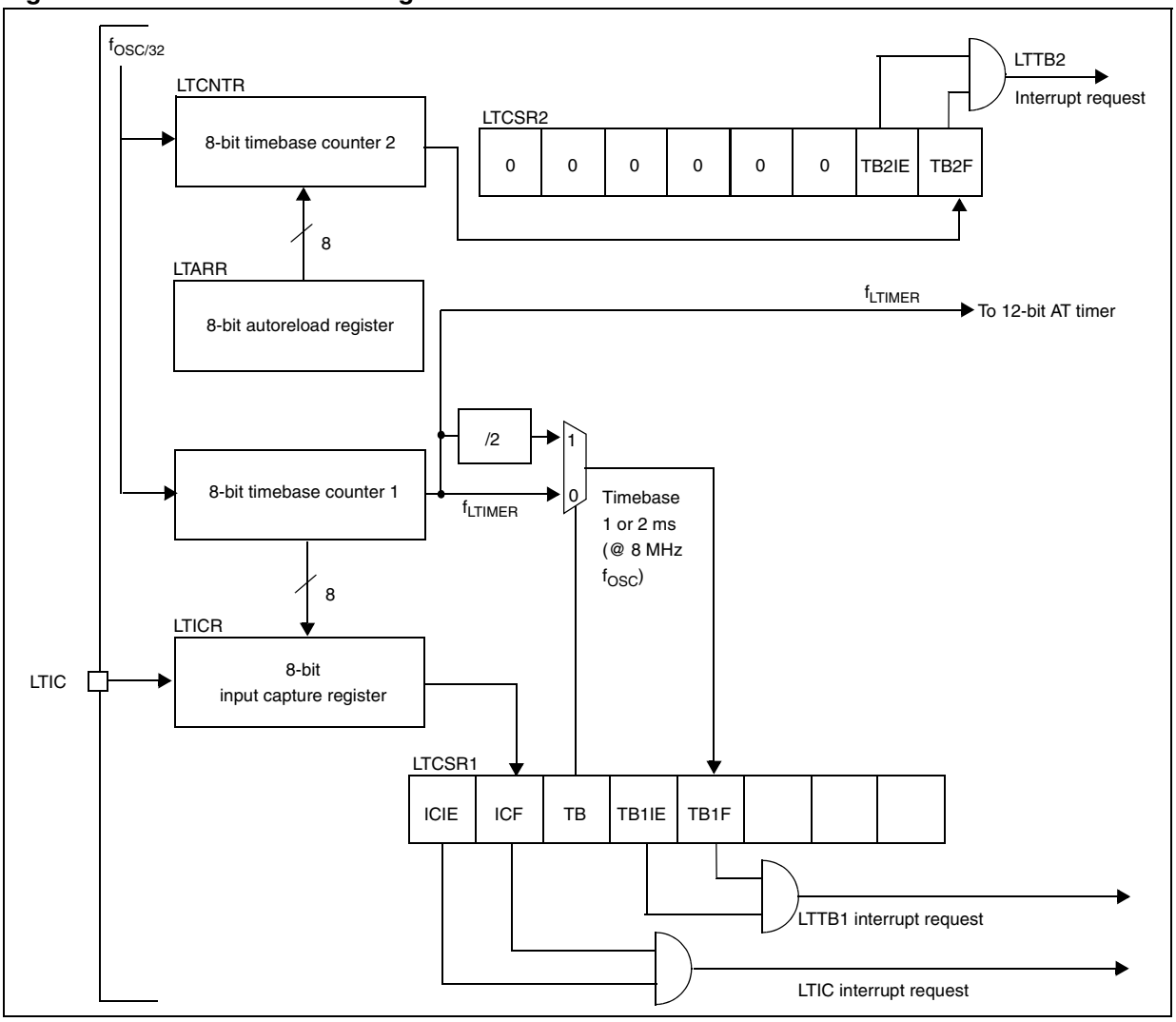
11.3.1 Introduction

The lite timer is 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

- Real-time 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 47. Lite timer 2 block diagram



11.3.3 Functional description

Timebase counter 1

The 8-bit value of counter 1 cannot be read or written by software. After an MCU reset, it starts incrementing from 0 at a frequency of $f_{OSC}/32$. An overflow event occurs when the counter rolls over from F9h to 00h. If f_{OSC} = 8 MHz, then the time period between two counter overflow events is 1 ms. This period can be doubled by setting the TB bit in the LTCSR1 register.

When counter 1 overflows, the TB1F bit is set by hardware and an interrupt request is generated if the TB1IE bit is set. The TB1F bit is cleared by software reading the LTCSR1 register.

Timebase counter 2

Counter 2 is an 8-bit autoreload upcounter. It can be read by accessing the LTCNTR register. After an MCU reset, it increments at a frequency of $f_{OSC}/32$ starting from the value stored in the LTARR register. A counter overflow event occurs when the counter rolls over from FFh to the LTARR reload value. Software can write a new value at anytime in the LTARR register, this value will be automatically loaded in the counter when the next overflow occurs.

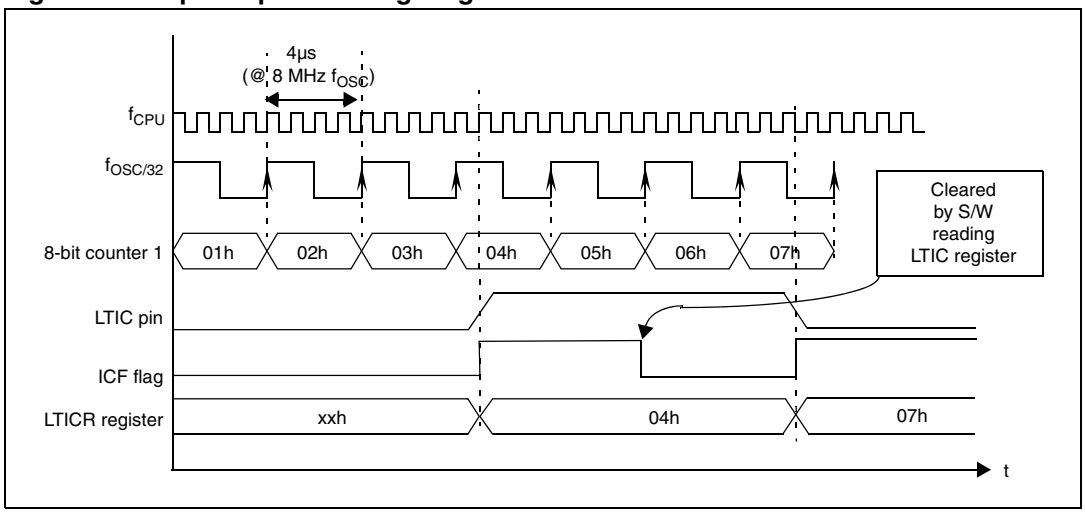
When counter 2 overflows, the TB2F bit in the LTCSR2 register is set by hardware and an interrupt request is generated if the TB2IE bit is set. The TB2F bit is cleared by software reading the LTCSR2 register.

Input capture

The 8-bit input capture register is used to latch the free-running upcounter (counter 1) 1 after a rising or falling edge is detected on the LTIC pin. When an input capture occurs, the ICF bit is set and the LTICR register contains the value of counter 1. An interrupt is generated if the ICIE bit is set. The ICF bit is cleared by reading the LTICR register.

The LTICR is a read-only register and always contains the data from the last input capture. Input capture is inhibited if the ICF bit is set.

Figure 48. Input capture timing diagram



11.3.4 Low power modes

Table 48. Effect of low power modes on lite timer 2

Mode	Description
Slow	No effect on lite timer (this peripheral is driven directly by $f_{OSC}/32$)
Wait	No effect on lite timer
Active halt	
Halt	Lite timer stops counting

Table 49. Lite timer 2 interrupt control/wake-up capability(1)

Interrupt event	Event flag	Enable control bit	Exit from wait	Exit from active halt	Exit from halt
Timebase 1 event	TB1F	TB1IE	Yes	Yes	No
Timebase 2 event	TB2F	TB2IE		No	
IC event	ICF	ICIE			

1. The TBxF and ICF interrupt events are connected to separate interrupt vectors (see *Section 8: Interrupts*). They generate an interrupt if the enable bit is set in the LTCSR1 or LTCSR2 register and the interrupt mask in the CC register is reset (RIM instruction).

11.3.5 Register description

Lite timer control/status register 2 (LTCSR2)

LTCSR2 Reset value: 0x00 0000 (x0h)

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	TB2IE	TB2F
-	-	-	-	-	-	R/W	R/W

Table 50. LTCSR2 register description

Bit	Bit name	Function
7:2	-	Reserved, must be kept cleared
1	TB2IE	Timebase 2 interrupt enable This bit is set and cleared by software. 0: Timebase (TB2) interrupt disabled 1: Timebase (TB2) interrupt enabled
0	TB2F	Timebase 2 interrupt flag This bit is set by hardware and cleared by software reading the LTCSR register. Writing to this bit has no effect. 0: No counter 2 overflow 1: A counter 2 overflow has occurred

Lite timer autoreload register (LTARR)

LTARR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
AR[7:0]							
R/W							

Table 51. LTARR register description

Bit	Bit name	Function
7:0	AR[7:0]	Counter 2 reload value These bits are read/write by software. The LTARR value is automatically loaded into counter 2 (LTCNTR) when an overflow occurs.

Lite timer counter 2 (LTCNTR)

LTCNTR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CNT[7:0]							
R							

Table 52. LTCNTR register description

Bit	Bit name	Function
7:0	CNT[7:0]	Counter 2 reload value This register is read by software. The LTARR value is automatically loaded into counter 2 (LTCNTR) when an overflow occurs.

Lite timer control/status register (LTCSR1)

LTCSR1 Reset value: 0x00 0000 (x0h)

7	6	5	4	3	2	1	0
ICIE	ICF	TB	TB1IE	TB1F	Reserved	Reserved	Reserved
R/W	R/W	R/W	R/W	R/W	-	-	-

Table 53. LTCSR1 register description

Bit	Bit name	Function
7	ICIE	Interrupt enable This bit is set and cleared by software. 0: Input capture (IC) interrupt disabled 1: Input capture (IC) interrupt enabled
6	ICF	Input capture flag This bit is set by hardware and cleared by software by reading the LTICR register. Writing to this bit does not change the bit value. 0: No input capture 1: An input capture has occurred *Note: After an MCU reset, software must initialize the ICF bit by reading the LTICR register*
5	TB	Timebase period selection This bit is set and cleared by software. 0: Timebase period = t_{OSC} * 8000 (1ms @ 8 MHz) 1: Timebase period = t_{OSC} * 16000 (2ms @ 8 MHz)
4	TB1IE	Timebase interrupt enable This bit is set and cleared by software. 0: Timebase (TB1) interrupt disabled 1: Timebase (TB1) interrupt enabled

Table 53. LTCSR1 register description (continued)

Bit	Bit name	Function
3	TB1F	Timebase interrupt flag This bit is set by hardware and cleared by software reading the LTCSR register. Writing to this bit has no effect. 0: No counter overflow 1: A counter overflow has occurred
2:0	-	Reserved, must be kept cleared

Lite timer input capture register (LTICR)

LTICR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ICR[7:0]							
R							

Table 54. LTICR register description

Bit	Bit name	Function
7:0	ICR[7:0]	Input capture value These bits are read by software and cleared by hardware after a reset. If the ICF bit in the LTCSR is cleared, the value of the 8-bit up-counter is captured when a rising or falling edge occurs on the LTIC pin.

Table 55. Lite timer register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
08	LTCSR2 Reset value	0	0	0	0	0	0	TB2IE 0	TB2F 0
09	LTARR Reset value	AR7 0	AR6 0	AR5 0	AR4 0	AR3 0	AR2 0	AR1 0	AR0 0
0A	LTCNTR Reset value	CNT7 0	CNT6 0	CNT5 0	CNT4 0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
0B	LTCSR1 Reset value	ICIE 0	ICF x	TB 0	TB1IE 0	TB1F 0	0	0	0
0C	LTICR Reset value	ICR7 0	ICR6 0	ICR5 0	ICR4 0	ICR3 0	ICR2 0	ICR1 0	ICR0 0

11.4 Serial peripheral interface (SPI)

11.4.1 Introduction

The serial peripheral interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves or a system in which devices may be either masters or slaves.

11.4.2 Main features

- Full duplex synchronous transfers (on three lines)
- Simplex synchronous transfers (on two lines)
- Master or slave operation
- 6 master mode frequencies ($f_{CPU}/4$ max.)
- $f_{CPU}/2$ max. slave mode frequency (see note below)
- $\overline{SS}$ management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, master mode fault and overrun flags

Note: In slave mode, continuous transmission is not possible at maximum frequency due to the software overhead for clearing status flags and to initiate the next transmission sequence.

11.4.3 General description

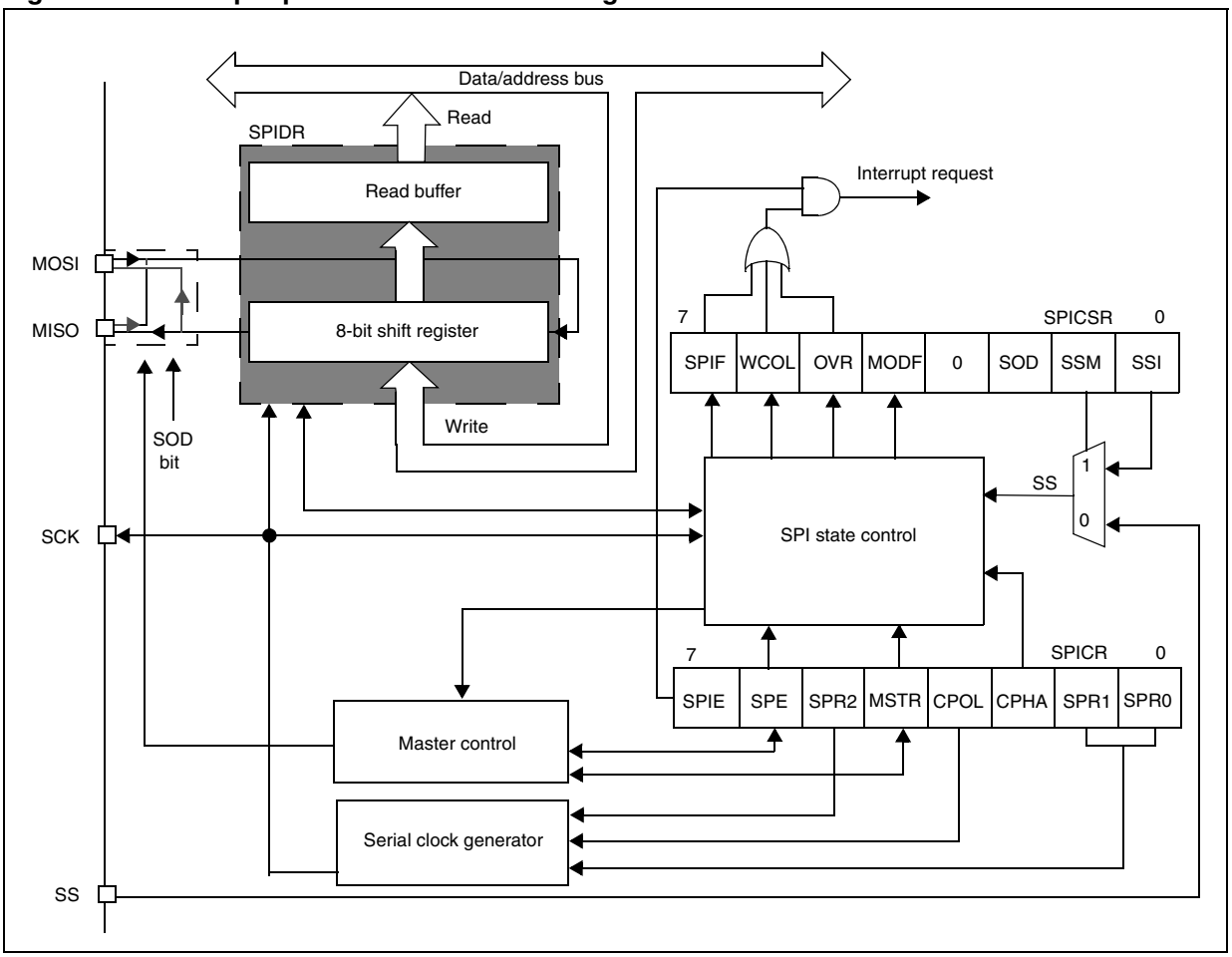
Figure 49: Serial peripheral interface block diagram on page 109 shows the serial peripheral interface (SPI) block diagram. There are three registers:

- SPI control register (SPICR)
- SPI control/status register (SPICSR)
- SPI data register (SPIDR)

The SPI is connected to external devices through four pins:

- MISO: master in/slave out data
- MOSI: master out/slave In data
- SCK: Serial clock out by SPI masters and input by SPI slaves
- $\overline{SS}$: Slave select: This input signal acts as a ‘chip select’ to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave $\overline{SS}$ inputs can be driven by standard I/O ports on the master device.

Figure 49. Serial peripheral interface block diagram



Functional description

A basic example of interconnections between a single master and a single slave is illustrated in *Figure 50: Single master/single slave application on page 110*.

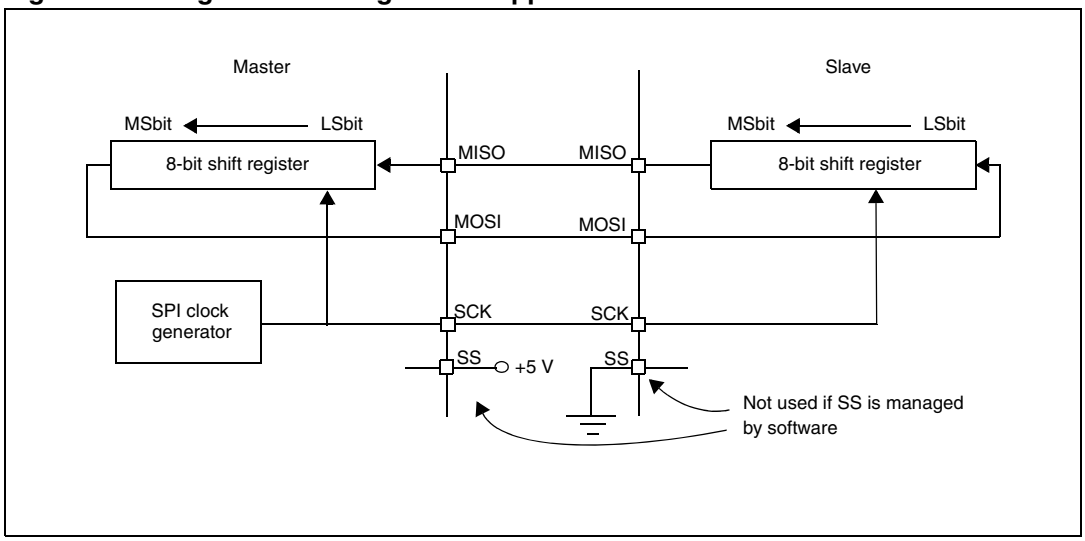
The MOSI pins are connected together and the MISO pins are connected together. In this way data are transferred serially between master and slave (most significant bit first).

The communication is always initiated by the master. When the master device transmits data to a slave device via MOSI pin, the slave device responds by sending data to the master device via the MISO pin. This implies full duplex communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

To use a single data line, the MISO and MOSI pins must be connected at each node (in this case only simplex communication is possible).

Four possible data/clock timing relationships may be chosen (see *Figure 53: Data clock timing diagram on page 114*) but master and slave must be programmed with the same timing mode.

Figure 50. Single master/single slave application



Slave select management

As an alternative to using the $\overline{SS}$ pin to control the slave select signal, the application can choose to manage the slave select signal by software. This is configured by the SSM bit in the SPICSR register (see *Figure 52: Hardware/software slave select management on page 111*).

In software management, the external $\overline{SS}$ pin is free for other application uses and the internal $\overline{SS}$ signal level is driven by writing to the SSI bit in the SPICSR register.

In master mode:

– $\overline{SS}$ internal must be held high continuously

In slave mode:

There are two cases depending on the data/clock timing relationship (see *Figure 51: Generic SS timing diagram on page 111*):

If CPHA = 1 (data latched on second clock edge):

– $\overline{SS}$ internal must be held low during the entire transmission. This implies that in single slave applications the $\overline{SS}$ pin either can be tied to V_{SS}, or made free for standard I/O by managing the $\overline{SS}$ function by software (SSM = 1 and SSI = 0 in the in the SPICSR register)

If CPHA = 0 (data latched on first clock edge):

– $\overline{SS}$ internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If $\overline{SS}$ is not pulled high, a write collision error occurs when the slave writes to the shift register (see *Write collision error (WCOL) on page 115*).

Figure 51. Generic $\overline{SS}$ timing diagram

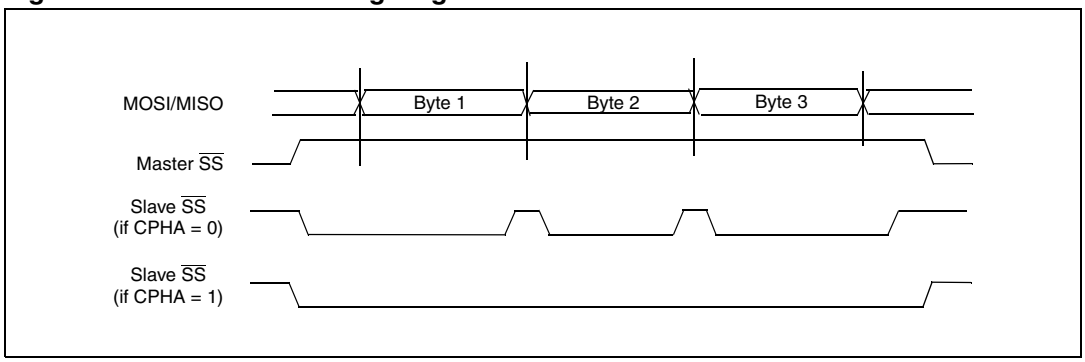
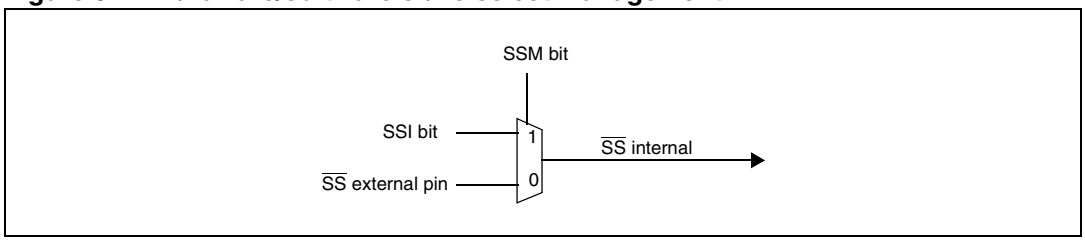


Figure 52. Hardware/software slave select management



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 53: Data clock timing diagram on page 114* 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 $\overline{SS}$ pin high for the complete byte transmit sequence.
3. Write to the SPICR register:
   - Set the MSTR and SPE bits

Note: 1 MSTR and SPE bits remain set only if $\overline{SS}$ is high).

2 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.

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.

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 53: Data clock timing diagram on page 114*).

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

   - Manage the $\overline{SS}$ pin as described in *Slave select management on page 110* and *Figure 51: Generic SS timing diagram on page 111*. If CPHA = 1 $\overline{SS}$ must be held low continuously. If CPHA = 0 $\overline{SS}$ 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.

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

Note: 1 While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

2 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 Overrun condition (OVR) on page 115).

11.4.4 Clock phase and clock polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (see *Figure 53: Data clock timing diagram on page 114*).

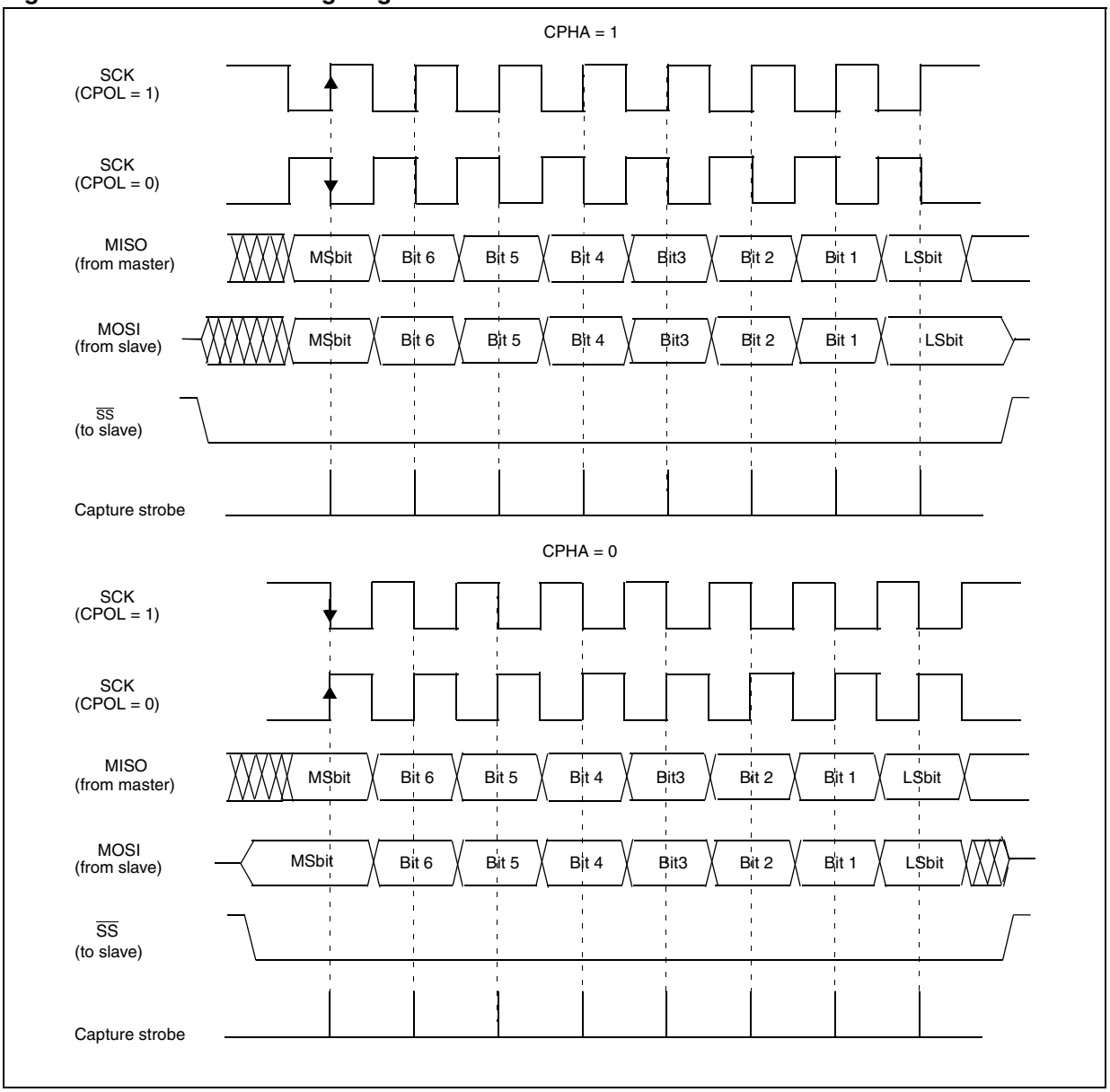
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL = 1 or pulling down SCK if CPOL = 0).

The combination of the CPOL clock polarity and CPHA (clock phase) bits selects the data capture clock edge.

Figure 53: Data clock timing diagram on page 114 shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin, the MISO pin and the MOSI pin are directly connected between the master and the slave device.

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

Figure 53. Data clock timing diagram



1. This figure should not be used as a replacement for parametric information. Refer to *Section 13: Electrical characteristics*.

11.4.5 Error flags

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.

Note: 1 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.

2 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.

3 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.

4 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.

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.

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 *Slave select management on page 110*.

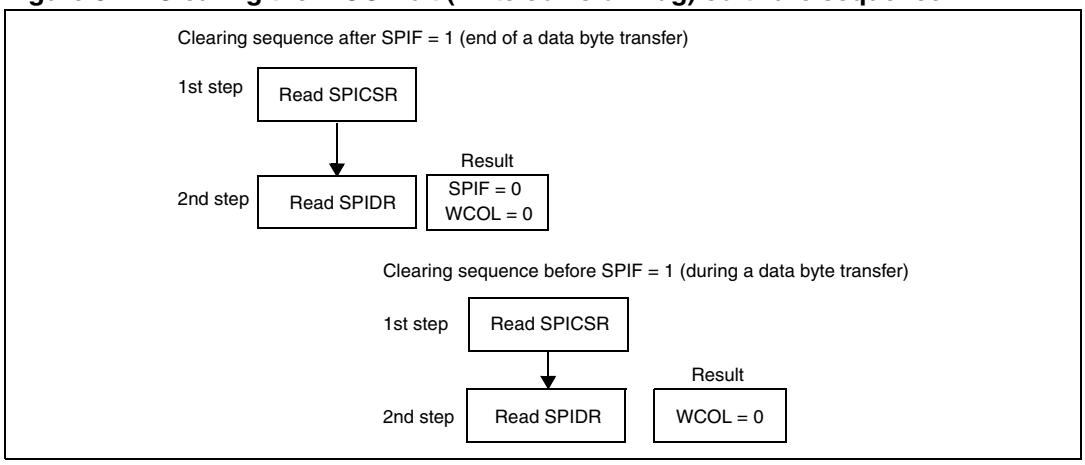
Note: A 'read collision' will never occur since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

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 54*).

Figure 54. Clearing the WCOL bit (write collision flag) software sequence



1. Writing to the SPIDR register instead of reading it does not reset the WCOL bit.

Single master and multimaster configurations

There are two types of SPI systems:

- Single master system
- Multimaster system

Single Master System

A typical single master system may be configured using a device as the master and four devices as slaves (see *Figure 55: Single master/multiple slave configuration on page 117*).

The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices.

The $\overline{SS}$ pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices.

Note: To prevent a bus conflict on the MISO line, the master allows only one active slave device during a transmission.

For more security, the slave device may respond to the master with the received data byte. Then the master will receive the previous byte back from the slave device if all MISO and MOSI pins are connected and the slave has not written to its SPIDR register.

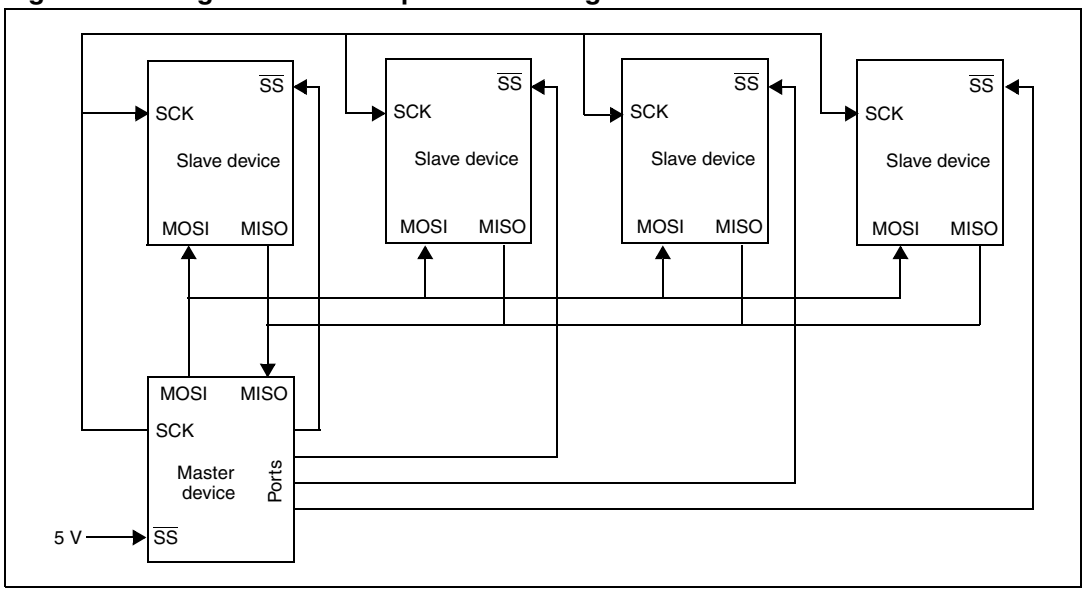
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

Multimaster system

A multimaster system may also be configured by the user. Transfer of master control could be implemented using a handshake method through the I/O ports or by an exchange of code messages through the serial peripheral interface system.

The multimaster system is principally handled by the MSTR bit in the SPICR register and the MODF bit in the SPICSR register.

Figure 55. Single master/multiple slave configuration



11.4.6 Low power modes

Table 56. Effect of low power modes on SPI

Mode	Description
Wait	No effect on SPI SPI interrupt events cause the device to exit from wait mode.
Halt	SPI registers are frozen In halt mode, the SPI is inactive. SPI operation resumes when the device is woken up by an interrupt with 'exit from Halt mode' capability. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetching). If several data are received before the wakeup event, then an overrun error is generated. This error can be detected after the fetch of the interrupt routine that woke up the device.

Using the SPI to wake up the device from halt mode

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

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

Caution: The SPI can wake up the device from halt mode only if the slave select signal (external $\overline{SS}$ pin or the SSI bit in the SPICSR register) is low when the device enters halt mode. So, if slave selection is configured as external (see *Slave select management on page 110*), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters halt mode.

11.4.7 Interrupts

Table 57. SPI interrupt control/wake-up capability(1)

Interrupt event	Event flag	Enable control bit	Exit from wait	Exit from halt
SPI end of transfer event	SPIF	SPIE	Yes	Yes
Master mode fault event	MODF			No
Overrun error	OVR			

1. The SPI interrupt events are connected to the same interrupt vector (see *Section 8: Interrupts*). They generate an interrupt if the corresponding enable control bit is set and the interrupt mask in the CC register is reset (RIM instruction).

11.4.8 Register description

SPI control register (SPICR)

SPICR Reset value: 0000 xxxx (0xh)

7	6	5	4	3	2	1	0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Table 58. SPICR register description

Bit	Bit name	Function
7	SPIE	*S*erial 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)
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 *Master mode fault (MODF) on page 115*). 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
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 (see bits [1:0] below). 0: Divider by 2 enabled 1: Divider by 2 disabled *Note: The SPR2 bit has no effect in slave mode*

Table 58. SPICR register description (continued)

Bit	Bit name	Function
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 *Master mode fault (MODF) on page 115*). 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.
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.*
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.*
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: 100: serial clock = $f_{CPU}/4$ 000: serial clock = $f_{CPU}/8$ 001: serial clock = $f_{CPU}/16$ 110: serial clock = $f_{CPU}/32$ 010: serial clock = $f_{CPU}/64$ 011: serial clock = $f_{CPU}/128$ *Note: These 2 bits have no effect in slave mode.*

SPI control/status register (SPICSR)

SPICSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SPIF	WCOL	OVR	MODF	Reserved	SOD	SSM	SSI
R	R	R	R	-	R/W	R/W	R/W

Table 59. SPICSR register description

Bit	Bit name	Function
7	SPIF	Serial peripheral data transfer flag 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.*
6	WCOL	Write collision status This bit is set by hardware when a write to the SPIDR register is made during a transmit sequence. It is cleared by a software sequence (see *Figure 54: Clearing the WCOL bit (write collision flag) software sequence on page 116*). 0: No write collision occurred 1: A write collision has been detected
5	OVR	SPI overrun error 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 *Overrun condition (OVR) on page 115*). An interrupt is generated if SPIE = 1 in SPICR register. The OVR bit is cleared by software reading the SPICSR register. 0: No overrun error 1: Overrun error detected
4	MODF	Mode fault flag This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see *Master mode fault (MODF) on page 115*). 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
3	-	Reserved, must be kept cleared.

Table 59. SPICSR register description (continued)

Bit	Bit name	Function
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
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 *Slave select management on page 110*. 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)
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)

SPIDR — Reset value: undefined

7	6	5	4	3	2	1	0
D[7:0]							
R/W							

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

Note: 1 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.

2 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 49: Serial peripheral interface block diagram on page 109*).

Table 60. SPI register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0031h	SPIDR Reset Value	MSB x	x	x	x	x	x	x	LSB x
0032h	SPICR Reset Value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
0033h	SPICSR Reset Value	SPIF 0	WCOL 0	OVR 0	MODF 0	0	SOD 0	SSM 0	SSI 0

11.5 LINSCI serial communication interface (LIN master/slave)

11.5.1 Introduction

The serial communications interface (SCI) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The SCI offers a very wide range of baud rates using two baud rate generator systems.

The LIN-dedicated features support the LIN (local interconnect network) protocol for both master and slave nodes.

This chapter is divided into SCI Mode and LIN mode sections. For information on general SCI communications, refer to the SCI mode section. For LIN applications, refer to both the SCI mode and LIN mode sections.

11.5.2 SCI features

- Full duplex, asynchronous communications
- NRZ standard format (mark/space)
- Independently programmable transmit and receive baud rates up to 500 K baud
- Programmable data word length (8 or 9 bits)
- Receive buffer full, transmit buffer empty and end of transmission flags
- 2 receiver wake-up modes:
  - Address bit (MSB)
  - Idle line
- Muting function for multiprocessor configurations
- Separate enable bits for transmitter and receiver
- Overrun, noise and frame error detection
- 6 interrupt sources
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error
  - Parity interrupt
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Reduced power consumption mode

11.5.3 LIN features

- LIN master
  - 13-bit LIN synch break generation
- LIN slave
  - Automatic header handling
  - Automatic baud rate resynchronization based on recognition and measurement of the LIN synch field (for LIN slave nodes)
  - Automatic baud rate adjustment (at CPU frequency precision)
  - 11-bit LIN synch break detection capability
  - LIN Parity check on the LIN identifier field (only in reception)
  - LIN error management
  - LIN header timeout
  - Hot plugging support

11.5.4 General description

The interface is externally connected to another device by two pins:

- TDO: Transmit data output. When the transmitter is disabled, the output pin returns to its I/O port configuration. When the transmitter is enabled and nothing is to be transmitted, the TDO pin is at high level.
- RDI: Receive data input is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

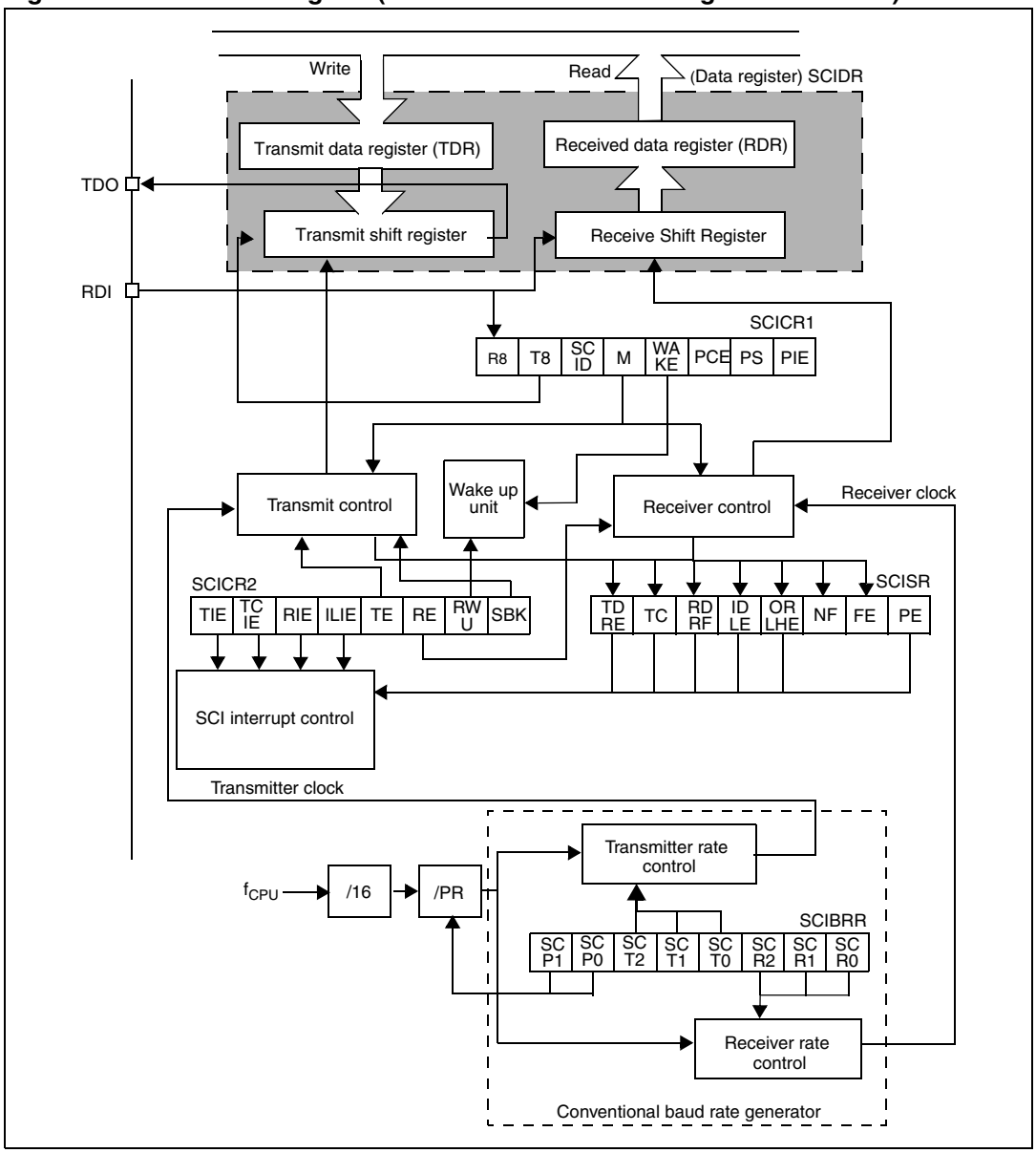
Through these pins, serial data is transmitted and received as characters comprising:

- An Idle line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) least significant bit first
- A stop bit indicating that the character is complete

This interface uses three types of baud rate generator:

- A conventional type for commonly-used baud rates
- An extended type with a prescaler offering a very wide range of baud rates even with non-standard oscillator frequencies
- A LIN baud rate generator with automatic resynchronization

Figure 56. SCI block diagram (in conventional baud rate generator mode)



11.5.5 SCI mode - functional description

Conventional baud rate generator mode

The block diagram of the serial control interface in conventional baud rate generator mode is shown in *Figure 56*.

It uses four registers:

- 2 control registers (SCICR1 and SCICR2)
- A status register (SCISR)
- A baud rate register (SCIBRR)

Extended prescaler mode

Two additional prescalers are available in extended prescaler mode. They are shown in *Figure 58: SCI baud rate and extended prescaler block diagram on page 131*.

- An extended prescaler receiver register (SCIERPR)
- An extended prescaler transmitter register (SCIETPR)

Serial data format

Word length may be selected as being either 8 or 9 bits by programming the M bit in the SCICR1 register (see *Figure 57*).

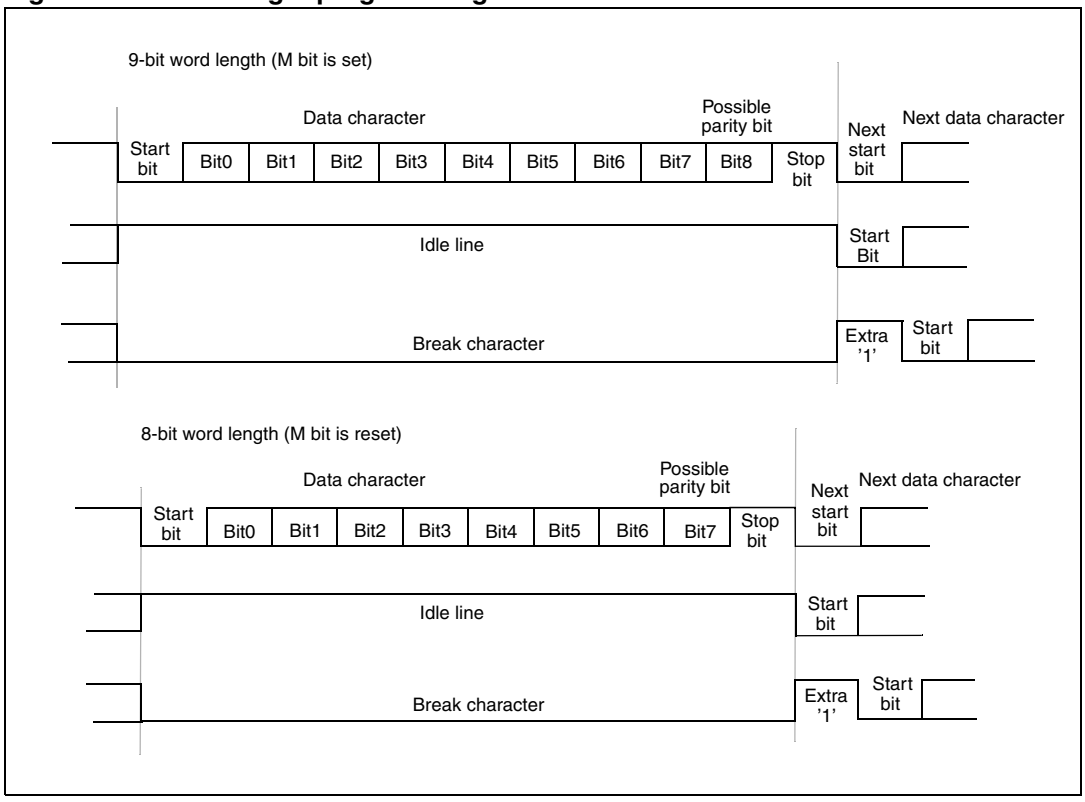
The TDO pin is in low state during the start bit.

The TDO pin is in high state during the stop bit.

An idle character is interpreted as a continuous logic high level for 10 (or 11) full bit times.

A break character is a character with a sufficient number of low level bits to break the normal data format followed by an extra “1” bit to acknowledge the start bit.

Figure 57. Word length programming



Transmitter

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

Character transmission

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the SCIDR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see *Figure 56*).

Procedure

- Select the M bit to define the word length.
- Select the desired baud rate using the SCIBRR and the SCIETPR registers.
- Set the TE bit to send a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones (idle line) as first transmission.
- Access the SCISR register and write the data to send in the SCIDR register (this sequence clears the TDRE bit). Repeat this sequence for each data to be transmitted.

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

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

The TDRE bit is set by hardware and it indicates:

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

This flag generates an interrupt if the TIE bit is set and the I[|1:0] bits are cleared in the CCR register.

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

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

When a character transmission is complete (after the stop bit) the TC bit is set and an interrupt is generated if the TCIE is set and the I[1:0] bits are cleared in the CCR register.

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

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

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

Break characters

Setting the SBK bit loads the shift register with a break character. The break character length depends on the M bit (see *Figure 57: Word length programming on page 126*).

As long as the SBK bit is set, the SCI sends break characters to the TDO pin. After clearing this bit by software, the SCI inserts a logic 1 bit at the end of the last break character to guarantee the recognition of the start bit of the next character.

Idle line

Setting the TE bit drives the SCI to send a preamble of 10 (M = 0) or 11 (M = 1) consecutive ‘1’s (idle line) before the first character.

In this case, clearing and then setting the TE bit during a transmission sends a preamble (idle line) after the current word. Note that the preamble duration (10 or 11 consecutive '1's depending on the M bit) does not take into account the stop bit of the previous character.

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

Receiver

The SCI can receive data words of either 8 or 9 bits. When the M bit is set, word length is 9 bits and the MSB is stored in the R8 bit in the SCICR1 register.

Character reception

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, the SCIDR register consists or a buffer (RDR) between the internal bus and the received shift register (see *Figure 56: SCI block diagram (in conventional baud rate generator mode) on page 125*).

Procedure

- Select the M bit to define the word length.
- Select the desired baud rate using the SCIBRR and the SCIERPR registers.
- Set the RE bit, this enables the receiver which begins searching for a start bit.

When a character is received:

- The RDRF bit is set. It indicates that the content of the shift register is transferred to the RDR
- An interrupt is generated if the RIE bit is set and the I[1:0] bits are cleared in the CCR register
- The error flags can be set if a frame error, noise or an overrun error has been detected during reception

Clearing the RDRF bit is performed by the following software sequence done by:

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

The RDRF bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Idle line

When an idle line is detected, there is the same procedure as a data received character plus an interrupt if the ILIE bit is set and the I[I1:0] bits are cleared in the CCR register.

Overrun error

An overrun error occurs when a character is received when RDRF has not been reset. Data can not be transferred from the shift register to the TDR register as long as the RDRF bit is not cleared.

When an overrun error occurs:

- The OR bit is set
- The RDR content will not be lost
- The shift register will be overwritten
- An interrupt is generated if the RIE bit is set and the I[|1:0] bits are cleared in the CCR register.

The OR bit is reset by an access to the SCISR register followed by a SCIDR register read operation.

Noise error

Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

When noise is detected in a character:

- The NF bit is set at the rising edge of the RDRF bit
- Data is transferred from the Shift register to the SCIDR register
- No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt

The NF bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

Framing error

A framing error is detected when:

- The stop bit is not recognized on reception at the expected time, following either a desynchronization or excessive noise.
- A break is received

When the framing error is detected:

- the FE bit is set by hardware
- Data is transferred from the shift register to the SCIDR register
- No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The FE bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

Break character

When a break character is received, the SCI handles it as a framing error. To differentiate a break character from a framing error, it is necessary to read the SCIDR. If the received value is 00h, it is a break character. Otherwise it is a framing error.

Conventional baud rate generation

The baud rates for the receiver and transmitter (Rx and Tx) are set independently and calculated as follows:

$$Tx = \frac{f_{CPU}}{(16*PR)*TR} \qquad Rx = \frac{f_{CPU}}{(16*PR)*RR}$$

where:

PR = 1, 3, 4 or 13 (see SCP[1:0] bits)

TR = 1, 2, 4, 8, 16, 32, 64,128 (see SCT[2:0] bits)

RR = 1, 2, 4, 8, 16, 32, 64,128 (see SCR[2:0] bits)

All these bits are in the *Baud rate register (SCIBRR) on page 139*.

Example: If f_{CPU} is 8 MHz (normal mode) and if PR = 13 and TR = RR = 1, the transmit and receive baud rates are 38400 baud.

Note: The baud rate registers MUST NOT be changed while the transmitter or the receiver is enabled.

Extended baud rate generation

The extended prescaler option gives a very fine tuning on the baud rate, using a 255 value prescaler, whereas the conventional baud rate generator retains industry standard software compatibility.

The extended baud rate generator block diagram is described in *Figure 58: SCI baud rate and extended prescaler block diagram on page 131*.

The output clock rate sent to the transmitter or to the receiver will be the output from the 16 divider divided by a factor ranging from 1 to 255 set in the SCIERPR or the SCIETPR register.

Note: The extended prescaler is activated by setting the SCIETPR or SCIERPR register to a value other than zero.

The baud rates are calculated as follows:

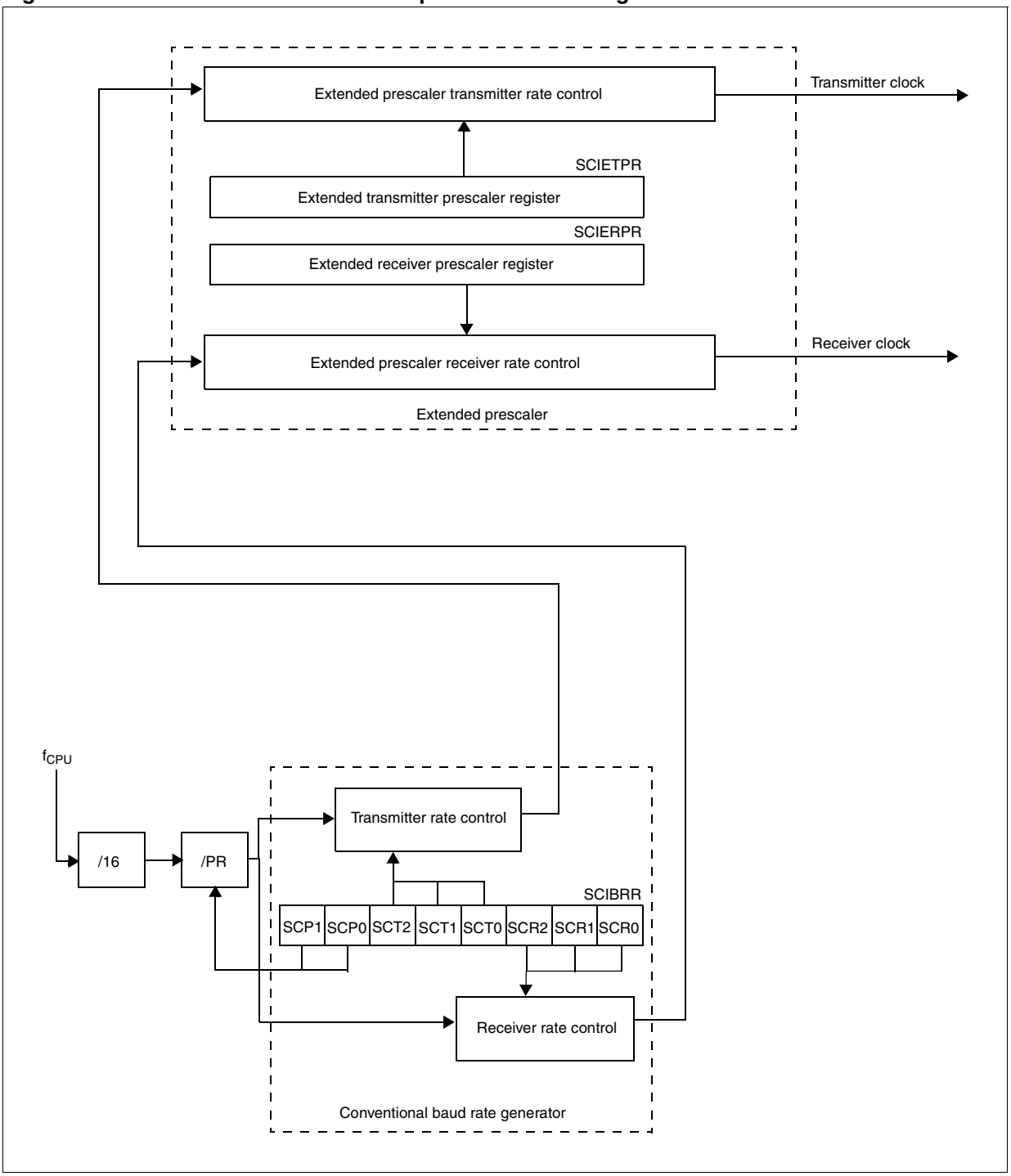
$$Tx = \frac{f_{CPU}}{16*ETPR*(PR*TR)} \qquad Rx = \frac{f_{CPU}}{16*ERPR*(PR*RR)}$$

where:

ETPR = 1, ..., 255 (see SCIETPR register *on page 140*)

ERPR = 1, ..., 255 (see SCIERPR register *on page 140*)

Figure 58. SCI baud rate and extended prescaler block diagram



Receiver muting and wake-up feature

In multiprocessor configurations it is often desirable that only the intended message recipient should actively receive the full message contents, thus reducing redundant SCI service overhead for all non-addressed receivers.

The non-addressed devices may be placed in sleep mode by means of the muting function.

Setting the RWU bit by software puts the SCI in sleep mode:

All the reception status bits cannot be set.

All the receive interrupts are inhibited.

A muted receiver may be woken up in one of the following ways:

- by idle line detection if the WAKE bit is reset,
- by address mark detection if the WAKE bit is set.

Idle line detection

The receiver wakes up by idle line detection when the receive line has recognized an idle line. Then the RWU bit is reset by hardware but the IDLE bit is not set.

This feature is useful in a multiprocessor system when the first characters of the message determine the address and when each message ends by an idle line: As soon as the line becomes idle, every receiver is awakened and the first characters of the message are analysed which indicates the addressed receiver. The receivers which are not addressed set the RWU bit to enter in mute mode. Consequently, they will not read the next characters constituting the next part of the message. At the end of the message, an idle line is sent by the transmitter: this wakes up every receiver which are ready to analyse the addressing characters of the new message.

In such a system, the inter-characters space must be smaller than the idle time.

Address mark detection

The receiver wakes up by address mark detection when it receives a ‘1’ as the most significant bit of a word, thus indicating that the message is an address. The reception of this particular word wakes up the receiver, resets the RWU bit and sets the RDRF bit, which allows the receiver to receive this word normally and to use it as an address word.

This feature is useful in a multiprocessor system when the most significant bit of each character (except for the break character) is reserved for address detection. As soon as the receivers receive an address character (most significant bit = ’1’), the receivers are woken up. The receivers which are not addressed set RWU bit to enter in mute mode. Consequently, they will not treat the next characters constituting the next part of the message.

Parity control

Hardware byte parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the SCICR1 register. Depending on the character format defined by the M bit, the possible SCI character formats are as listed in *Table 61*.

Note: In case of wake-up by an address mark, the MSB bit of the data is taken into account and not the parity bit

Table 61. Character formats(1)

M bit	PCE bit	Character format
0	0	I SB I 8 bit data I STB I
	1	I SB I 7-bit data I PB I STB I
1	0	I SB I 9-bit data I STB I
	1	I SB I 8-bit data I PB I STB I

1. Legend: SB = start bit, STB = stop bit, PB = parity bit

Even parity

The parity bit is calculated to obtain an even number of ‘1s’ inside the character made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit will be 0 if even parity is selected (PS bit = 0).

Odd parity

The parity bit is calculated to obtain an odd number of ‘1s’ inside the character made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit will be 1 if odd parity is selected (PS bit = 1).

Transmission mode

If the PCE bit is set then the MSB bit of the data written in the data register is not transmitted but is changed by the parity bit.

Reception mode

If the PCE bit is set then the interface checks if the received data byte has an even number of ‘1s’ if even parity is selected (PS = 0) or an odd number of ‘1s’ if odd parity is selected (PS = 1). If the parity check fails, the PE flag is set in the SCISR register and an interrupt is generated if PCIE is set in the SCICR1 register.

11.5.6 Low power modes

Table 62. Effect of low power modes on SCI

Mode	Description
Wait	No effect on SCI SCI interrupts cause the device to exit from wait mode
Halt	SCI registers are frozen In halt mode, the SCI stops transmitting/receiving until halt mode is exited

11.5.7 Interrupts

Table 63. SCI interrupt control/wake-up capability

Interrupt event	Event flag	Enable control bit	Exit from wait	Exit from halt
Transmit data register empty	TDRE	TIE	Yes	No
Transmission complete	TC	TCIE		
Received data ready to be read	RDRF	RIE		
Overrun error or LIN synch error detected	OR/LHE			
Idle line detected	IDLE	ILIE		
Parity error	PE	PIE		
LIN header detection	LHDF	LHIE		

The SCI interrupt events are connected to the same interrupt vector (see *Section 8: Interrupts*).

These events generate an interrupt if the corresponding enable control bit is set and the interrupt mask in the CC register is reset (RIM instruction).

11.5.8 SCI mode registers

Status register (SCISR)

SCISR — Reset value: 1100 0000 (C0h)

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	OR(1)	NF(1)	FE(1)	PE(1)
R	R	R	R	R	R	R	R

1. This bit has a different function in LIN mode, please refer to the LIN mode register description

Table 64. SCISR register description

Bit	Bit name	Function
7	TDRE	Transmit data register empty This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TIE bit = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register). 0: Data is not transferred to the shift register 1: Data is transferred to the shift register
6	TC	Transmission complete This bit is set by hardware when transmission of a character containing data is complete. An interrupt is generated if TCIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register). 0: Transmission is not complete 1: Transmission is complete *Note: TC is not set after the transmission of a preamble or a break.*
5	RDRF	Received data ready flag This bit is set by hardware when the content of the RDR register has been transferred to the SCIDR register. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register). 0: Data are not received 1: Received data are ready to be read

Table 64. SCISR register description (continued)

Bit	Bit name	Function
4	IDLE	Idle line detect This bit is set by hardware when an idle line is detected. An interrupt is generated if the ILIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register). 0: No idle line is detected 1: Idle line is detected *Note: The IDLE bit is not set again until the RDRF bit is set (i.e. a new idle line occurs).*
3	OR	Overrun error This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RDRF = 1. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register). 0: No overrun error 1: Overrun error is detected *Note: When the IDLE bit is set the RDR register content is not lost but the shift register is overwritten.*
2	NF	Noise flag This bit is set by hardware when noise is detected on a received character. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register). 0: No noise is detected 1: Noise is detected *Note: The NF bit does not generate an interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt.*
1	FE	Framing error This bit is set by hardware when a de-synchronization, excessive noise or a break character is detected. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register). 0: No framing error is detected 1: Framing error or break character is detected *Note: The FE bit does not generate an interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt. If the word currently being transferred causes both frame error and overrun error, it is transferred and only the OR bit is set.*
0	PE	Parity error This bit is set by hardware when a 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 parity error 1: Parity error

Control register 1 (SCICR1)

SCICR1 Reset value: x000 0000 (x0h)

7	6	5	4	3	2	1	0
R8	T8	SCID	M	WAKE	PCE(1)	PS	PIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

1. This bit has a different function in LIN mode, please refer to the LIN mode register description

Table 65. SCICR1 register description

Bit	Bit name	Function
7	R8	Receive data bit 8 This bit is used to store the 9th bit of the received word when M = 1.
6	T8	Transmit data bit 8 This bit is used to store the 9th bit of the transmitted word when M = 1.
5	SCID	Disabled for low power consumption When this bit is set the SCI prescalers and outputs are stopped at the end of the current byte transfer in order to reduce power consumption.This bit is set and cleared by software. 0: SCI enabled 1: SCI prescaler and outputs disabled
4	M	Word length This bit determines the word length. It is set or cleared by software. 0: 1 Start bit, 8 data bits, 1 stop bit 1: 1 Start bit, 9 data bits, 1 stop bit *Note: The M bit must not be modified during a data transfer (both transmission and reception).*
3	WAKE	Wake up method This bit determines the SCI wake up method, it is set or cleared by software. 0: Idle line 1: Address mark *Note: If the LINE bit is set, the WAKE bit is deactivated and replaced by the LHDM bit.*
2	PCE	Parity control enable This bit is set and cleared by software. It selects the hardware parity control (generation and detection for byte parity, detection only for LIN parity). 0: Parity control disabled 1: Parity control enabled

Table 65. SCICR1 register description (continued)

Bit	Bit name	Function
1	PS	Parity selection This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity is selected after the current byte. 0: Even parity 1: Odd parity
0	PIE	Parity interrupt enable This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). The parity error involved can be a byte parity error (if bit PCE is set and bit LPE is reset) or a LIN parity error (if bit PCE is set and bit LPE is set). 0: Parity error interrupt disabled 1: Parity error interrupt enabled

Control register 2 (SCICR2)

SCICR2 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU(1)	SBK(1)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

1. This bit has a different function in LIN mode, please refer to the LIN mode register description

Table 66. SCICR2 register description

Bit	Bit name	Function
7	TIE	Transmitter interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever TDRE = 1 in the SCISR register
6	TCIE	Transmission complete interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever TC = 1 in the SCISR register
5	RIE	Receiver interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever OR = 1 or RDRF = 1 in the SCISR register
4	ILIE	Idle line interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever IDLE = 1 in the SCISR register

Table 66. SCICR2 register description (continued)

Bit	Bit name	Function
3	TE	Transmitter enable This bit enables the transmitter. It is set and cleared by software. 0: Transmitter is disabled 1: Transmitter is enabled *Note: During transmission, an '0' pulse on the TE bit ('0' followed by '1') sends a preamble (idle line) after the current word.* *When TE is set there is a 1 bit-time delay before the transmission starts.*
2	RE	Receiver enable This bit enables the receiver. It is set and cleared by software. 0: Receiver is disabled in the SCISR register 1: Receiver is enabled and begins searching for a start 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 *Note: Before selecting mute mode (by setting the RWU bit), the SCI must first receive a data byte, otherwise it cannot function in mute mode with wakeup by idle line detection.* *In address mark detection wake up configuration (WAKE bit = 1) the RWU bit cannot be modified by software while the RDRF bit is set.*
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 sends a BREAK word at the end of the current word.*

Data register (SCIDR)

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

SCIDR — Reset value: undefined

7	6	5	4	3	2	1	0
DR[7:0]							
R/W							

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

The TDR register provides the parallel interface between the internal bus and the output shift register (see *Figure 56: SCI block diagram (in conventional baud rate generator mode) on page 125*).

The RDR register provides the parallel interface between the input shift register and the internal bus (see *Figure 56*).

Baud rate register (SCIBRR)

SCIBRR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SCP[1:0]		SCT[2:0]			SCR[2:0]		
R/W		R/W			R/W		

Note: When LIN slave mode is disabled, the SCIBRR register controls the conventional baud rate generator.

Table 67. SCIBRR register description

Bit	Bit name	Function
7:6	SCP[1:0]	First SCI prescaler These 2 prescaling bits allow several standard clock division ranges: 00: PR prescaling factor = 1 01: PR prescaling factor = 3 10: PR prescaling factor = 4 11: PR prescaling factor = 13
5:3	SCT[2:0]	SCI transmitter rate divisor These 3 bits, in conjunction with the SCP1 and SCP0 bits define the total division applied to the bus clock to yield the transmit rate clock in conventional baud rate generator mode: 000: TR dividing factor = 1 001: TR dividing factor = 2 010: TR dividing factor = 4 011: TR dividing factor = 8 100: TR dividing factor = 16 101: TR dividing factor = 32 110: TR dividing factor = 64 111: TR dividing factor = 128
2:0	SCR[2:0]	SCI receiver rate divider These 3 bits, in conjunction with the SCP[1:0] bits define the total division applied to the bus clock to yield the receive rate clock in conventional baud rate generator mode: 000: RR dividing factor = 1 001: RR dividing factor = 2 010: RR dividing factor = 4 011: RR dividing factor = 8 100: RR dividing factor = 16 101: RR dividing factor = 32 110: RR dividing factor = 64 111: RR dividing factor = 128

Extended receive prescaler division register (SCIERPR)

SCIERPR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ERPR[7:0]							
R/W							

Table 68. SCIERPR register description

Bit	Bit name	Function
7:0	ERPR[7:0]	8-bit extended receive prescaler register The extended baud rate generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see *Figure 58: SCI baud rate and extended prescaler block diagram on page 131*) is divided by the binary factor set in the SCIERPR register (in the range 1 to 255). The extended baud rate generator is not active after a reset.

Extended transmit prescaler division register (SCIETPR)

SCIETPR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ETPR[7:0]							
R/W							

Table 69. SCIETPR register description

Bit	Bit name	Function
7:0	ETPR[7:0]	8-bit extended transmit prescaler register The extended baud rate generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see *Figure 58: SCI baud rate and extended prescaler block diagram on page 131*) is divided by the binary factor set in the SCIETPR register (in the range 1 to 255). The extended baud rate generator is not used after a reset. *Note: In LIN slave mode, the conventional and extended baud rate generators are disabled.*

11.5.9 LIN mode - functional description.

The block diagram of the serial control interface, in LIN slave mode is shown in *Figure 60: SCI block diagram in LIN slave mode on page 143*.

It uses six registers:

- 3 control registers: SCICR1, SCICR2 and SCICR3
- 2 status registers: SCISR register and LHLR register mapped at the SCIERPR address
- A baud rate register: LPR mapped at the SCIBRR address and an associated fraction register LPFR mapped at the SCIETPR address

The bits dedicated to LIN are located in the SCICR3. Refer to the register descriptions in *Section 11.5.10: LIN mode register description* for the definitions of each bit.

Entering LIN mode

To use the LINSCI in LIN mode the following configuration must be set in SCICR3 register:

- Clear the M bit to configure 8-bit word length.
- Set the LINE bit.

Master

To enter master mode the LSLV bit must be reset In this case, setting the SBK bit will send 13 low bits.

Then the baud rate can programmed using the SCIBRR, SCIERPR and SCIETPR registers.

In LIN master mode, the conventional and/or extended prescaler define the baud rate (as in standard SCI mode)

Slave

Set the LSLV bit in the SCICR3 register to enter LIN slave mode. In this case, setting the SBK bit will have no effect.

In LIN slave mode the LIN baud rate generator is selected instead of the conventional or extended prescaler. The LIN baud rate generator is common to the transmitter and the receiver.

Then the baud rate can be programmed using LPR and LPRF registers.

Note: It is mandatory to set the LIN configuration first before programming LPR and LPRF, because the LIN configuration uses a different baud rate generator from the standard one.

LIN transmission

In LIN mode the same procedure as in SCI mode has to be applied for a LIN transmission.

To transmit the LIN header the proceed as follows:

- First set the SBK bit in the SCICR2 register to start transmitting a 13-bit LIN synch break
- Reset the SBK bit
- Load the LIN synch field (0x55) in the SCIDR register to request synch field transmission
- Wait until the SCIDR is empty (TDRE bit set in the SCISR register)
- Load the LIN message Identifier in the SCIDR register to request Identifier transmission.

Figure 59. LIN characters

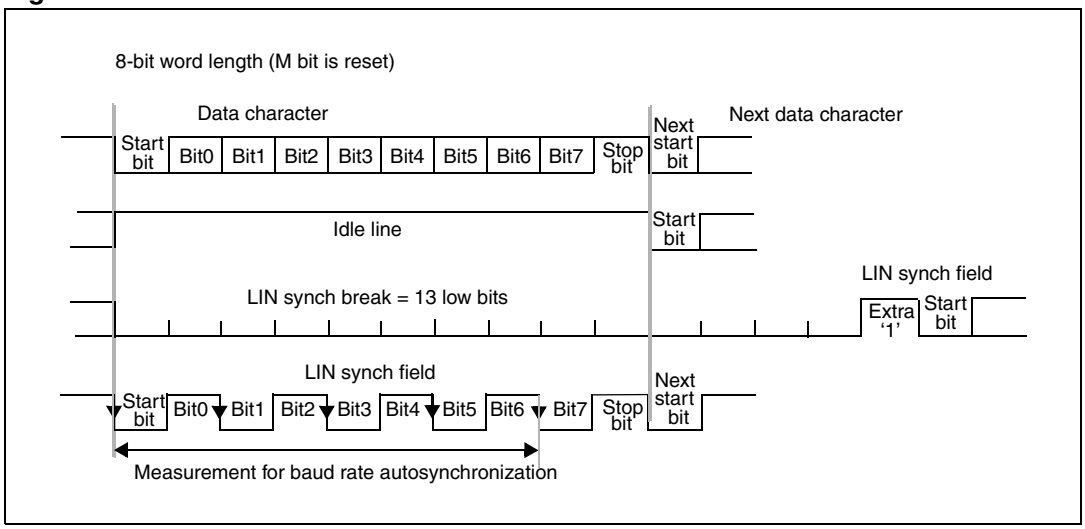
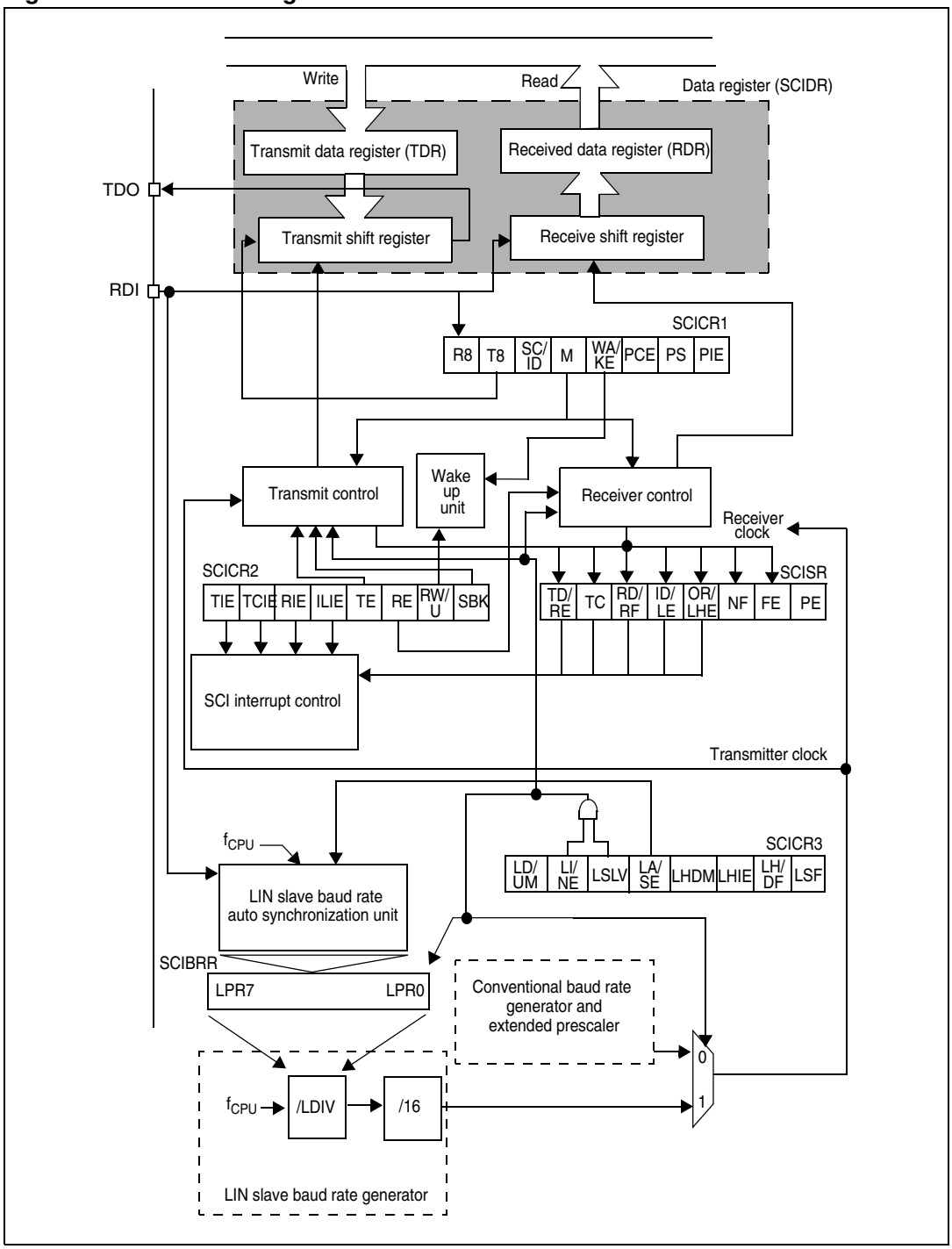


Figure 60. SCI block diagram in LIN slave mode



LIN reception

In LIN mode the reception of a byte is the same as in SCI mode but the LINSCI has features for handling the LIN header automatically (identifier detection) or semi-automatically (synch break detection) depending on the LIN header detection mode. The detection mode is selected by the LHDM bit in the SCICR3.

Additionally, an automatic resynchronization feature can be activated to compensate for any clock deviation, for more details please refer to *LIN baud rate on page 148*.

LIN header handling by a slave

Depending on the LIN header detection method the LINSCI will signal the detection of a LIN header after the LIN synch break or after the Identifier has been successfully received.

Note: 1 It is recommended to combine the header detection function with mute mode. Putting the LINSCI in mute mode allows the detection of headers only and prevents the reception of any other characters.

2 This mode can be used to wait for the next header without being interrupted by the data bytes of the current message in case this message is not relevant for the application.

Synch break detection (LHDM = 0)

When a LIN synch break is received:

- The RDRF bit in the SCISR register is set. It indicates that the content of the shift register is transferred to the SCIDR register, a value of 0x00 is expected for a break.
- The LHDF flag in the SCICR3 register indicates that a LIN synch break field has been detected.
- An interrupt is generated if the LHIE bit in the SCICR3 register is set and the I[1:0] bits are cleared in the CCR register.
- Then the LIN synch field is received and measured.
  - If automatic resynchronization is enabled (LASE bit = 1), the LIN synch field is not transferred to the shift register: There is no need to clear the RDRF bit.
  - If automatic resynchronization is disabled (LASE bit = 0), the LIN synch field is received as a normal character and transferred to the SCIDR register and RDRF is set.

Note: In LIN slave mode, the FE bit detects all frame error which does not correspond to a break.

Identifier detection (LHDM = 1):

This case is the same as the previous one except that the LHDF and the RDRF flags are set only after the entire header has been received (this is true whether automatic resynchronization is enabled or not). This indicates that the LIN Identifier is available in the SCIDR register.

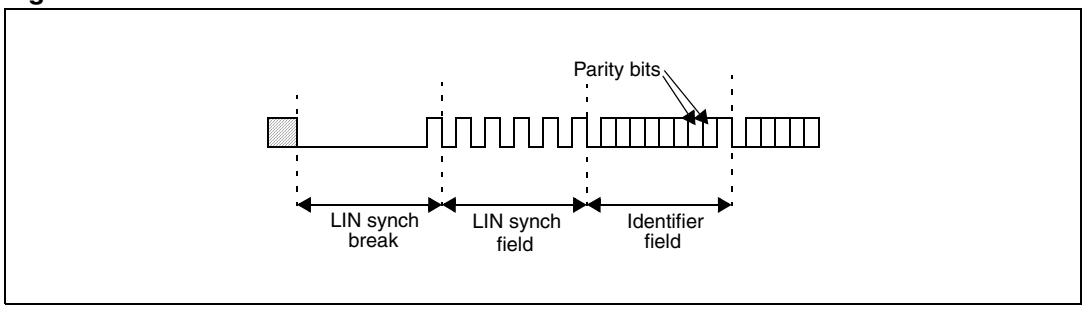
Note: During LIN synch field measurement, the SCI state machine is switched off and no characters are transferred to the data register.

LIN slave parity

In LIN slave mode (LINE and LSLV bits are set) LIN parity checking can be enabled by setting the PCE bit.

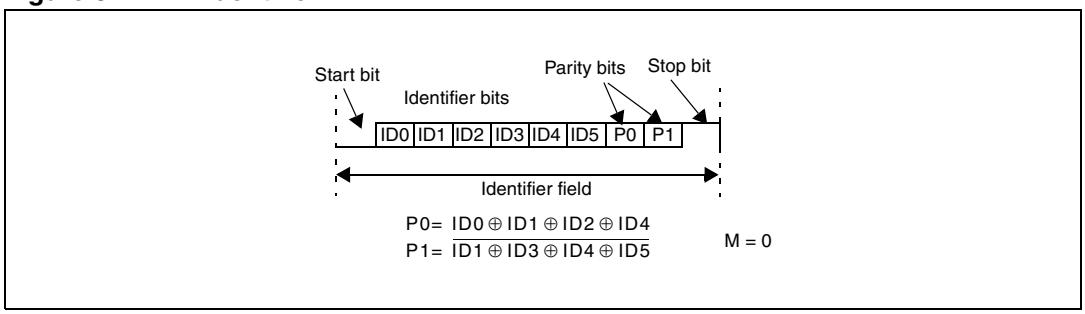
In this case, the parity bits of the LIN identifier field are checked. The identifier character is recognized as the third received character after a break character (included). See *Figure 61: LIN header on page 145*.

Figure 61. LIN header



The bits involved are the two MSB positions (7th and 8th bits if M = 0; 8th and 9th bits if M = 0) of the identifier character. The check is performed as specified in *Figure 62* by the LIN specification.

Figure 62. LIN identifier



LIN error detection

LIN header error flag

The LIN header error flag indicates that an invalid LIN header has been detected.

When a LIN header error occurs:

- The LHE flag is set
- An interrupt is generated if the RIE bit is set and the I[1:0] bits are cleared in the CCR register.

If autosynchronization is enabled (LASE bit = 1), this can mean that the LIN synch field is corrupted, and that the SCI is in a blocked state (LSF bit is set). The only way to recover is to reset the LSF bit and then to clear the LHE bit.

- The LHE bit is reset by an access to the SCISR register followed by a read of the SCIDR register.

LHE/OVR error conditions

When auto resynchronization is disabled (LASE bit = 0), the LHE flag detects:

- That the received LIN synch field is not equal to 55h.
- That an overrun occurred (as in standard SCI mode)
- Furthermore, if LHDM is set it also detects that a LIN header reception timeout occurred (only if LHDM is set).

When the LIN auto-resynchronization is enabled (LASE bit = 1), the LHE flag detects:

- That the deviation error on the synch field is outside the LIN specification which allows up to ± 15.5% of period deviation between the slave and master oscillators.
- A LIN header reception timeout occurred. If $T_{HEADER} > T_{HEADER\_MAX}$ then the LHE flag is set. Refer to *Figure 63* (only if LHDM is set to 1).
- An overflow during the synch field measurement, which leads to an overflow of the divider registers. If LHE is set due to this error then the SCI goes into a blocked state (LSF bit is set).
- That an overrun occurred on Fields other than the Synch Field (as in standard SCI mode)

Deviation error on the synch field

The deviation error is checked by comparing the current baud rate (relative to the slave oscillator) with the received LIN synch field (relative to the master oscillator). Two checks are performed in parallel:

- The first check is based on a measurement between the first falling edge and the last falling edge of the synch field. Let us refer to this period deviation as D:

If the LHE flag is set, it means that D > 15.625%

If LHE flag is not set, it means that D < 16.40625%

If 15.625% ≤ D < 16.40625%, then the flag can be either set or reset depending on the dephasing between the signal on the RDI line and the CPU clock.

- The second check is based on the measurement of each bit time between both edges of the synch field. This checks that each of these bit times is large enough compared to the bit time of the current baud rate.

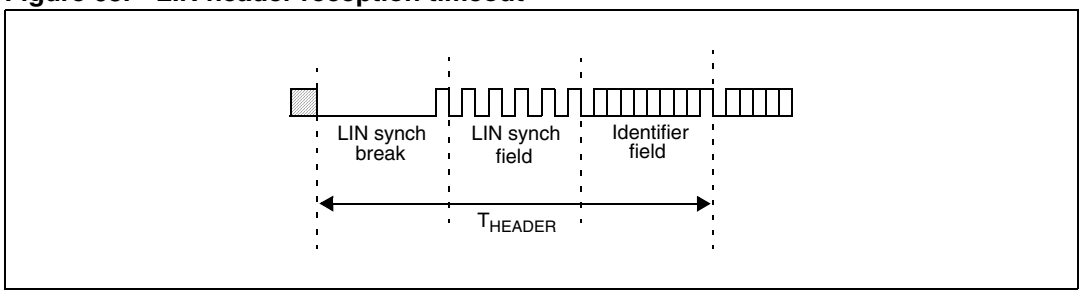
When LHE is set due to this error then the SCI goes into a blocked state (LSF bit is set).

LIN header time-out error

When the LIN identifier field detection method is used (by configuring LHDM to 1) or when LIN auto-resynchronization is enabled (LASE bit = 1), the LINSCI automatically monitors the T_{HEADER_MAX} condition given by the LIN protocol.

If the entire header (up to and including the STOP bit of the LIN identifier field) is not received within the maximum time limit of 57 bit times then a LIN header error is signalled and the LHE bit is set in the SCISR register.

Figure 63. LIN header reception timeout



The time-out counter is enabled at each break detection. It is stopped in the following conditions:

1. A LIN Identifier field has been received
2. An LHE error occurred (other than a timeout error)
3. A software reset of LSF bit (transition from high to low) occurred during the analysis of the LIN synch field
4. If LHE bit is set due to this error during the LIN synchr field (if LASE bit = 1) then the SCI goes into a blocked state (LSF bit is set)

If LHE bit is set due to this error during fields other than LIN synch field or if LASE bit is reset then the current received header is discarded and the SCI searches for a new break field.

Note on LIN header time-out limit

According to the LIN specification, the maximum length of a LIN header which does not cause a timeout is equal to 1.4 * (34 + 1) = 49 T_{BIT_MASTER}. T_{BIT_MASTER} refers to the master baud rate.

When checking this timeout, the slave node is desynchronized for the reception of the LIN break and synch fields. Consequently, a margin must be allowed, taking into account the worst case: This occurs when the LIN identifier lasts exactly 10 T_{BIT_MASTER} periods. In this case, the LIN break and synch fields lasts 49 - 10 = 39 T_{BIT_MASTER} periods.

Assuming the slave measures these first 39 bits with a desynchronized clock of 15.5%. This leads to a maximum allowed header length of: 39 x (1/0.845) T_{BIT_MASTER} + 10T_{BIT_MASTER} = 56.15 T_{BIT_SLAVE}. A margin is provided so that the time-out occurs when the header length is greater than 57 T_{BIT_SLAVE} periods. If it is less than or equal to 57 T_{BIT_SLAVE} periods, then no timeout occurs.

LIN header length

Even if no timeout occurs on the LIN header, it is possible to have access to the effective LIN header length (T_{HEADER}) through the LHL register. This allows monitoring at software level the T_{FRAME_MAX} condition given by the LIN protocol.

This feature is only available when LHDM bit = 1 or when LASE bit = 1.

Mute mode and errors

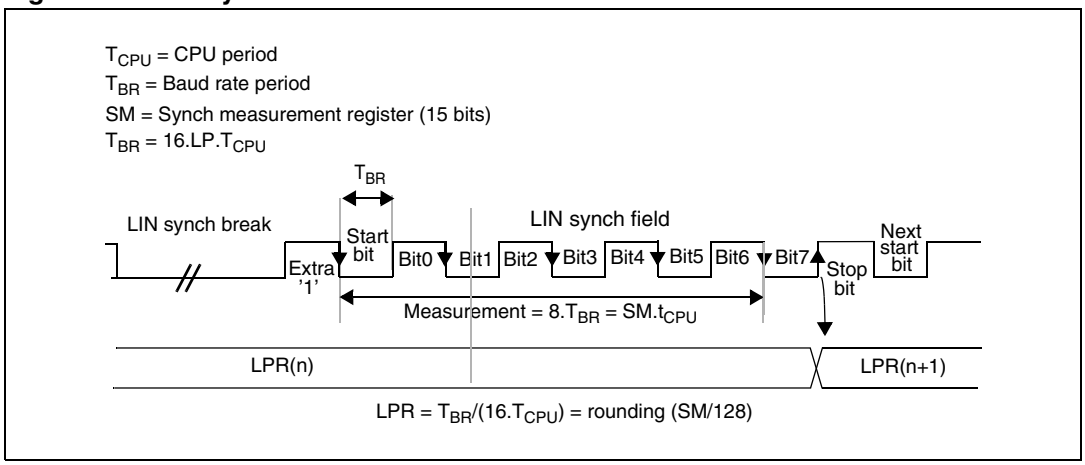
In mute mode when LHDM bit = 1, if an LHE error occurs during the analysis of the LIN synch field or if a LIN header time-out occurs then the LHE bit is set but it does not wake up from mute mode. In this case, the current header analysis is discarded. If needed, the software has to reset LSF bit. Then the SCI searches for a new LIN header.

In mute mode, if a framing error occurs on a data (which is not a break), it is discarded and the FE bit is not set.

When LHDM bit = 1, any LIN header which respects the following conditions causes a wake-up from mute mode:

- A valid LIN break field (at least 11 dominant bits followed by a recessive bit)
- A valid LIN synch field (without deviation error)
- A LIN identifier field without framing error. Note that a LIN parity error on the LIN identifier field does not prevent wake-up from mute mode.
- No LIN header time-out should occur during header reception.

Figure 64. LIN synch field measurement



LIN baud rate

Baud rate programming is done by writing a value in the LPR prescaler or performing an automatic resynchronization as described below.

Automatic resynchronization

To automatically adjust the baud rate based on measurement of the LIN synch field:

- Write the nominal LIN prescaler value (usually depending on the nominal baud rate) in the LPFR/LPR registers
- Set the LASE bit to enable the auto synchronization unit

When auto synchronization is enabled, after each LIN synch break, the time duration between five falling edges on RDI is sampled on f_{CPU} and the result of this measurement is stored in an internal 15-bit register called SM (not user accessible) (see *Figure 64*). Then the LDIV value (and its associated LPFR and LPR registers) are automatically updated at the end of the fifth falling edge. During LIN synch field measurement, the SCI state machine is stopped and no data is transferred to the data register.

LIN slave baud rate generation

In LIN mode, transmission and reception are driven by the LIN baud rate generator

Note: LIN master mode uses the extended or conventional prescaler register to generate the baud rate.

If LINE bit = 1 and LSLV bit = 1 then the conventional and extended baud rate generators are disabled The baud rate for the receiver and transmitter are both set to the same value, depending on the LIN slave baud rate generator:

$$Tx = Rx = \frac{f_{CPU}}{(16*LDIV)}$$

where:

LDIV is an unsigned fixed point number. The mantissa is coded on 8 bits in the LPR register and the fraction is coded on 4 bits in the LPFR register.

If LASE bit = 1 then LDIV is automatically updated at the end of each LIN synch field.

Three registers are used internally to manage the auto-update of the LIN divider (LDIV):

- LDIV_NOM (nominal value written by software at LPR/LPFR addresses)
- LDIV_MEAS (results of the field synch measurement)
- LDIV (used to generate the local baud rate)

The control and interactions of these registers, explained in *Figure 65* and *Figure 66*, depend on the LDUM bit setting (LIN divider update method).

Note: As explained in Figure 65 and Figure 66, LDIV can be updated by two concurrent actions: a transfer from LDIV_MEAS at the end of the LIN sync field and a transfer from LDIV_NOM due to a software write of LPR. If both operations occur at the same time, the transfer from LDIV_NOM has priority.

Figure 65. LDIV read/write operations when LDUM = 0

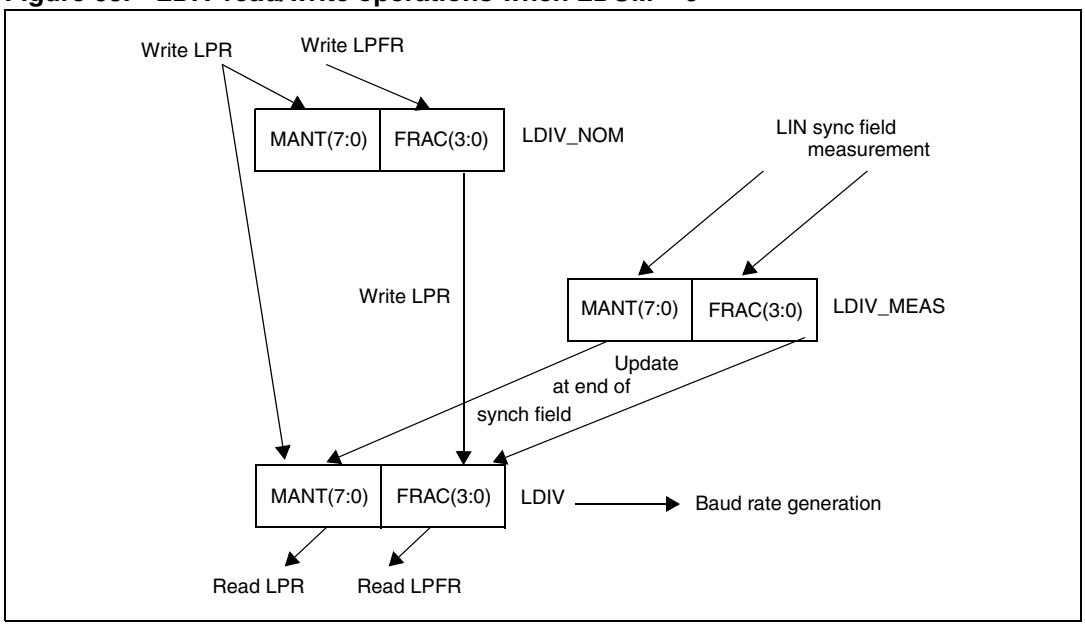
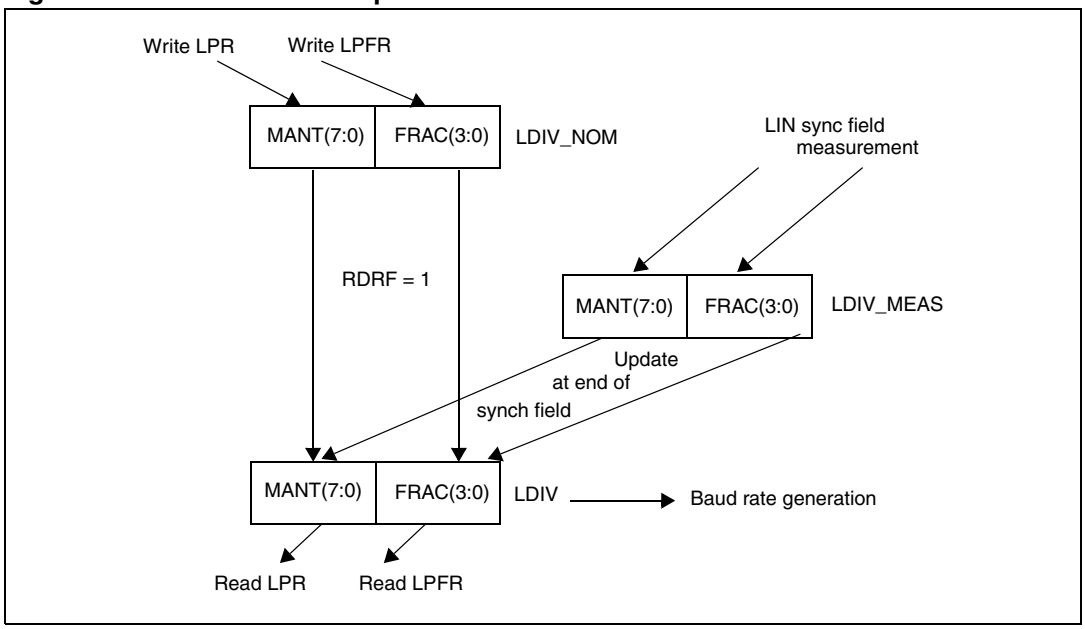


Figure 66. LDIV read/write operations when LDUM = 1



LINSCI clock tolerance

LINSCI clock tolerance when unsynchronized

When LIN slaves are unsynchronized (meaning no characters have been transmitted for a relatively long time), the maximum tolerated deviation of the LINSCI clock is ±15%.

If the deviation is within this range then the LIN synch break is detected properly when a new reception occurs.

This is made possible by the fact that masters send 13 low bits for the LIN synch break, which can be interpreted as 11 low bits (13 bits -15% = 11.05) by a ‘fast’ slave and then considered as a LIN synch break. According to the LIN specification, a LIN synch break is valid when its duration is greater than t_{SBRKTS} = 10. This means that the LIN synch break must last at least 11 low bits.

Note: If the period desynchronization of the slave is +15% (slave too slow), the character ‘00h’ which represents a sequence of 9 low bits must not be interpreted as a break character (9 bits + 15% = 10.35). Consequently, a valid LIN synch break must last at least 11 low bits.

LINSCI clock tolerance when synchronized

When synchronization has been performed, following reception of a LIN synch break, the LINSCI, in LIN mode, has the same clock deviation tolerance as in SCI mode, which is explained below:

During reception, each bit is oversampled 16 times. The mean of the 8th, 9th and 10th samples is considered as the bit value.

Consequently, the clock frequency should not vary more than 6/16 (37.5%) within one bit.

The sampling clock is resynchronized at each start bit, so that when receiving 10 bits (one start bit, 1 data byte, 1 stop bit), the clock deviation should not exceed 3.75%.

Clock deviation causes

The causes which contribute to the total deviation are:

– D_{TRA}: Deviation due to transmitter error.

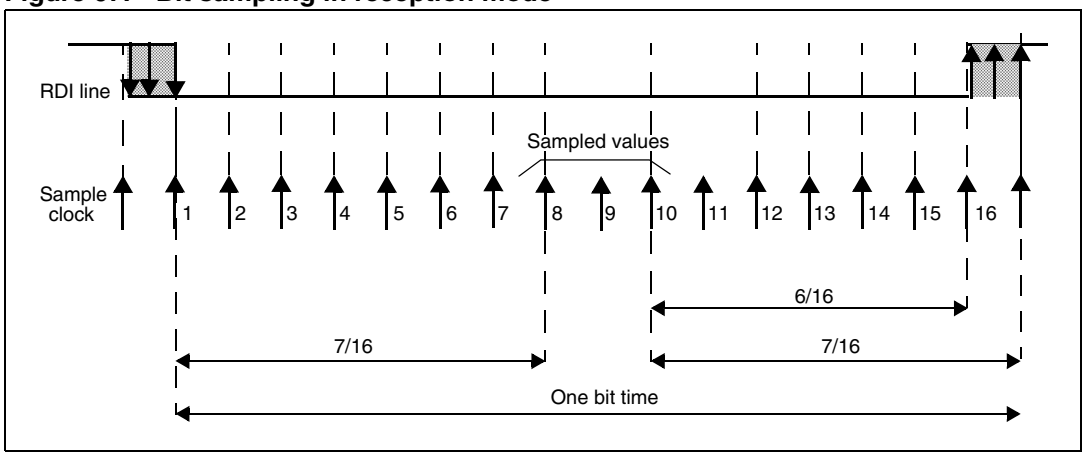
Note: The transmitter can be either a master or a slave (in case of a slave listening to the response of another slave)

– D_{MEAS}: Error due to the LIN synch measurement performed by the receiver
– D_{QUANT}: Error due to the baud rate quantization of the receiver
– D_{REC}: Deviation of the local oscillator of the receiver. This deviation can occur during the reception of one complete LIN message assuming that the deviation has been compensated at the beginning of the message.
– D_{TCL}: Deviation due to the transmission line (generally due to the transceivers)

All the deviations of the system should be added and compared to the LINSCI clock tolerance:

$D_{TRA} + D_{MEAS} + D_{QUANT} + D_{REC} + D_{TCL} < 3.75\%$

Figure 67. Bit sampling in reception mode



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%.

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 $1/(2*16*LDIV_{MIN})$.

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 20 Kbits/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.

11.5.10 LIN mode register description

Status register (SCISR)

SCISR — Reset value: 1100 0000 (C0h)

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	LHE	NF	FE	PE
RO	RO	RO	RO	RO	RO	RO	RO

Table 70. SCISR register description(1)

Bit	Name	Function
7	TDRE	Transmit data register empty This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register). 0: Data is not transferred to the shift register 1: Data is transferred to the shift register
6	TC	Transmission complete This bit is set by hardware when transmission of a character containing data is complete. An interrupt is generated if TCIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register). 0: Transmission is not complete 1: Transmission is complete *Note: TC is not set after the transmission of a preamble or a break.*
5	RDRF	Received data ready flag This bit is set by hardware when the content of the RDR register has been transferred to the SCIDR register. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register). 0: Data is not received 1: Received data are ready to be read
4	IDLE	Idle line detected This bit is set by hardware when an idle line is detected. An interrupt is generated if the ILIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register). 0: No idle line is detected 1: idle line is detected *Note: The IDLE bit is not set again until the RDRF bit has been set itself (that is, a new idle line occurs).*
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 *Status register (SCISR) on page 134*). 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 overrun error as in SCI mode, (see description in Status register (SCISR) on page 134).*

Table 70. SCISR register description(1) (continued)

Bit	Name	Function
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 *Status register (SCISR) on page 134*. In LIN slave mode (LINE bit = 1 and LSLV bit = 1) this bit has no meaning.
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
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

1. Bits 7:4 have the same function as in SCI mode, please refer to *Status register (SCISR) on page 134*.

Control register 1 (SCICR1)

SCICR1 Reset value: x000 0000 (x0h)

7	6	5	4	3	2	1	0
R8	T8	SCID	M	WAKE	PCE	Reserved	PIE
R/W	R/W	R/W	R/W	R/W	R/W	-	R/W

Table 71. SCICR1 register description(1)

Bit	Name	Function
7	R8	Receive data bit 8 This bit is used to store the 9th bit of the received word when M = 1.
6	T8	Transmit data bit 8 This bit is used to store the 9th bit of the transmitted word when M = 1.
5	SCID	Disabled for low power consumption When this bit is set the SCI prescalers and outputs are stopped and the end of the current byte transfer in order to reduce power consumption.This bit is set and cleared by software. 0: SCI enabled 1: SCI prescaler and outputs disabled

Table 71. SCICR1 register description(1) (continued)

Bit	Name	Function
4	M	Word length This bit determines the word length. It is set or cleared by software. 0: 1 start bit, 8 data bits, 1 stop bit 1: 1 start bit, 9 data bits, 1 stop bit *Note: The M bit must not be modified during a data transfer (both transmission and reception).*
3	WAKE	Wake-up method This bit determines the SCI wake-up method. It is set or cleared by software. 0: Idle line 1: Address mark *Note: If the LINE bit is set, the WAKE bit is deactivated and replaced by the LHDM 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.
1	-	Reserved, must be kept cleared
0	PIE	Parity interrupt enable This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). The parity error involved can be a byte parity error (if bit PCE is set and bit LPE is reset) or a LIN parity error (if bit PCE is set and bit LPE is set). 0: Parity error interrupt disabled 1: Parity error interrupt enabled

1. Bits 7:3 and bit 0 have the same function as in SCI mode; please refer to *Control register 1 (SCICR1) on page 136*.

Control register 2 (SCICR2)

SCICR2 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 72. SCICR2 register description(1)

Bit	Name	Function
7	TIE	Transmitter interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever TDRE = 1 in the SCISR register
6	TCIE	Transmission complete interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever TC = 1 in the SCISR register
5	RIE	Receiver interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever OR = 1 or RDRF = 1 in the SCISR register
4	ILIE	Idle line interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SCI interrupt is generated whenever IDLE = 1 in the SCISR register
3	TE	Transmitter enable This bit enables the transmitter. It is set and cleared by software. 0: Transmitter is disabled 1: Transmitter is enabled *Note: During transmission, a '0' pulse on the TE bit ('0' followed by '1') sends a preamble (idle line) after the current word.* *When TE is set, there is a 1 bit-time delay before the transmission starts.*
2	RE	Receiver enable This bit enables the receiver. It is set and cleared by software. 0: Receiver is disabled in the SCISR register 1: Receiver is enabled and begins searching for a start 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 *Note: Mute mode is recommended for detecting only the header and avoiding the reception of any other characters. For more details please refer to LIN reception on page 143.* *In LIN slave mode, when RDRF is set, the software can not set or clear the RWU 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 sends a BREAK word at the end of the current word.*

1. Bits 7:2 have the same function as in SCI mode; please refer to *Control register 2 (SCICR2) on page 137*.

Control register 3 (SCICR3)

SCICR3 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
LDUM	LINE	LSLV	LASE	LHDM	LHIE	LHDF	LSF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

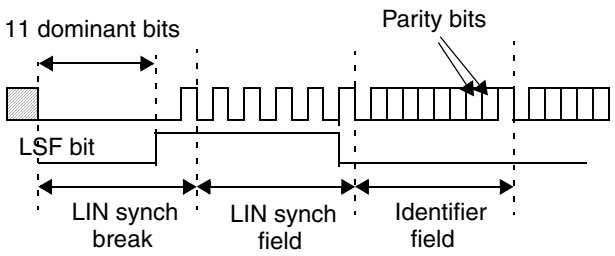
Table 73. SCICR3 register description

Bit	Name	Function
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 *Note: If no write to LPR is performed between the setting of LDUM bit and the reception of the next character, LDIV is 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.*
6:5	LINE, LSLV	LIN mode enable bits These bits configure the LIN mode: 0x: LIN mode disabled 10: LIN master mode 11: LIN slave mode **The LIN master configuration** enables sending of 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)
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

Table 73. SCICR3 register description (continued)

Bit	Name	Function
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 SCICR2 register is set, the LHDM bit selects the wake-up method (replacing the WAKE bit). 0: LIN synch break detection method 1: LIN identifier field detection method
2	LHIE	LIN header interrupt enable This bit is set and cleared by software. It is only usable in LIN slave mode. 0: LIN header interrupt is inhibited 1: An SCI interrupt is generated whenever LHDF = 1
1	LHDF	LIN header detection flag This bit is set by hardware when a LIN header is detected and cleared by a software sequence (an access to the SCISR register followed by a read of the SCICR3 register). It is only usable in LIN slave mode. 0: No LIN header detected 1: LIN header detected *Note: The header detection method depends on the LHDM bit:* *- If LHDM = 0, a header is detected as a LIN synch break* *- If LHDM = 1, a header is detected as a LIN Identifier, meaning that a LIN synch break field + a LIN synch field + a LIN identifier field have been consecutively received.*
0	LSF	LIN synch field state This bit indicates that the LIN synch field is being analyzed. It is only used in LIN slave mode. In auto synchronization mode (LASE bit = 1), when the SCI is in the LIN synch field state it waits or counts the falling edges on the RDI line. It is set by hardware as soon as a LIN synch break is detected and cleared by hardware when the LIN synch field analysis is finished (see *Figure 68*). This bit can also be cleared by software to exit LIN Synch state and return to idle mode. 0: The current character is not the LIN synch field 1: LIN synch field state (LIN synch field undergoing analysis)

Figure 68. LSF bit set and clear



LIN divider registers

LDIV is coded using the two registers LPR and LPFR. In LIN slave mode, the LPR register is accessible at the address of the SCIBRR register and the LPFR register is accessible at the address of the SCIETPR register.

LIN prescaler register (LPR)

LPR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
LPR[7:0]							
R/W							

Table 74. LPR register description

Bit	Name	Function
7:0	LPR[7:0]	LIN prescaler (mantissa of LDIV) These 8 bits define the value of the mantissa of the LDIV (see *Table 75*).

Table 75. LIN mantissa rounded values

LPR[7:0]	Rounded mantissa (LDIV)
00h	SCI clock disabled
01h	1
...	...
FEh	254
FFh	255

Caution: LPR and LPFR registers have different meanings when reading or writing to them. Consequently bit manipulation instructions (BRES or BSET) should never be used to modify the LPR[7:0] bits, or the LPFR[3:0] bits.

LIN prescaler fraction register (LPFR)

LPFR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved				LPFR[3:0]			
-				R/W			

Table 76. LPFR register description

Bit	Name	Function
7:4	-	Reserved, must be kept cleared
3:0	LPFR[3:0]	Fraction of LDIV These 4 bits define the fraction of the LDIV (see *Table 77*). *Note: When initializing LDIV, the LPFR register must be written first. Then, the write to the LPR register effectively updates LDIV and so the clock generation.* *In LIN slave mode, if the LPR[7:0] register is equal to 00h, the transceiver and receiver input clocks are switched off.*

Table 77. LDIV fractions

LPFR[3:0]	Fraction (LDIV)
0h	0
1h	1/16
...	...
Eh	14/16
Fh	15/16

Examples of LDIV coding

Example 1: LPR = 27d and LPFR = 12d

This leads to:

Mantissa (LDIV) = 27d

Fraction (LDIV) = 12/16 = 0.75d

Therefore LDIV = 27.75d

Example 2: LDIV = 25.62d

This leads to:

LPFR = rounded(16*0.62d) = rounded(9.92d) = 10d = Ah

LPR = mantissa (25.620d) = 25d = 1Bh

Example 3: LDIV = 25.99d

This leads to:

LPFR = rounded(16*0.99d) = rounded(15.84d) = 16d

The carry must be propagated to the mantissa: LPR = mantissa (25.99) + 1 = 26d = 1Ch.

LIN header length register (LHLR)

LHLR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
LHL[7:0]							
R							

Note: In LIN slave mode when LASE = 1 or LHDM = 1, the LHLR register is accessible at the address of the SCIERPR register. Otherwise this register is always read as 00h.

Table 78. LHLR register description

Bit	Name	Function
7:0	LHL[7:0]	LIN header length This is a read-only register, which is updated by hardware if one of the following conditions occurs: After each break detection, it is loaded with 'FFh' If a timeout occurs on T_{HEADER}, it is loaded with 00h After every successful LIN header reception (at the same time as the setting of LHDF bit), it is loaded with a value (LHL) which gives access to the number of bit times of the LIN header length (T_{HEADER}). LHL register coding is as follows: T_{HEADER_MAX} = 57 LHL (7:2) represents the mantissa of (57 - T_{HEADER}) (see *Table 79*) LHL (1:0) represents the fraction (57 - T_{HEADER}) (see *Table 80*)

Table 79. LIN header mantissa values

LHL[7:2]	Mantissa (57 - T_{HEADER})	Mantissa (T_{HEADER})
0h	0	57
1h	1	56
...	...	...
39h	56	1
3Ah	57	0
3Bh	58	Never occurs
...	...	...
3Eh	62	Never occurs
3Fh	63	Initial value

Table 80. LIN header fractions

LHL[1:0]	Fraction (57 - T_{HEADER})
0h	0
1h	1/4
2h	1/2
3h	3/4

Examples of LHL coding:

Example 1: LHL = 33h = 001100 11b

LHL(7:3) = 1100b = 12d

LHL(1:0) = 11b = 3d

This leads to:

Mantissa (57 - T_{HEADER}) = 12d

Fraction (57 - T_{HEADER}) = 3/4 = 0.75

Therefore:

(57 - T_{HEADER}) = 12.75d and T_{HEADER} = 44.25d

Example 2:

57 - T_{HEADER} = 36.21d

LHL(1:0) = rounded(4*0.21d) = 1d

LHL(7:2) = Mantissa (36.21d) = 36d = 24h

Therefore LHL(7:0) = 10010001 = 91h

Example 3:

57 - T_{HEADER} = 36.90d

LHL(1:0) = rounded(4*0.90d) = 4d

The carry must be propagated to the matissa:

LHL(7:2) = Mantissa (36.90d) + 1 = 37d

Therefore LHL(7:0) = 10110000 = A0h

Table 81. LINSCI1 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 81. LINSCI1 register map and reset values (continued)

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 -

11.6 10-bit A/D converter (ADC)

11.6.1 Introduction

The on-chip analog to digital converter (ADC) peripheral is a 10-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to seven multiplexed analog input channels (refer to device pinout description) that allow the peripheral to convert the analog voltage levels from up to seven different sources.

The result of the conversion is stored in a 10-bit data register. The A/D converter is controlled through a control/status register.

11.6.2 Main features

- 10-bit conversion
- Up to 7 channels with multiplexed input
- Linear successive approximation
- Data register (DR) which contains the results
- Conversion complete status flag
- On/off bit (to reduce consumption)

The block diagram is shown in *Figure 69: ADC block diagram on page 164*.

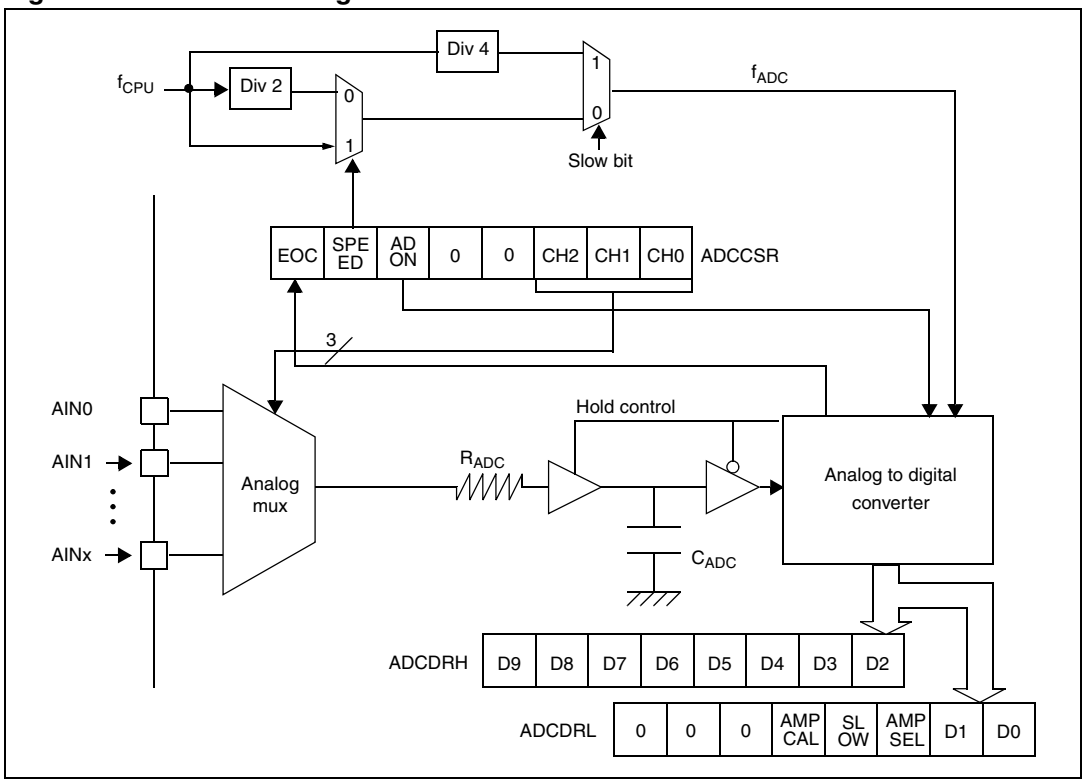
11.6.3 Functional description

Analog power supply

V_{DDA} and V_{SSA} are the high and low level reference voltage pins. In some devices (refer to *Section 2: Pin description*) they are internally connected to the V_{DD} and V_{SS} pins.

Conversion accuracy may therefore be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines.

Figure 69. ADC block diagram



Digital A/D conversion result

The conversion is monotonic, meaning that the result never decreases if the analog input does not decrease and never increases if the analog input does not increase.

If the input voltage (V_{AIN}) is greater than V_{DDA} (high-level voltage reference) then the conversion result is FFh in the ADCDRH register and 03h in the ADCDRL register (without overflow indication).

If the input voltage (V_{AIN}) is lower than V_{SSA} (low-level voltage reference) then the conversion result in the ADCDRH and ADCDRL registers is 00 00h.

The A/D converter is linear and the digital result of the conversion is stored in the ADCDRH and ADCDRL registers. The accuracy of the conversion is described in the *Section 13: Electrical characteristics*.

R_{AIN} is the maximum recommended impedance for an analog input signal. If the impedance is too high, this results in a loss of accuracy due to leakage and sampling not being completed in the allotted time.

A/D conversion

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to *Section 10: I/O ports*. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

In the ADCCSR register, select the CH[2:0] bits to assign the analog channel to convert.

ADC conversion mode

In the ADCCSR register, set the ADON bit to enable the A/D converter and to start the conversion. From this time on, the ADC performs a continuous conversion of the selected channel.

When a conversion is complete:

- The EOC bit is set by hardware
- The result is in the ADCDR registers

A read to the ADCDRH resets the EOC bit.

To read the 10 bits, perform the following steps:

1. Poll the EOC bit
2. Read ADCDRL
3. Read ADCDRH. This clears EOC automatically

To read only 8 bits, perform the following steps:

1. Poll EOC bit
2. Read ADCDRH. This clears EOC automatically

11.6.4 Low-power modes

Note: The A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed and between single shot conversions

Table 82. Effect of low power modes on the A/D converter

Mode	Description
Wait	No effect on A/D converter
Halt	A/D converter disabled After wakeup from halt mode, the A/D converter requires a stabilization time t_{STAB} (*Section 13: Electrical characteristics*) before accurate conversions can be performed.

11.6.5 Interrupts

None.

11.6.6 Register description

Control/status register (ADCCSR)

ADCCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
EOC	SPEED	ADON	Reserved	Reserved	CH[2:0]		
R	R/W	R/W	-	-	R/W		

Table 83. ADCCSR register description

Bit	Bit name	Function
7	EOC	End of conversion This bit is set by hardware. It is cleared by software reading the ADCDRH register. 0: Conversion is not complete 1: Conversion complete
6	SPEED	ADC clock selection This bit is set and cleared by software. It is used together with the SLOW bit to configure the ADC clock speed. Refer to *Table 86: ADC clock configuration on page 167* concerning the SLOW bit description of the ADCDRL register.
5	ADON	A/D converter on This bit is set and cleared by software. 0: A/D converter is switched off 1: A/D converter is switched on
4:3	-	Reserved, must be kept cleared.
2:0	CH[2:0]	Channel selection These bits are set and cleared by software. They select the analog input to convert: 000: channel pin = AIN0 001: channel pin = AIN1 010: channel pin = AIN2 011: channel pin = AIN3 100: channel pin = AIN4 101: channel pin = AIN5 110: channel pin = AIN6 *Note: The number of channels is device dependent. Refer to Section 2: Pin description.*

Data register high (ADCDRH)

ADCDRH Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
D[9:2]							
R							

Table 84. ADCDRH register description

Bit	Bit name	Function
7:0	D[9:2]	MSB of analog converted value

Control and data register low (ADCDRL)

ADCDRL Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	SLOW	Reserved	D[1:0]	
-	-	-	-	R/W	-	R/W	

Table 85. ADCDRL register description

Bit	Bit name	Function
7:5	-	Reserved, must be kept cleared
4	-	Reserved, must be kept cleared
3	SLOW	Slow mode This bit is set and cleared by software. It is used together with the SPEED bit to configure the ADC clock speed as shown in *Table 86: ADC clock configuration on page 167*
2	-	Reserved, must be kept cleared
1:0	D[1:0]	LSB of converted analog value

Table 86. ADC clock configuration(1)

f_{ADC}	SLOW	SPEED
$f_{CPU}/2$	0	0
f_{CPU}	0	1
$f_{CPU}/4$	1	x

1. Max f_{ADC} allowed = 4 MHz (see *Section 13.11: 10-bit ADC characteristics on page 210*)

Table 87. 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	ADCDRL Reset value	0 0	0 0	0 0	0 0	SLOW 0	0 0	D1 x	D0 x

12 Instruction set

12.1 ST7 addressing modes

The ST7 core features 17 different addressing modes which can be classified in seven main groups (see *Table 88*).

Table 88. CPU addressing mode groups

Addressing mode	Example
Inherent	nop
Immediate	ld A,#$55
Direct	ld A,$55
Indexed	ld A,($55,X)
Indirect	ld A,([$55],X)
Relative	jrne loop
Bit operation	bset byte,#5

The ST7instruction set is designed to minimize the number of bytes required per instruction. To do so, most of the addressing modes may be subdivided in two sub-modes called long and short:

- Long addressing mode is more powerful because it can use the full 64 Kbyte address space, however it uses more bytes and more CPU cycles.
- Short addressing mode is less powerful because it can generally only access page zero (0000h - 00FFh range), but the instruction size is more compact, and faster. All memory to memory instructions use short addressing modes only (CLR, CPL, NEG, BSET, BRES, BTJT, BTJF, INC, DEC, RLC, RRC, SLL, SRL, SRA, SWAP)

The ST7 assembler optimizes the use of long and short addressing modes.

Table 89. CPU addressing mode overview

Mode			Syntax	Destination/source	Pointer address	Pointer size	Length (bytes)
Inherent			nop				+ 0
Immediate			ld A,#$55				+ 1
Short	Direct		ld A,$10	00..FF			+ 1
Long	Direct		ld A,$1000	0000..FFFF			+ 2
No offset	Direct	Indexed	ld A,(X)	00..FF			+ 0 (with X register) +1 (with Y register)
Short	Direct	Indexed	ld A,($10,X)	00..1FE			+ 1
Long	Direct	Indexed	ld A,($1000,X)	0000..FFFF			+ 2
Short	Indirect		ld A,[$10]	00..FF	00..FF	byte	+ 2
Long	Indirect		ld A,[$10.w]	0000..FFFF	00..FF	word	+ 2
Short	Indirect	Indexed	ld A,([$10],X)	00..1FE	00..FF	byte	+ 2
Long	Indirect	Indexed	ld A,([$10.w],X)	0000..FFFF	00..FF	word	+ 2
Relative	Direct		jrne loop	PC-128/PC+127(1)			+ 1
Relative	Indirect		jrne [$10]	PC-128/PC+127(1)	00..FF	byte	+ 2
Bit	Direct		bset $10,#7	00..FF			+ 1
Bit	Indirect		bset [$10],#7	00..FF	00..FF	byte	+ 2
Bit	Direct	Relative	btjt $10,#7,skip	00..FF			+ 2
Bit	Indirect	Relative	btjt [$10],#7,skip	00..FF	00..FF	byte	+ 3

1. At the time the instruction is executed, the program counter (PC) points to the instruction following JRxx.

12.1.1 Inherent

All inherent instructions consist of a single byte. The opcode fully specifies all the required information for the CPU to process the operation.

Table 90. Inherent instructions

Inherent instruction	Function
NOP	No operation
TRAP	S/W interrupt
WFI	Wait for interrupt (low power mode)
HALT	Halt oscillator (lowest power mode)
RET	Subroutine return
IRET	Interrupt subroutine return
SIM	Set interrupt mask
RIM	Reset interrupt mask
SCF	Set carry flag
RCF	Reset carry flag
RSP	Reset stack pointer
LD	Load
CLR	Clear
PUSH/POP	Push/pop to/from the stack
INC/DEC	Increment/decrement
TNZ	Test negative or zero
CPL, NEG	1 or 2 complement
MUL	Byte multiplication
SLL, SRL, SRA, RLC, RRC	Shift and rotate operations
SWAP	Swap nibbles

12.1.2 Immediate

Immediate instructions have 2 bytes, the first byte contains the opcode, the second byte contains the operand value.

Table 91. Immediate instructions

Immediate instruction	Function
LD	Load
CP	Compare
BCP	Bit compare
AND, OR, XOR	Logical operations
ADC, ADD, SUB, SBC	Arithmetic operations

12.1.3 Direct

In direct instructions, the operands are referenced by their memory address.

The direct addressing mode consists of two sub-modes:

- Direct instructions (short)

  The address is a byte, thus requires only one byte after the opcode, but only allows 00 - FF addressing space.
- Direct instructions (long)

  The address is a word, thus allowing 64 Kbyte addressing space, but requires two bytes after the opcode.

12.1.4 Indexed (no offset, short, long)

In this mode, the operand is referenced by its memory address, which is defined by the unsigned addition of an index register (X or Y) with an offset.

The indirect addressing mode consists of three sub-modes:

- Indexed (no offset)

  There is no offset, (no extra byte after the opcode), and allows 00 - FF addressing space.
- Indexed (short)

  The offset is a byte, thus requires only one byte after the opcode and allows 00 - 1FE addressing space.
- Indexed (long)

  The offset is a word, thus allowing 64 Kbyte addressing space and requires two bytes after the opcode.

12.1.5 Indirect (short, long)

The required data byte to do the operation is found by its memory address, located in memory (pointer).

The pointer address follows the opcode. The indirect addressing mode consists of two sub-modes:

- Indirect (short)

  The pointer address is a byte, the pointer size is a byte, thus allowing 00 - FF addressing space, and requires 1 byte after the opcode.
- Indirect (long)

  The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

12.1.6 Indirect indexed (short, long)

This is a combination of indirect and short indexed addressing modes. The operand is referenced by its memory address, which is defined by the unsigned addition of an index register value (X or Y) with a pointer value located in memory. The pointer address follows the opcode.

The indirect indexed addressing mode consists of two sub-modes:

- Indirect indexed (short)

  The pointer address is a byte, the pointer size is a byte, thus allowing 00 - 1FE addressing space, and requires 1 byte after the opcode.

- Indirect indexed (long)

  The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

Table 92. Instructions supporting direct, indexed, indirect and indirect indexed addressing modes

Long and short instructions	Function
LD	Load
CP	Compare
AND, OR, XOR	Logical operations
ADC, ADD, SUB, SBC	Arithmetic addition/subtraction operations
BCP	Bit compare

Table 93. Short instructions and functions

Short instructions only	Function
CLR	Clear
INC, DEC	Increment/decrement
TNZ	Test negative or zero
CPL, NEG	1 or 2 complement
BSET, BRES	Bit operations
BTJT, BTJF	Bit test and jump operations
SLL, SRL, SRA, RLC, RRC	Shift and rotate operations
SWAP	Swap nibbles
CALL, JP	Call or jump subroutine

12.1.7 Relative mode (direct, indirect)

This addressing mode is used to modify the PC register value, by adding an 8-bit signed offset to it.

Table 94. Relative mode instructions (direct and indirect)

Available relative direct/indirect instructions	Function
JRxx	Conditional jump
CALLR	Call relative

The relative addressing mode consists of two sub-modes:

- Relative (direct)
  The offset follows the opcode
- Relative (indirect)
  The offset is defined in memory, of which the address follows the opcode.

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 *Table 95*.

Table 95. Instruction groups

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				

12.2.1 Using a prebyte

The instructions are described with one to four 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 instructions in Y as well as indirect addressing modes to be implemented. They precede the opcode of the instructions in X or the instructions using direct addressing mode. The prebytes are:

PDY 90 Replaces an X based instruction using immediate, direct, indexed, or inherent addressing mode by a Y one

PIX 92 Replaces an instruction using direct, direct bit, or direct relative addressing mode by 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 Replaces an instruction using X indirect indexed addressing mode by a Y one

12.2.2 Illegal opcode reset

In order to provide the device with enhanced robustness 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.

Table 96. Instruction set overview

Mnemo.	Description	Function/example	Dst	Src	H	I	N	Z	C
ADC	Add with carry	A = A + M + C	A	M	H		N	Z	C
ADD	Addition	A = A + M	A	M	H		N	Z	C
AND	Logical and	A = A . M	A	M			N	Z	
BCP	Bit compare A, memory	tst (A . M)	A	M			N	Z	
BRES	Bit reset	bres Byte, #3	M						
BSET	Bit set	bset Byte, #3	M						
BTJF	Jump if bit is false (0)	btjf Byte, #3, Jmp1	M						C
BTJT	Jump if bit is true (1)	btjt Byte, #3, Jmp1	M						C
CALL	Call subroutine								
CALLR	Call subroutine relative								
CLR	Clear		reg, M				0	1	
CP	Arithmetic compare	tst(Reg - M)	reg	M			N	Z	C
CPL	One complement	A = FFH-A	reg, M				N	Z	1
DEC	Decrement	dec Y	reg, M				N	Z	
HALT	Halt					0			
IRET	Interrupt routine return	Pop CC, A, X, PC			H	I	N	Z	C
INC	Increment	inc X	reg, M				N	Z	
JP	Absolute jump	jp [TBL.w]							
JRA	Jump relative always								
JRT	Jump relative								
JRF	Never jump	jrf *							
JRIH	Jump if ext. interrupt = 1								
JRIL	Jump if ext. interrupt = 0								
JRH	Jump if H = 1	H = 1 ?							
JRNH	Jump if H = 0	H = 0 ?							
JRM	Jump if I = 1	I = 1 ?							
JRNM	Jump if I = 0	I = 0 ?							
JRMI	Jump if N = 1 (minus)	N = 1 ?							
JRPL	Jump if N = 0 (plus)	N = 0 ?							
JREQ	Jump if Z = 1 (equal)	Z = 1 ?							
JRNE	Jump if Z = 0 (not equal)	Z = 0 ?							
JRC	Jump if C = 1	C = 1 ?							
JRNC	Jump if C = 0	C = 0 ?							
JRULT	Jump if C = 1	Unsigned <							
JRUGE	Jump if C = 0	Jmp if unsigned >=							

Table 96. Instruction set overview (continued)

Mnemo.	Description	Function/example	Dst	Src	H	I	N	Z	C
JRUGT	Jump if (C + Z = 0)	Unsigned >							
JRULE	Jump if (C + Z = 1)	Unsigned <=							
LD	Load	dst <= src	reg, M	M, reg			N	Z	
MUL	Multiply	X,A = X * A	A, X, Y	X, Y, A	0				0
NEG	Negate (2's compl)	neg $10	reg, M				N	Z	C
NOP	No operation								
OR	OR operation	A = A + M	A	M			N	Z	
POP	Pop from the stack	pop reg	reg	M					
		pop CC	CC	M	H	I	N	Z	C
PUSH	Push onto the stack	push Y	M	reg, CC					
RCF	Reset carry flag	C = 0							0
RET	Subroutine return								
RIM	Enable interrupts	I = 0				0			
RLC	Rotate left true C	C <= Dst <= C	reg, M				N	Z	C
RRC	Rotate right true C	C => Dst => C	reg, M				N	Z	C
RSP	Reset stack pointer	S = Max allowed							
SBC	Subtract with carry	A = A - M - C	A	M			N	Z	C
SCF	Set carry flag	C = 1							1
SIM	Disable interrupts	I = 1				1			
SLA	Shift left arithmetic	C <= Dst <= 0	reg, M				N	Z	C
SLL	Shift left logic	C <= Dst <= 0	reg, M				N	Z	C
SRL	Shift right logic	0 => Dst => C	reg, M				0	Z	C
SRA	Shift right arithmetic	Dst7 => Dst => C	reg, M				N	Z	C
SUB	Subtraction	A = A - M	A	M			N	Z	C
SWAP	SWAP nibbles	Dst[7..4] <=> Dst[3..0]	reg, M				N	Z	
TNZ	Test for neg & zero	tnz lbl1					N	Z	
TRAP	S/W trap	S/W interrupt				1			
WFI	Wait for Interrupt					0			
XOR	Exclusive OR	A = A XOR M	A	M			N	Z	

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} = 5 V (for the 4.5 V ≤ V_{DD} ≤ 5.5 V voltage range) and V_{DD} = 3.3 V (for the 3 V ≤ V_{DD} ≤ 3.6 V voltage range). They are given only as design guidelines and are not tested.

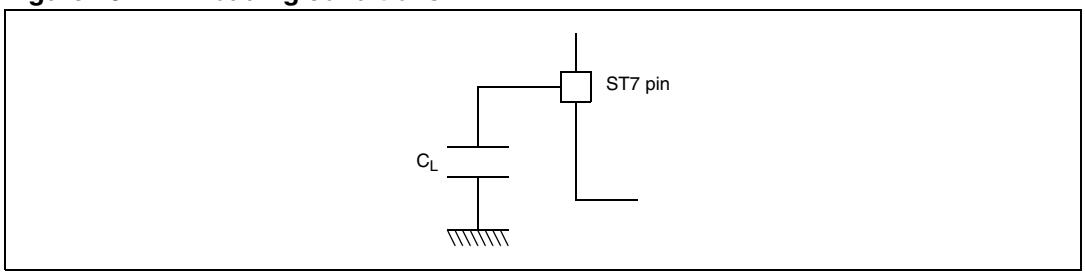
13.1.3 Typical curves

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 70*.

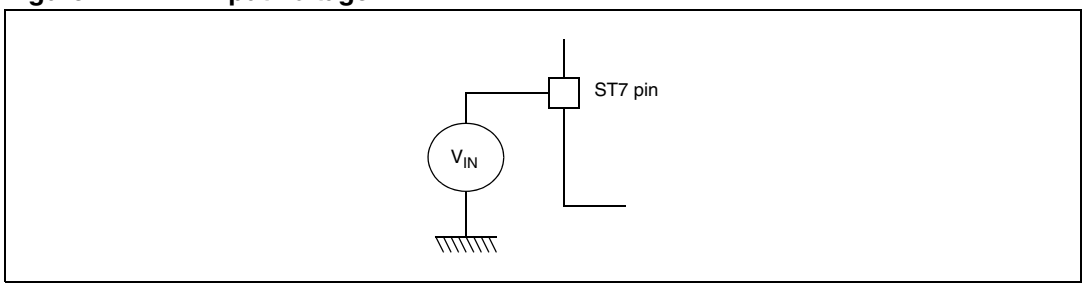
Figure 70. Pin loading conditions



13.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in *Figure 71*.

Figure 71. Pin input voltage



13.2 Absolute maximum ratings

Stresses above those listed as ‘absolute maximum ratings’ may cause permanent damage to the device. This is a stress rating only and functional operation of the device under these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

13.2.1 Voltage characteristics

Table 97. Voltage characteristics

Symbol	Ratings	Maximum value	Unit
V_{DD} - V_{SS}	Supply voltage	7.0	V
V_{IN}	Input voltage on any pin(1)(2)	V_{SS} - 0.3 to V_{DD} + 0.3	
$V_{ESD(HBM)}$	Electrostatic discharge voltage (human body model)	See *Section 13.7.3: Absolute maximum ratings (electrical sensitivity) on page 198*	
$V_{ESD(MM)}$	Electrostatic discharge voltage (machine model)		

1. Directly connecting the $\overline{\text{RESET}}$ and I/O pins to V_{DD} or V_{SS} could damage the device if an unintentional internal reset is generated or an unexpected change of the I/O configuration occurs (for example, due to a corrupted program counter). To guarantee safe operation, this connection must be made through a pull-up or pull-down resistor (typical: 4.7 kΩ for $\overline{\text{RESET}}$, 10 kΩ for I/Os). Unused I/O pins must be tied in the same way to V_{DD} or V_{SS} according to their reset configuration.
2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN} > V_{DD}$ while a negative injection is induced by $V_{IN} < V_{SS}$. For true open-drain pads, there is no positive injection current and the corresponding V_{IN} maximum must always be respected

13.2.2 Current characteristics

Table 98. Current characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source)(1)	150	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink)(1)	150	
I_{IO}	Output current sunk by any standard I/O and control pin	20	
	Output current sunk by any high sink I/O pin	40	
	Output current source by any I/Os and control pin	- 25	
$I_{INJ(PIN)}$(2)(3)	Injected current on $\overline{\text{RESET}}$ pin	± 5	
	Injected current on OSC1 and OSC2 pins	± 5	
	Injected current on any other pin(4)	± 5	
$\Sigma I_{INJ(PIN)}$(2)	Total injected current (sum of all I/O and control pins)(4)	± 20	mA

1. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN} > V_{DD}$ while a negative injection is induced by $V_{IN} < V_{SS}$.
3. Negative injection disturbs the analog performance of the device. In particular, it induces leakage currents throughout the device including the analog inputs. To avoid undesirable effects on the analog functions, care must be taken:
   - Analog input pins must have a negative injection less than 0.8 mA (assuming that the impedance of the analog voltage is lower than the specified limits)
   - Pure digital pins must have a negative injection less than 1.6 mA. In addition, it is recommended to inject the current as far as possible from the analog input pins.
4. When several inputs are submitted to a current injection, the maximum $\Sigma I_{INJ(PIN)}$ is the absolute sum of the positive and negative injected currents (instantaneous values). These results are based on characterization with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.

13.2.3 Thermal characteristics

Table 99. Thermal characteristics

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature (see *Table 129: Thermal characteristics on page 214*)		

13.3 Operating conditions

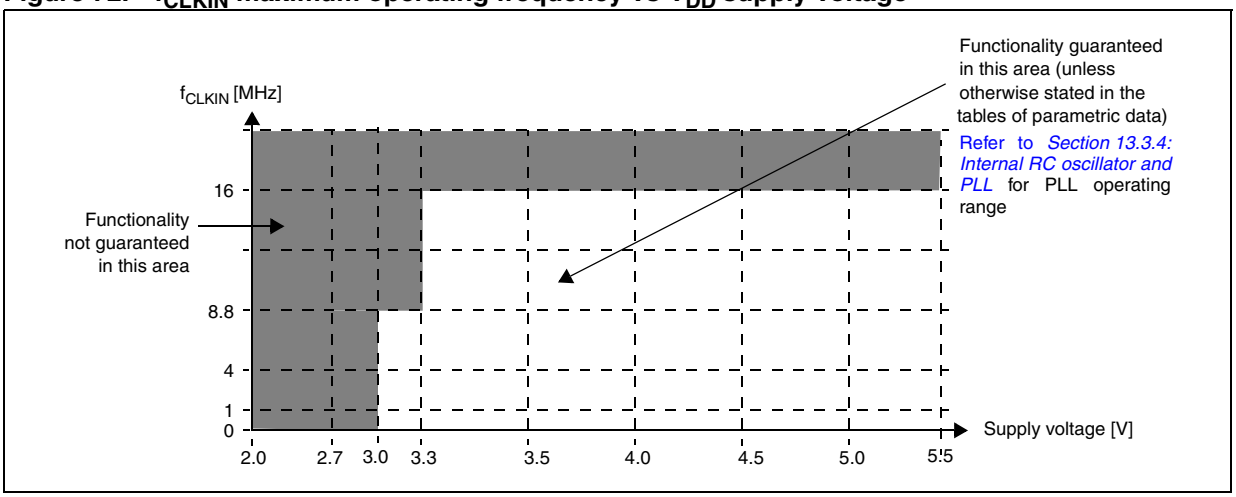
13.3.1 General operating conditions

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

Table 100. General operating conditions

Symbol	Parameter	Conditions	Min.	Max.	Unit
V_{DD}	Supply voltage	f_{OSC} = 16 MHz max. T_A = -40°C to T_A max.	3.0	5.5	V
f_{CLKIN}	External clock frequency on CLKIN pin	V_{DD} = 3 to 3.3 V	0	8.8	MHz
		V_{DD} = 3.3 to 5.5 V	0	16	MHz
T_A	Ambient temperature range	A suffix version	-40	+85	°C
		C suffix version		+125	

Figure 72. f_{CLKIN} maximum operating frequency vs V_{DD} supply voltage



1. For further information on clock management block diagram for f_{CLKIN} description, refer to *Figure 13: Clock management block diagram on page 40*.

The RC oscillator and PLL characteristics are temperature-dependent.

Table 101. Operating conditions (tested for T_A = -40 to +125 °C) @ V_{DD} = 4.5 to 5.5 V

Symbol	Parameter	Conditions	Flash			ROM			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
f_{RC}(1)	Internal RC oscillator frequency	RCCR = FF (reset value), T_A = 25 °C, V_{DD} = 5 V		630			630		kHz
		RCCR = RCCR0(2), T_A = -40 to 125 °C, V_{DD} = 5 V	930	1000	1050	TBD	1000	TBD	
ACC_{RC}	RC resolution	V_{DD} = 5 V	-1		+1	TBD		TBD	%
	Accuracy of internal RC oscillator with RCCR = RCCR0(2)(3)	T_A = -40 to +125 °C, V_{DD} = 5 V	-3		+5	TBD		TBD	
		T_A = -40 to +125 °C, V_{DD} = 4.5 V to 5.5 V(4)	-4.5		+6.5	TBD		TBD	

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 *Section 7.1: Internal RC oscillator adjustment on page 37*
3. Minimum value is obtained for hot temperature and maximum value is obtained for cold temperature
4. Data based on characterization results, not tested in production

Table 102. Operating conditions (tested for T_A = -40 to +125 °C) @ V_{DD} = 4.5 to 5.5 V

Symbol	Parameter	Conditions	Flash and ROM			Unit
			Min.	Typ.	Max.	
$I_{DD(RC)}$	RC oscillator current consumption	T_A = 25 °C, V_{DD} = 5 V		600(1)(2)		µA
$t_{su(RC)}$	RC oscillator setup time				10(3)	µs
f_{PLL}	x8 PLL input clock			1		MHz
t_{LOCK}	PLL lock time(4)			2		ms
t_{STAB}	PLL stabilization time(4)			4		
ACC_{PLL}	x8 PLL accuracy	f_{CLKIN}/2 or f_{RC} = 1 MHz @ T_A = -40 to +125 °C		0.2(5)		%
JIT_{PLL}	PLL jitter ($\Delta f_{CPU}/f_{CPU}$)			1(6)		
$I_{DD(PLL)}$	PLL current consumption	T_A = 25 °C		550(2)		µA

1. Measurement made with RC calibrated at 1 MHz
2. Data based on characterization results, not tested in production
3. See *Section 7.1: Internal RC oscillator adjustment on page 37*
4. After the LOCKED bit is set ACC_{PLL} is maximum 10% until t_{STAB} has elapsed. See *Figure 12: PLL output frequency timing diagram on page 38*.
5. Averaged over a 4ms period. After the LOCKED bit is set, a period of t_{STAB} is required to reach ACC_{PLL} accuracy
6. Guaranteed by design

Figure 73. Typical accuracy with RCCR = RCCR0 vs. V_{DD} = 4.5 to 5.5 V and temperature

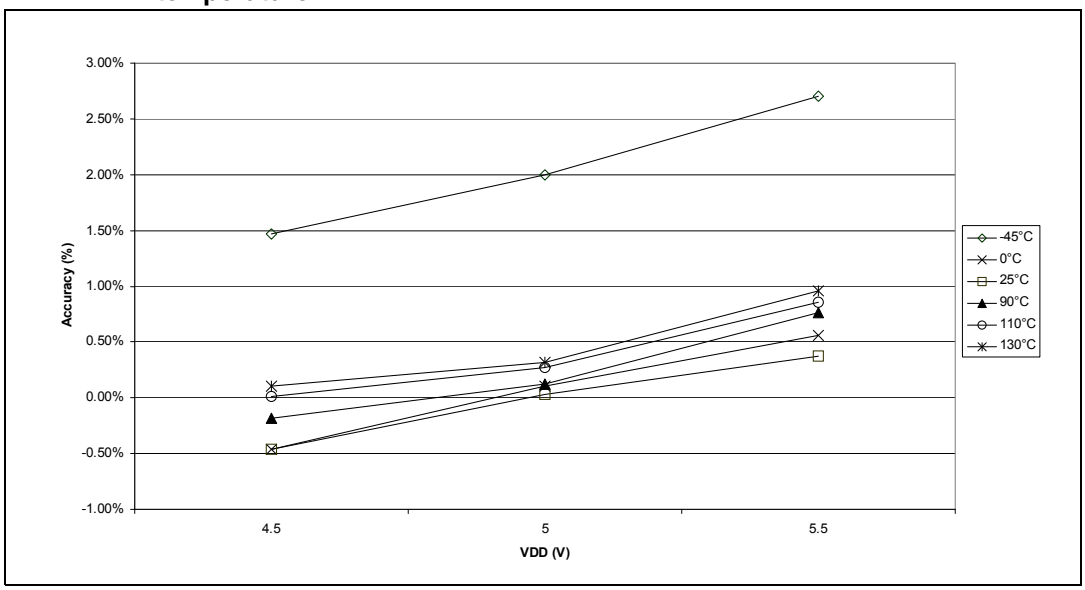


Figure 74. f_{RC} vs. V_{DD} and temperature for calibrated RCCR0

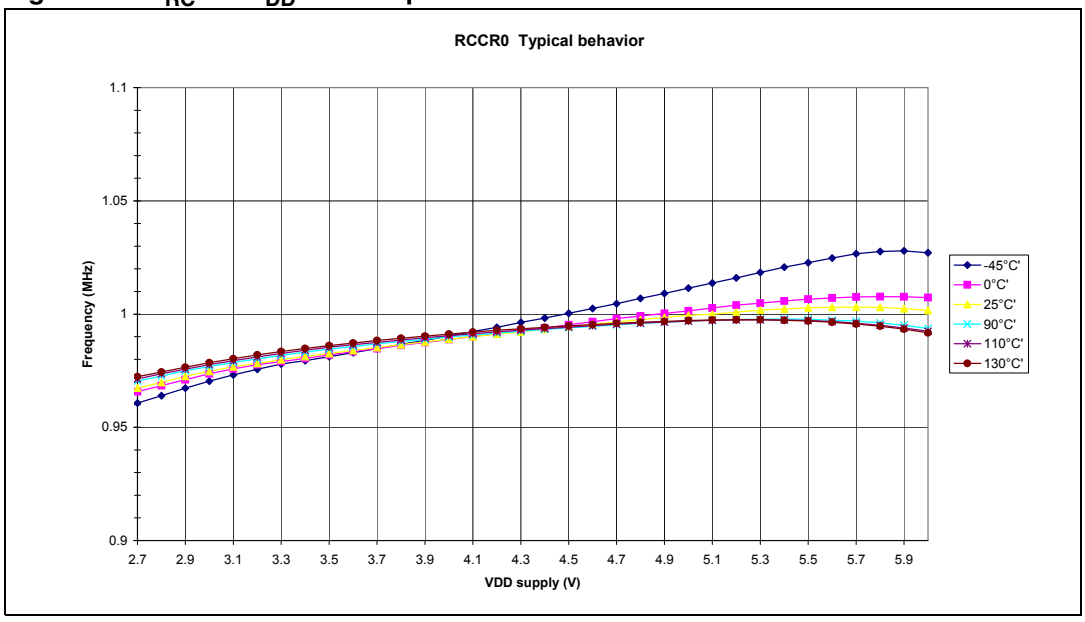


Table 103. Operating conditions (tested for T_A = -40 to +125 °C) @ V_{DD} = 3.0 to 3.6 V

Symbol	Parameter	Conditions	Flash			ROM			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
f_{RC}(1)	Internal RC oscillator frequency	RCCR = FF (reset value), T_A = 25 °C, V_{DD} = 3.3 V		630			630		kHz
		RCCR = RCCR(2), T_A = -40 to +125 °C, V_{DD} = 3.3 V	970	1000	1050	TBD	1000	TBD	

Table 103. Operating conditions (tested for T_A = -40 to +125 °C) @ V_{DD} = 3.0 to 3.6 V

Symbol	Parameter	Conditions	Flash			ROM			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
ACC_{RC}	RC resolution	V_{DD} = 3.3 V	-1		+1	TBD		TBD	%
	Accuracy of internal RC oscillator with RCCR = RCCR0(2)(3)	T_A = -40 to +125 °C, V_{DD} = 3.3 V	-3		+5	TBD		TBD	
		T_A = -40 to +125 °C, V_{DD} = 3.0 V to 3.6 V(4)	-4		+6	TBD		TBD	

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 *Section 7.1: Internal RC oscillator adjustment on page 37*.
3. Minimum value is obtained for hot temperature and max. value is obtained for cold temperature.
4. Data based on characterization results, not tested in production.

Table 104. Operating conditions (tested for T_A = -40 to +125 °C) @ V_{DD} = 3.0 to 3.6 V(1)

	Parameter(1)	Conditions	Flash and ROM			
			Min.	Typ.	Max.	
$I_{DD(RC)}$	RC oscillator current consumption	T_A = 25 °C, V_{DD} = 3.3 V		500(2)		µA
$t_{su(RC)}$	RC oscillator setup time				10(2)	µs

1. Data based on characterization results, not tested in production.
2. Measurement made with RC calibrated at 1 MHz.

Figure 75. Typical accuracy with RCCR = RCCR1 vs. V_{DD} = 3 to 3.6 V and temperature

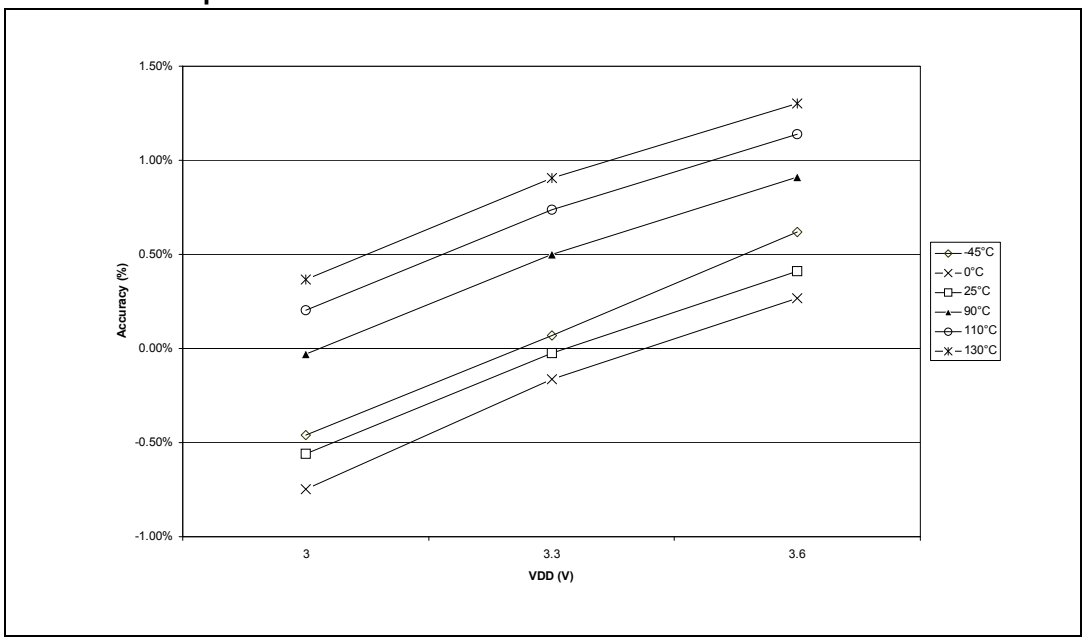


Figure 76. f_{RC} vs. V_{DD} and temperature for calibrated RCCR1

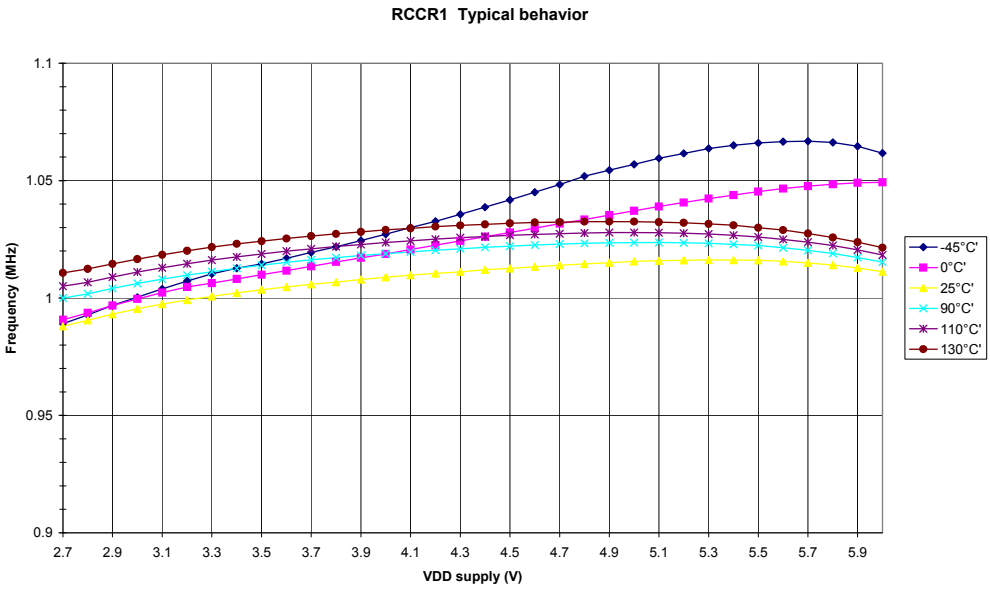
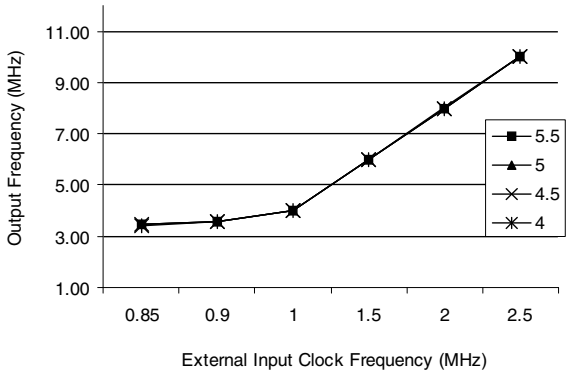


Figure 77. PLL x 8 output vs. CLKIN frequency



1. $f_{OSC} = f_{CLKIN}/2*PLL8$

13.3.2 Operating conditions with low voltage detector (LVD)

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

Table 105. Operating conditions with low voltage detector

Symbol	Parameter	Conditions(1)	Min.	Typ.	Max.	Unit
$V_{IT+(LVD)}$	Reset release threshold (V_{DD} rise)	High threshold	3.60(2)	4.15	4.50	V
$V_{IT-(LVD)}$	Reset generation threshold (V_{DD} fall)	High threshold	3.40	3.95	4.40(2)	

Table 105. Operating conditions with low voltage detector

Symbol	Parameter	Conditions(1)	Min.	Typ.	Max.	Unit
V_{hys}	LVD voltage threshold hysteresis	$V_{IT+(LVD)}$-$V_{IT-(LVD)}$		200		mV
Vt_{POR}	V_{DD} rise time rate(3)(4)		20(2)		10000 (2)	µs/V
$t_{g(VDD)}$	Filtered glitch delay on V_{DD}	Not detected by the LVD			150(5)	ns
$I_{DD(LVD)}$	LVD/AVD current consumption			200		µA

1. LVD functionality guaranteed only within the V_{DD} operating range specified in *Section 13.3.1: General operating conditions on page 182*.
2. Not tested in production.
3. Not tested in production. The V_{DD} rise time rate condition is needed to insure a correct device power-on and LVD reset. When the V_{DD} slope is outside these values, the LVD may not ensure a proper reset of the MCU.
4. Use of LVD with capacitive power supply: With this type of power supply, if power cuts occur in the application, it is recommended to pull V_{DD} down to 0 V to ensure optimum restart conditions. Refer to circuit example in *Figure 97: RESET pin protection when LVD is disabled on page 206*.
5. Based on design simulation.

13.3.3 Auxiliary voltage detector (AVD) thresholds

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

Table 106. Auxiliary voltage detector (AVD) thresholds

Symbol	Parameter	Conditions(1)	Min.	Typ.	Max.	Unit
$V_{IT + (AVD)}$	1 = >0 AVDF flag toggle threshold (V_{DD} rise)	High threshold	3.85(2)	4.45	4.90	V
$V_{IT - (AVD)}$	0 = >1 AVDF flag toggle threshold (V_{DD} fall)	High threshold	3.80	4.40	4.85(2)	
V_{hys}	AVD voltage threshold hysteresis	$V_{IT + (AVD)} - V_{IT - (AVD)}$		150		mV
ΔV_{IT-}	Voltage drop between AVD flag set and LVD reset activation	V_{DD} fall		0.45		V

1. LVD functionality guaranteed only within the V_{DD} operating range specified in *Figure 70: Pin loading conditions on page 178*
2. Not tested in production

13.3.4 Internal RC oscillator and PLL

The ST7 internal clock can be supplied by an internal RC oscillator and PLL (selectable by option byte).

Table 107. Internal RC oscillator and PLL

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
$V_{DD(RC)}$	Internal RC Oscillator operating voltage	Refer to operating range of V_{DD} with T_A, *Figure 70: Pin loading conditions on page 178*	3.0		5.5	V
$V_{DD(x8PLL)}$	x8 PLL operating voltage		3.6		5.5	

13.4 Supply current characteristics

The following current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock source current consumption. To obtain the total device consumption, the two current values must be added (except for halt mode for which the clock is stopped).

13.4.1 Supply current

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

Table 108. Supply current

Symbol	Parameter	Conditions		Typ.	Max.	Unit
I_{DD}	Supply current in run mode	(1)	f_{CPU} = 8 MHz(2)	6	9	mA
	Supply current in wait mode		f_{CPU} = 8 MHz(3)	2.4	4	
	Supply current in slow mode		f_{CPU} = 250 kHz(4)	0.7	1.1	
	Supply current in slow wait mode		f_{CPU} = 250 kHz(5)	0.6	1	
I_{DD}	Supply current in halt mode(6)	(1)	-40°C ≤ T_A ≤ +85°C	<1	10	µA
			-40°C ≤ T_A ≤ +125°C		50	
I_{DD}	Supply current in AWUFH mode(7)(8)(9)	(1)	-40°C ≤ T_A ≤ +85°C	20	50	µA
			-40°C ≤ T_A ≤ +125°C		300	
I_{DD}	Supply current in active halt mode	(1)	-40°C ≤ T_A ≤ +125°C	0.7	1	mA

1. V_{DD} = 5.5 V
2. CPU running with memory access, all I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.
3. All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.
4. Slow mode selected with f_{CPU} based on f_{OSC} divided by 32. All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.
5. Slow-wait mode selected with f_{CPU} based on f_{OSC} divided by 32. All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.
6. All I/O pins in output mode with a static value at V_{SS} (no load), LVD disabled. Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.
7. All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load). Data tested in production at V_{DD} max. and f_{CPU} max.
8. This consumption refers to the Halt period only and not the associated run period which is software dependent.
9. If low consumption is required, AWUFH mode is recommended.

Figure 78. Typical I_{DD} in run mode vs. f_{CPU}

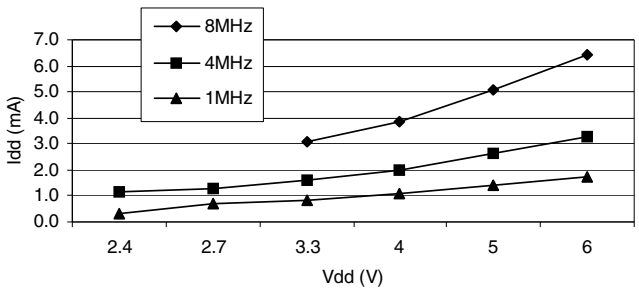


Figure 79. Typical I_{DD} in slow mode vs. f_{CPU}

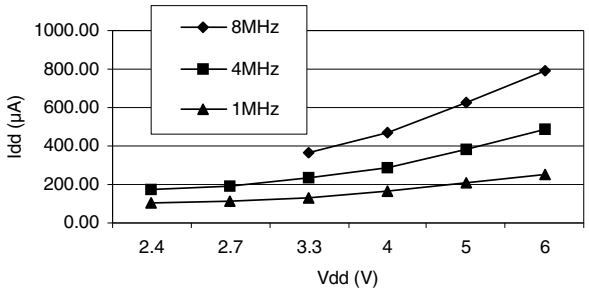


Figure 80. Typical I_{DD} in wait mode vs. f_{CPU}

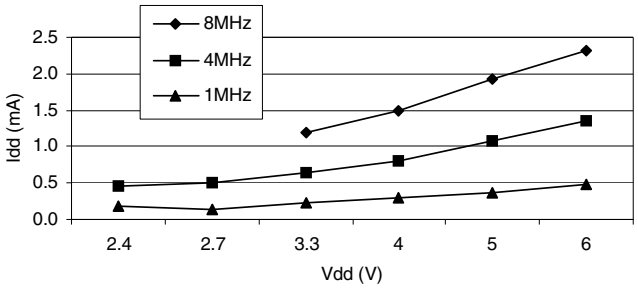


Figure 81. Typical I_{DD} in slow-wait mode vs. f_{CPU}

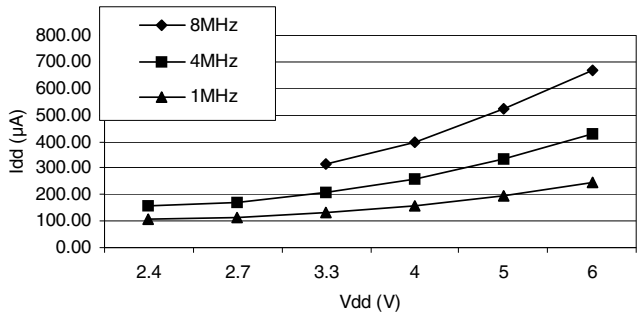


Figure 82. Typical I_{DD} vs. temperature at V_{DD} = 5 V and f_{CLKIN} = 16 MHz

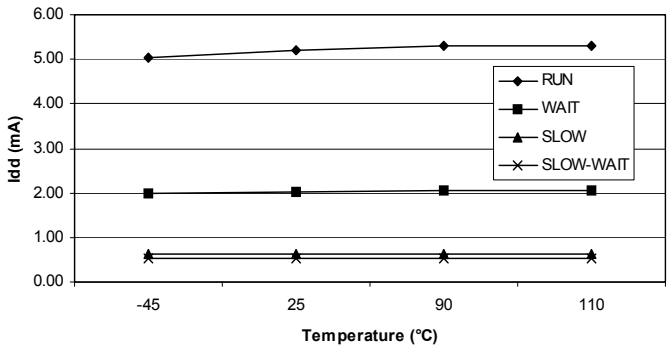
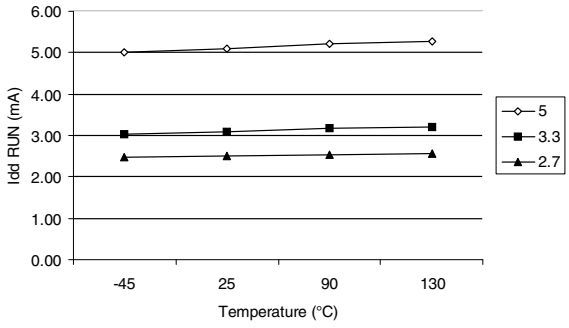


Figure 83. Typical I_{DD} vs. temperature and V_{DD} at f_{CLKIN} = 16 MHz



13.4.2 On-chip peripherals

Table 109. On-chip peripherals

Symbol	Parameter	Conditions		Typ.	Unit
$I_{DD(AT)}$	12-bit autoreload timer supply current(1)	f_{CPU} = 4 MHz	V_{DD} = 3.0 V	150	µA
		f_{CPU} = 8 MHz	V_{DD} = 5 V	1000	
$I_{DD(SPI)}$	SPI supply current(2)	f_{CPU} = 4 MHz	V_{DD} = 3.0 V	50	
		f_{CPU} = 8 MHz	V_{DD} = 5 V	200	
$I_{DD(ADC)}$	ADC supply current when converting(3)	f_{ADC} = 4 MHz	V_{DD} = 3.0 V	250	
			V_{DD} = 5 V	1100	
$I_{DD(LINSCI)}$	LINSCI supply current when transmitting(4)	f_{CPU} = 8 MHz	V_{DD} = 5.0V	650	µA

1. Data based on a differential I_{DD} measurement between reset configuration (timer stopped) and a timer running in PWM mode at f_{CPU} = 8 MHz.
2. Data based on a differential I_{DD} measurement between reset configuration and a permanent SPI master communication (data sent equal to 55h).
3. Data based on a differential I_{DD} measurement between reset configuration and continuous A/D conversions.
4. Data based on a differential I_{DD} measurement between LINSCI running at maximum speed configuration (500 Kbaud, continuous transmission of AA +RE enabled and LINSCI off. This measurement includes the pad toggling consumption.

13.5 Clock and timing characteristics

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

13.5.1 General timings

Table 110. General timings

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

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. $Dt_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.

13.5.2 Crystal and ceramic resonator oscillators

The ST7 internal clock can be supplied with four different crystal/ceramic resonator oscillators. All the information given in this paragraph are based on characterization results with specified typical external components. In the application, 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. Refer to the crystal/ceramic resonator manufacturer for more details (frequency, package, accuracy...).

Table 111. Oscillator parameters

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
f_{CrOSC}	Crystal oscillator frequency(1)		2		16	MHz
C_{L1} C_{L2}	Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator (R_S)		See *Table 112: Typical ceramic resonator characteristics*			pF

1. When PLL is used, please refer to *Section 7: Supply, reset and clock management* (f_{CrOSC} min. is 8 MHz with PLL).

Table 112. Typical ceramic resonator characteristics

Supplier	f_{CrOSC} (MHz)	Typical ceramic resonators(1)		CL1 [pF]	CL2 [pF]	Supply voltage range (V)
		Reference(2)	Oscillator modes			
Murata	2	CSTCC2M00G56-R0	LP or MP	(47)	(47)	3.0 V to 5.5 V
	4	CSTCR4M00G55-R0	MP or MS	(39)	(39)	
	8	CSTCE8M00G55-R0	MS or HS	(33)	(33)	
	16	CSTCE16M0V53-R0	HS	(15)	(15)	

1. Resonator characteristics given by the ceramic resonator manufacturer. For more information on these resonators, please consult www.murata.com
2. SMD = [-R0: Plastic tape package (∅ = 180mm), -B0: Bulk]

13.6 Memory characteristics

13.6.1 RAM and hardware registers

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

Table 113. RAM and hardware registers

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
V_{RM}	Data retention mode(1)	Halt mode (or reset)	1.6			V

1. Minimum V_{DD} supply voltage without losing data stored in RAM (in halt mode or under reset) or in hardware registers (only in halt mode). Not tested in production.

13.6.2 Flash program memory

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

Table 114. Characteristics of dual voltage HDFlash memory

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
V_{DD}	Operating voltage for Flash write/erase	Refer to operating range of V_{DD} with T_A, *Section 13.3.1: General operating conditions on page 182*	3.0		5.5	V
t_{prog}	Programming time for 1~32 bytes(1)	T_A = –40 to +85 °C		5	10	ms
	Programming time for 1.5 Kbytes	T_A = 25 °C		0.24	0.48	s
t_{RET}(2)	Data retention	T_A = 55 °C(3)	20			Years
N_{RW}	Write erase cycles	T_{PROG} = 25 °C	1K			Cycles
		T_{PROG} = 85 °C	300			
I_{DD}	Supply current	Read/write/erase modes f_{CPU} = 8 MHz, V_{DD} = 5.5 V			2.6(4)	mA
		No read/no write mode			100	µA
		Power down mode/halt		0	0.1	

1. Up to 32 bytes can be programmed at a time
2. Data based on reliability test results and monitored in production
3. The data retention time increases when the T_A decreases
4. Guaranteed by design. Not tested in production

13.6.3 EEPROM data memory

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

Table 115. Characteristics of EEPROM data memory

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
V_{DD}	Operating voltage for EEPROM write/erase	Refer to operating range of V_{DD} with T_A, *Section 13.3.1: General operating conditions on page 182*	3.0		5.5	V
t_{prog}	Programming time for 1~32 bytes	T_A = –40 to +125 °C		5	10	ms
t_{RET}(1)	Data retention with 1 k cycling (T_{PROG} = –40 to +125 °C	T_A = 55 °C(2)	20			Years
	Data retention with 10 k cycling (T_{PROG} = –40 to +125 °C)		10			
	Data retention with 100 k cycling (T_{PROG} = –40 to +125 °C)		1			

1. Data based on reliability test results and monitored in production
2. The data retention time increases when the T_A decreases

13.7 Electromagnetic compatibility (EMC) characteristics

Susceptibility tests are performed on a sample basis during product characterization.

13.7.1 Functional electromagnetic susceptibility (EMS)

Based on a simple running application on the product (toggling 2 LEDs through I/O ports), the product is stressed by two electromagnetic events until a failure occurs (indicated by the LEDs). See *Table 116: Electromagnetic test results on page 197*.

- ESD: Electro-static discharge (positive and negative) is applied on all pins of the device until a functional disturbance occurs. This test conforms with the IEC 1000 - 4 - 2 standard.
- FTB: A burst of fast transient voltage (positive and negative) is applied to V_{DD} and V_{SS} through a 100pF capacitor, until a functional disturbance occurs. This test conforms with the IEC 1000 - 4 - 4 standard.

A device reset allows normal operations to be resumed. The test results are given in the table below based on the EMS levels and classes defined in application note AN1709.

Designing hardened software to avoid noise problems

EMC characterization and optimization are performed at component level with a typical application environment and simplified MCU software. It should be noted that good EMC performance is highly dependent on the user application and the software in particular.

Therefore it is recommended that the user applies EMC software optimization and prequalification tests in relation with the EMC level requested for his application.

Software recommendations

The software flowchart must include the management of runaway conditions such as:

- Corrupted program counter
- Unexpected reset
- Critical data corruption (control registers...)

Prequalification trials

Most of the common failures (unexpected reset and program counter corruption) can be reproduced by manually forcing a low state on the reset pin or the oscillator pins for 1 second.

To complete these trials, ESD stress can be applied directly on the device, over the range of specification values. When unexpected behavior is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

Table 116. Electromagnetic test results

Symbol	Parameter	Conditions	Level/ class
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	V_{DD} = 5 V, T_A = 25 °C, f_{OSC} = 8 MHz, conforms to IEC 1000-4-2	3B
V_{FFTB}	Fast transient voltage burst limits to be applied through 100 pF on V_{DD} and V_{DD} pins to induce a functional disturbance	V_{DD} = 5 V, T_A = 25 °C, f_{OSC} = 8 MHz, conforms to IEC 1000-4-4	

13.7.2 Electromagnetic interference (EMI)

Based on a simple application running on the product (toggling 2 LEDs through the I/O ports), the product is monitored in terms of emission. This emission test is in line with the norm SAE J 1752/3 which specifies the board and the loading of each pin. See *Table 117: EMI emissions on page 197*.

Table 117. EMI emissions

Sym.	Parameter	Conditions	Monitored frequency band	Max. vs. [f_{OSC}/f_{CPU}]		Unit
				8/4 MHz	16/8 MHz	
S_{EMI}	Peak level(1)	V_{DD} = 5 V, T_A = 25 °C, SO20 package, conforming to SAE J 1752/3	0.1MHz to 30 MHz	15	15	dBµV
			30 MHz to 130 MHz	13	19	
			130 MHz to 1 GHz	9	13	
			SAE EMI level	2.5	3	-

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

13.7.3 Absolute maximum ratings (electrical sensitivity)

Based on three different tests (ESD, DLU and LU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity. For more details, refer to the application note AN1181.

Electrostatic discharge (ESD)

Electrostatic discharges (a positive then a negative pulse separated by 1 second) are applied to the pins of each sample according to each pin combination. The sample size depends on the number of supply pins in the device (3 parts*(n+1) supply pin). Two models can be simulated: the human body model and the machine model. This test conforms to the JESD22-A114A/A115A standard.

Table 118. ESD absolute maximum ratings

Symbol	Ratings	Conditions	Maximum value(1)	Unit
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human body model)	T_A = +25 °C	4000	V
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine model)		400	
$V_{ESD(CDM)}$	Electro-static discharge voltage (Charged device model)		1000	

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

Static and dynamic latch-up (LU)

LU: Three complementary static tests are required on 10 parts to assess the latch-up performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latch-up standard. For more details, refer to application note AN1181.

DLU: Electro-Static Discharges (one positive then one negative test) are applied to each pin of three samples when the micro is running to assess the latch-up performance in dynamic mode. Power supplies are set to the typical values, the oscillator is connected as near as possible to the pins of the micro and the component is put in reset mode. This test conforms to the IEC1000-4-2 and SAEJ1752/3 standards. For more details, refer to the application note AN1181.

Electrical sensitivities

Table 119. Latch up results

Symbol	Parameter	Conditions	Class(1)
LU	Static latch-up class	T_A = 25 °C T_A = 125 °C	A
DLU	Dynamic latch-up class	V_{DD} = 5.5 V, f_{OSC} = 4 MHz, T_A = 25 °C	

1. Class description: A class is an STMicroelectronics internal specification. All its limits are higher than the JEDEC specifications, which means when a device belongs to class A it exceeds the JEDEC standard. Class B strictly covers all the JEDEC criteria (international standard). Class B strictly covers all the JEDEC criteria (international standard)

13.8 I/O port pin characteristics

13.8.1 General characteristics

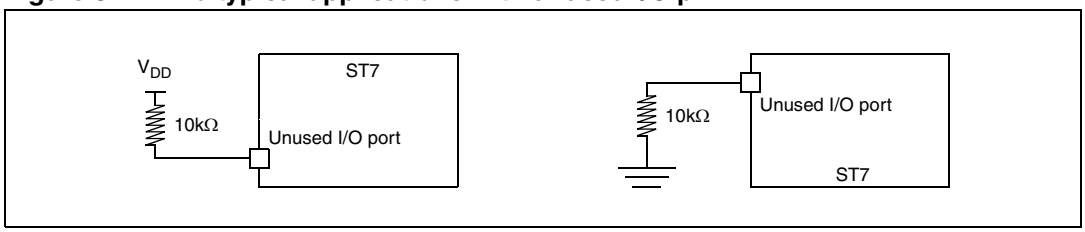
Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A (-40 to +125°C), unless otherwise specified.

Table 120. I/O general port pin characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
V_{IL}	Input low level voltage		V_{SS} - 0.3		0.3 x V_{DD}	V
V_{IH}	Input high level voltage		0.7 x V_{DD}		V_{DD} + 0.3	
V_{hys}	Schmitt trigger voltage hysteresis(1)			400		mV
I_L	Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			±1	µA
I_S	Static current consumption induced by each floating input pin(2)	Floating input mode		400		
R_{PU}	Weak pull-up equivalent resistor(3)	$V_{IN} = V_{SS}$, V_{DD} = 5 V	50	100	170	kΩ
C_{IO}	I/O pin capacitance			5		pF
$t_{f(IO)out}$	Output high to low level fall time(1)	C_L = 50 pF between 10% and 90%		25		ns
$t_{r(IO)out}$	Output low to high level rise time(1)					
$t_{w(IT)in}$	External interrupt pulse time(4)		1			t_{CPU}

1. Data based on characterization results, not tested in production
2. Configuration not recommended, all unused pins must be kept at a fixed voltage: Using the output mode of the I/O for example or an external pull-up or pull-down resistor (see *Figure 84*). Static peak current value taken at a fixed V_{IN} value, based on design simulation and technology characteristics, not tested in production. This value depends on V_{DD} and temperature values
3. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in *Figure 84: Two typical applications with unused I/O pin on page 199*)
4. To generate an external interrupt, a minimum pulse width must be applied on an I/O port pin configured as an external interrupt source.

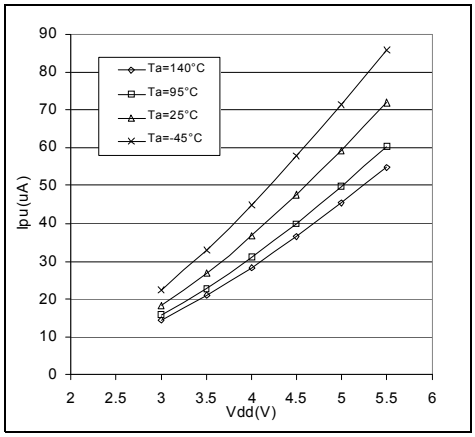
Figure 84. Two typical applications with unused I/O pin



1. I/O can be left unconnected if it is configured as output (0 or 1) by the software. This has the advantage of greater EMC robustness and lower cost.

Caution: During normal operation the ICCCLK pin must be pulled up, internally or externally (external pull-up of 10 k mandatory in noisy environments). This is to avoid entering ICC mode unexpectedly during a reset.

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



13.8.2 Output driving current

Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A (-40 to +125 °C) unless otherwise specified.

Table 121. Output driving current

Symbol	Parameter	Conditions		Min.	Typ.	Max.	Unit
V_{OL}(1)	Output low level voltage for a standard I/O pin when eight pins are sunk at same time (see *Figure 88*)	V_{DD} = 5 V	I_{IO} = +5 mA			1.0	V
			I_{IO} = +2 mA			0.4	
	Output low level voltage for a high sink I/O pin when four pins are sunk at same time (see *Figure 94*)		I_{IO} = +20 mA			1.4	
			I_{IO} = +8 mA			0.75	
V_{OH}(2)	Output high level voltage for an I/O pin when four pins are sourced at same time (see *Figure 94*)		I_{IO} = -5 mA	V_{DD} - 1.5			
			I_{IO} = -2 mA	V_{DD} - 1.0			

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in *Section 13.2.2: Current characteristics on page 180* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}
2. The I_{IO} current sourced must always respect the absolute maximum rating specified in *Section 13.2.2: Current characteristics on page 180* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}.

Figure 86. Typical V_{OL} at V_{DD} = 3 V

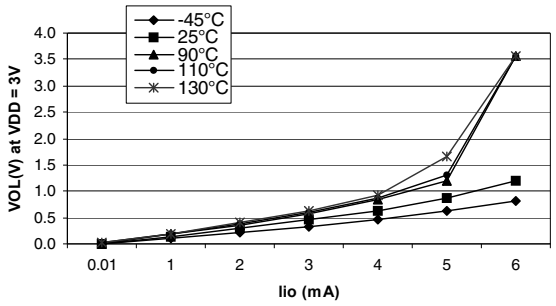


Figure 87. Typical V_{OL} at V_{DD} = 4 V

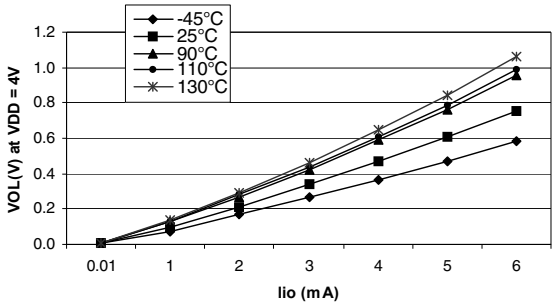


Figure 88. Typical V_{OL} at V_{DD} = 5 V

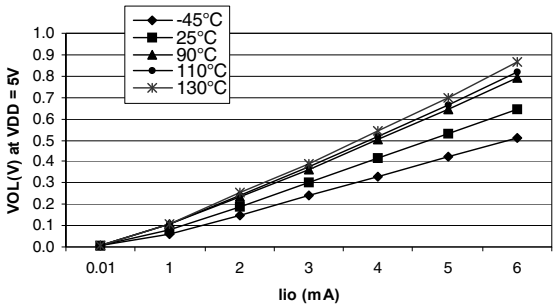


Figure 89. Typical V_{OL} at V_{DD} = 3 V (high-sink)

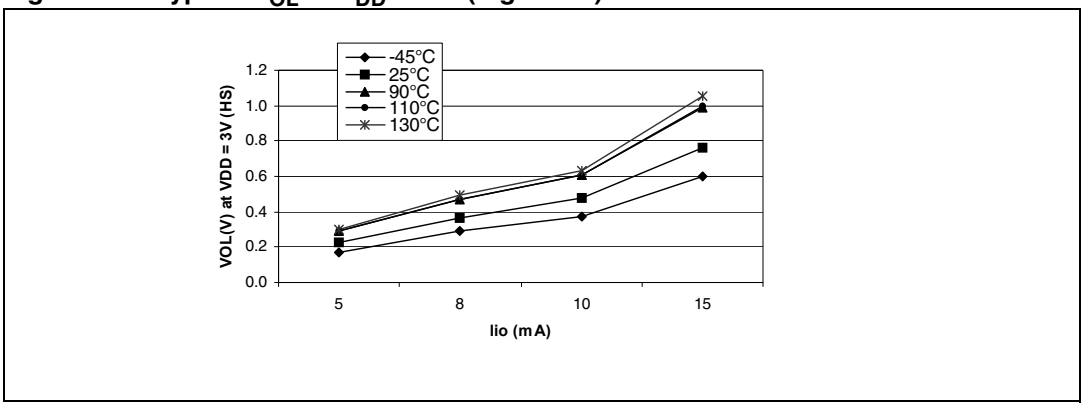


Figure 90. Typical V_{OL} at V_{DD} = 4 V (high-sink)

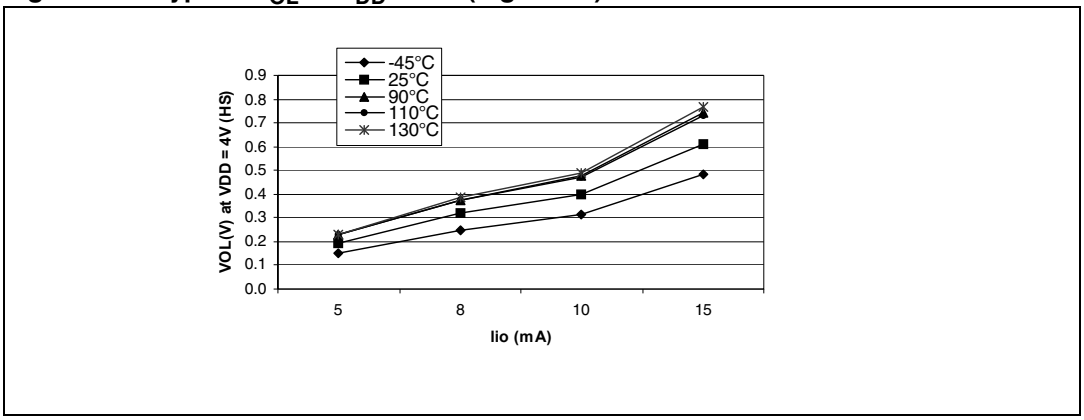


Figure 91. Typical V_{OL} at V_{DD} = 5 V (high-sink)

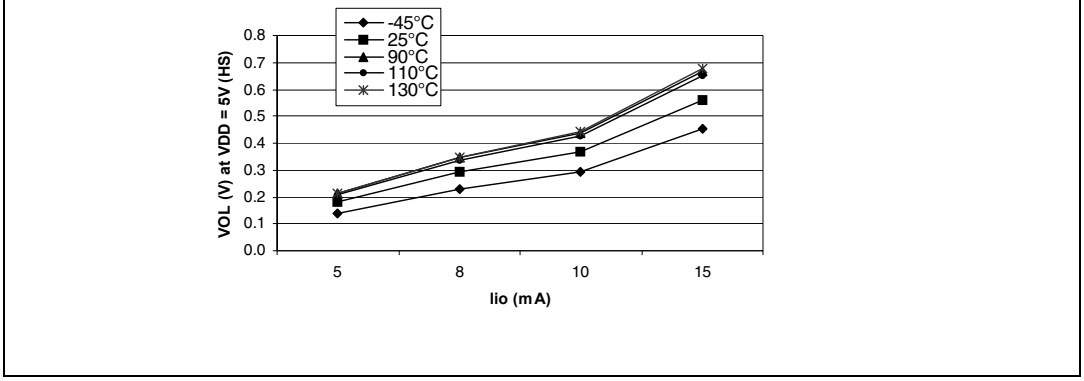


Figure 92. Typical V_{DD} - V_{OH} at V_{DD} = 3 V

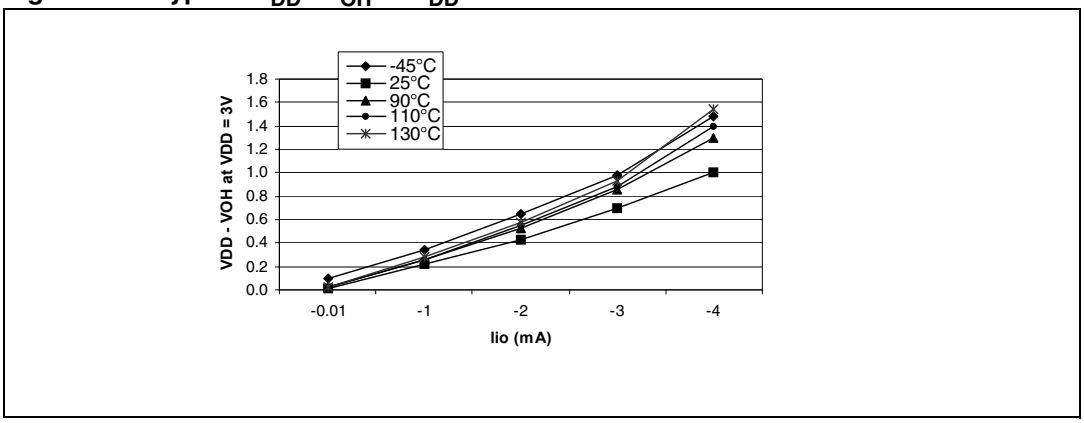


Figure 93. Typical V_{DD} - V_{OH} at V_{DD} = 4 V

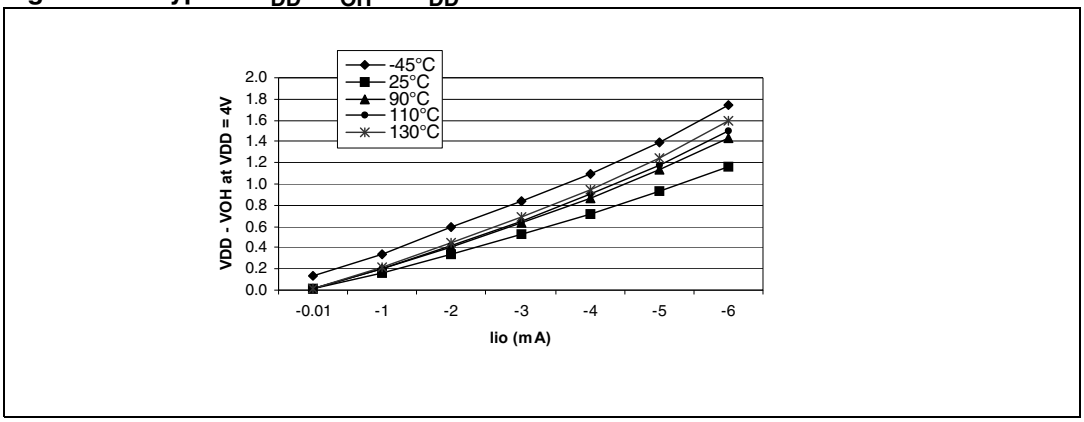


Figure 94. Typical V_{DD} - V_{OH} at V_{DD} = 5 V

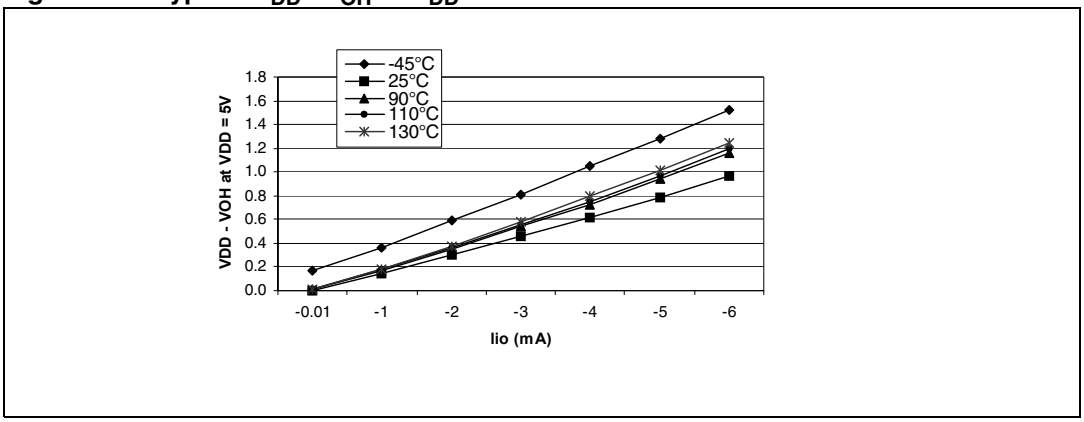
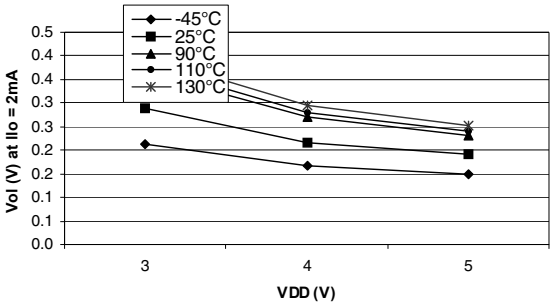


Figure 95. Typical V_{OL} vs. V_{DD} (standard I/Os)



Typical V_{OL} vs. V_{DD} (standard I/Os)

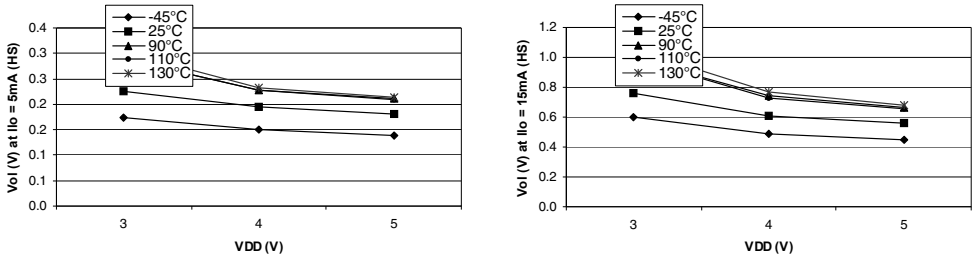
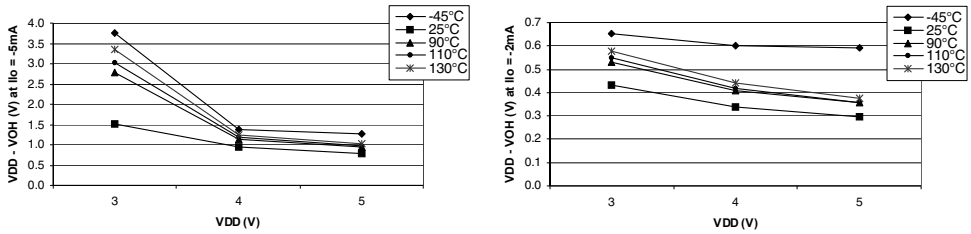


Figure 96. Typical V_{DD} - V_{OH} vs. V_{DD}



13.9 Control pin characteristics

13.9.1 Asynchronous $\overline{\text{RESET}}$ pin

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

Table 122. Asynchronous $\overline{\text{RESET}}$ pin

Symbol	Parameter	Conditions		Min.	Typ.	Max.	Unit
V_{IL}	Input low-level voltage			V_{ss} - 0.3		0.3 x V_{DD}	V
V_{IH}	Input high-level voltage			0.7 x V_{DD}		V_{DD} + 0.3	
V_{hys}	Schmitt trigger voltage hysteresis(1)				1		
V_{OL}	Output low-level voltage(1)	V_{DD} = 5 V	I_{IO} = +5 mA, $T_A \leq$ +85 °C $T_A \leq$ +125 °C		0.5	1.0(2) 1.2(2)	
			I_{IO} = +2 mA, $T_A \leq$ +85 °C $T_A \leq$ +125 °C		0.45	0.7(2) 0.9(2)	
R_{ON}	Pull-up equivalent resistor(1)(3)	V_{DD} = 5 V		10	39	70	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	Internal reset sources			30		µs
$t_{h(RSTL)in}$	External reset pulse hold time(4)			20			
$t_{g(RSTL)in}$	Filtered glitch duration				200		ns

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in *Section 13.2.2: Current characteristics on page 180* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
2. Guaranteed by design. Not tested in production.
3. The R_{ON} pull-up equivalent resistor is based on a resistive transistor. Specified for voltages on $\overline{\text{RESET}}$ pin between V_{ILmax} and V_{DD}.
4. To guarantee the reset of the device, a minimum pulse must be applied to the $\overline{\text{RESET}}$ pin. All short pulses applied on $\overline{\text{RESET}}$ pin with a duration below $t_{h(RSTL)in}$ can be ignored.

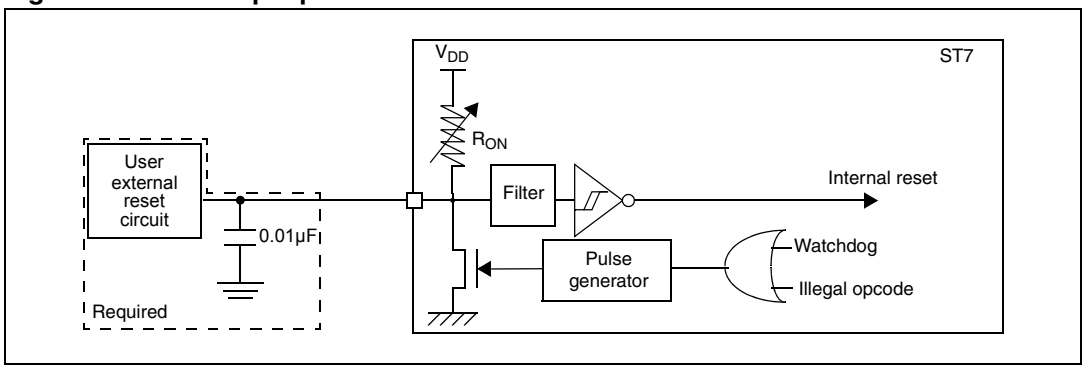
$\overline{\text{RESET}}$ circuit design recommendations

The reset network protects the device against parasitic resets. The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog). Whatever the reset source is (internal or external), the user must ensure that the level on the $\overline{\text{RESET}}$ pin can go below the V_{IL} level specified in *Section 13.9.1: Asynchronous RESET pin on page 205*. Otherwise the reset is not taken into account internally. Because the reset circuit is designed to allow the internal reset to be output in the $\overline{\text{RESET}}$ pin, the user must ensure that the current sunk on the $\overline{\text{RESET}}$ pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in *Section 13.2.2: Current characteristics on page 180*.

Refer to *Section 12.2.2: Illegal opcode reset on page 175* for details on illegal opcode reset conditions.

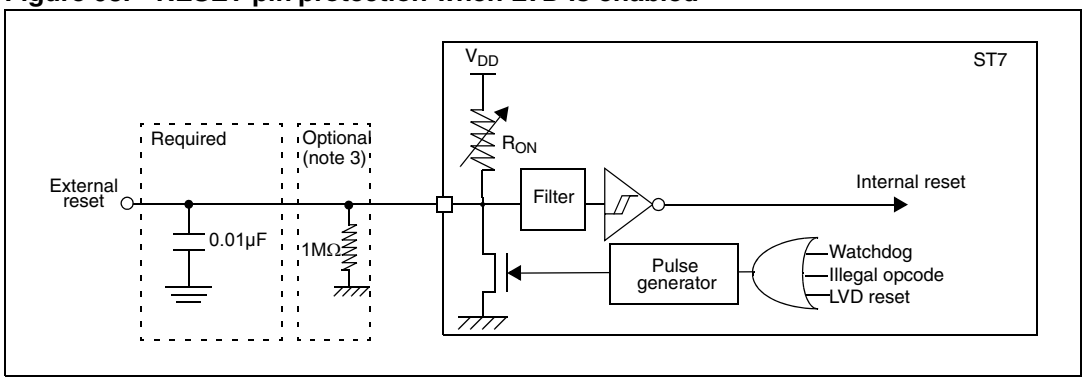
$\overline{\text{RESET}}$ pin protection when LVD is disabled

Figure 97. $\overline{\text{RESET}}$ pin protection when LVD is disabled



$\overline{\text{RESET}}$ pin protection when LVD is enabled

Figure 98. $\overline{\text{RESET}}$ pin protection when LVD Is enabled



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

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

Tips when using the LVD

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

13.10 Communication interface characteristics

13.10.1 Serial peripheral interface (SPI)

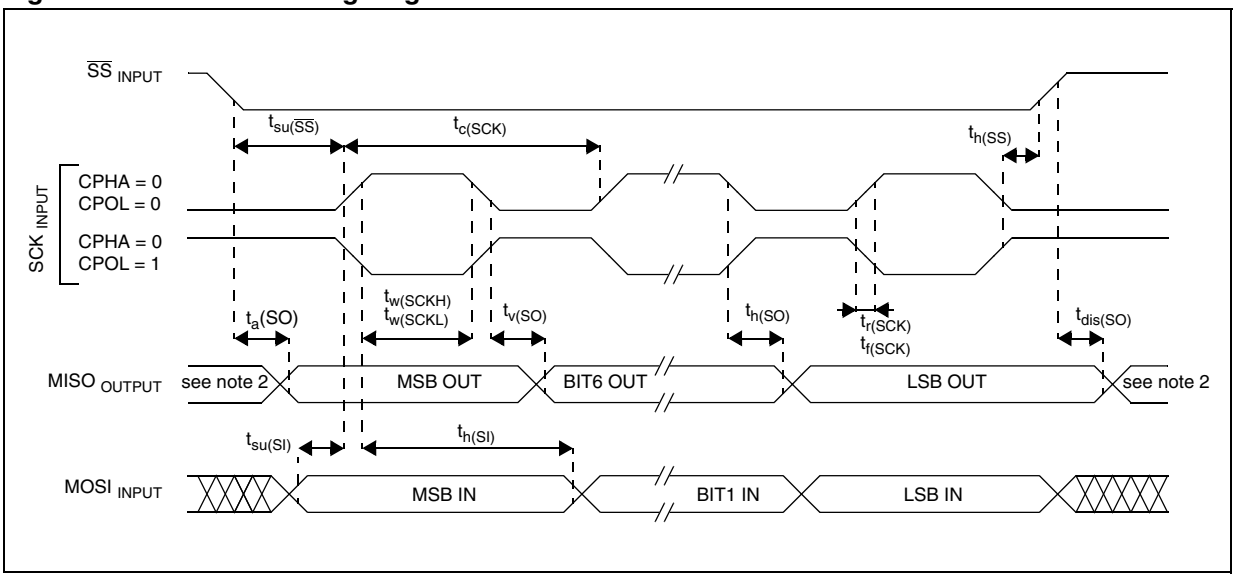
Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified. Refer to *Section 10: I/O ports* for more details on the input/output alternate function characteristics ($\overline{SS}$, SCK, MOSI, MISO).

Table 123. SPI characteristics

Symbol	Parameter	Conditions	Min.	Max.	Unit
$f_{SCK} = 1/t_{c(SCK)}$	SPI clock frequency	Master, f_{CPU} = 8 MHz	$f_{CPU}/128 = 0.0625$	$f_{CPU}/4 = 2$	MHz
		Slave, f_{CPU} = 8 MHz	0	$f_{CPU}/2 = 4$	
$t_{r(SCK)}$	SPI clock rise and fall time		See *Table 2: Device pin description on page 17*		
$t_{f(SCK)}$					
$t_{su(\overline{SS})}$ (1)	$\overline{SS}$ setup time(2)	Slave	(4 x T_{CPU}) + 50		ns
$t_{h(\overline{SS})}$ (1)	$\overline{SS}$ hold time		120		
$t_{w(SCKH)}$ (1)	SCK high and low time	Master	100		
$t_{w(SCKL)}$ (1)		Slave	90		
$t_{su(MI)}$ (1)	Data input setup time	Master	100		
$t_{su(SI)}$ (1)		Slave			
$t_{h(MI)}$ (1)	Data input hold time	Master			
$t_{h(SI)}$ (1)		Slave			
$t_{a(SO)}$ (1)	Data output access time	Slave	0	120	
$t_{dis(SO)}$ (1)	Data output disable time			240	
$t_{v(SO)}$ (1)	Data output valid time	Slave (after enable edge)		120	
$t_{h(SO)}$ (1)	Data output hold time		0		
$t_{v(MO)}$ (1)	Data output valid time	Master (after enable edge)		120	t_{CPU}
$t_{h(MO)}$ (1)	Data output hold time		0		

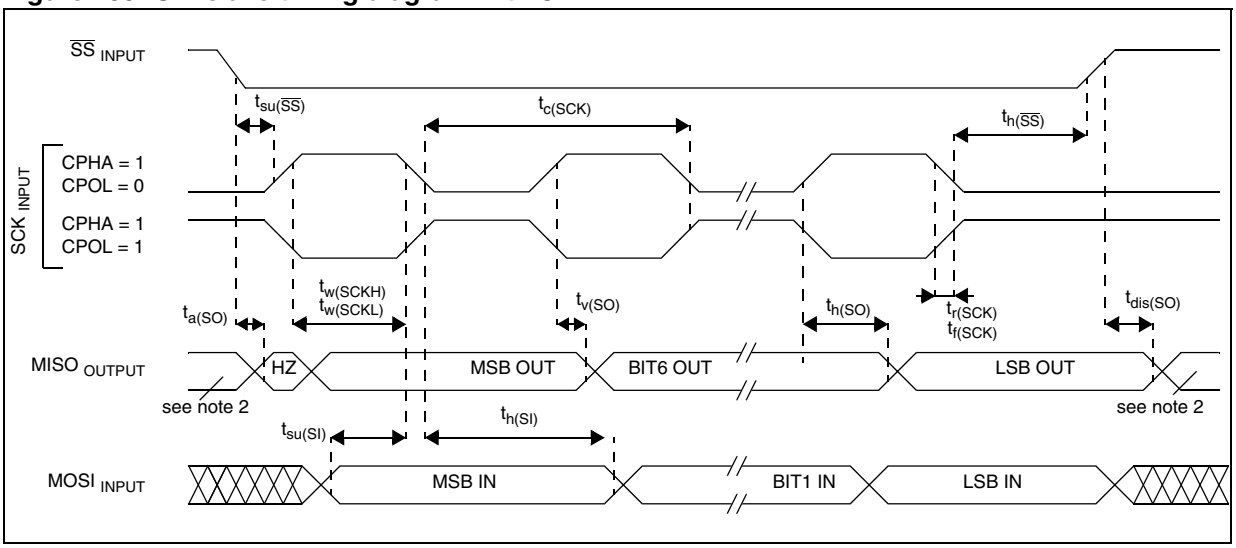
1. Data based on design simulation, not tested in production.
2. Depends on f_{CPU}. For example, if f_{CPU} = 8 MHz, then $t_{CPU} = 1/f_{CPU}$ = 125 ns and $t_{su}(\overline{SS})$ = 550 ns.

Figure 99. SPI slave timing diagram with CPHA = 0



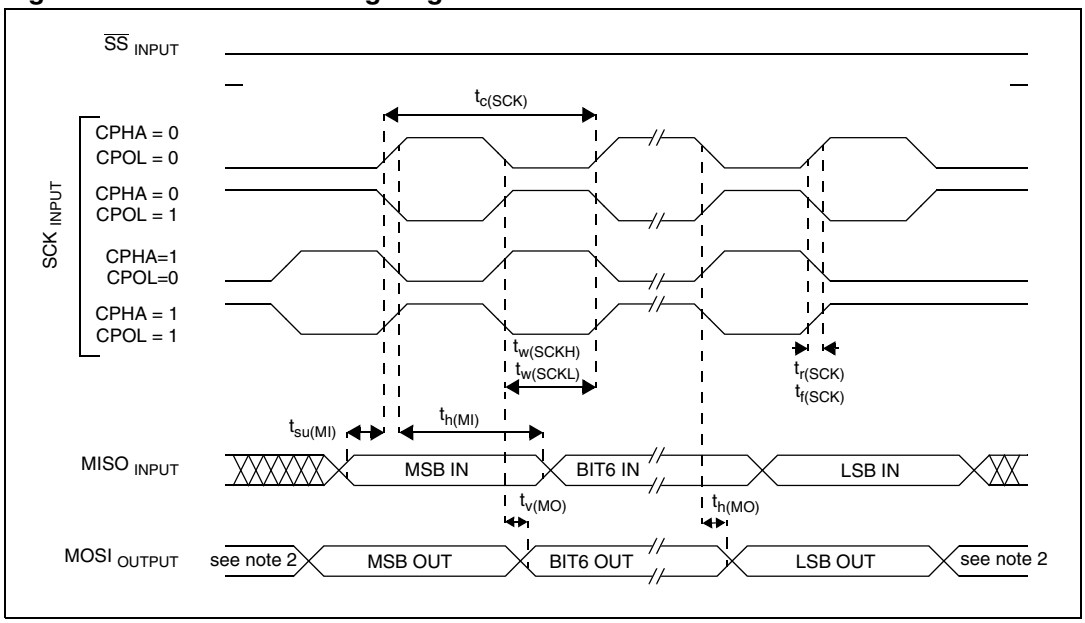
1. Measurement points are made at CMOS levels: 0.3 x V_{DD} and 0.7 x V_{DD}
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

Figure 100. SPI slave timing diagram with CPHA = 1



1. Measurement points are made at CMOS levels: 0.3 x V_{DD} and 0.7 x V_{DD}.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

Figure 101. SPI master timing diagram



1. Measurement points are done at CMOS levels: 0.3 x V_{DD} and 0.7 x V_{DD}.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

13.11 10-bit ADC characteristics

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

Table 124. 10-bit ADC characteristics

Symbol	Parameter	Conditions	Min.(1)	Typ.(2)	Max.(1)	Unit
f_{ADC}	ADC clock frequency		0.5		4	MHz
V_{AIN}	Conversion voltage range(3)		V_{SSA}		V_{DDA}	V
R_{AIN}	External input resistor				10(4)	kΩ
C_{ADC}	Internal sample and hold capacitor			6		pF
t_{STAB}	Stabilization time after ADC enable	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz	0(5)			µs
t_{ADC}	Conversion time (sample+hold)		3.5			
	- Sample capacitor loading time - Hold conversion time		4 10			$1/f_{ADC}$

1. Data based on characterization results, not tested in production
2. Unless otherwise specified, typical data is based on T_A = 25 °C and V_{DD} - V_{SS} = 5 V. They are given only as design guidelines and are not tested.
3. When V_{DDA} and V_{SSA} pins are not available on the pinout, the ADC refers to V_{DD} and V_{SS}
4. Any added external serial resistor downgrades the ADC accuracy (especially for resistance greater than 10kΩ). Data based on characterization results, not tested in production.
5. The stabilization time of the AD converter is masked by the first t_{LOAD}. The first conversion after the enable is then always valid.

Figure 102. Typical application with ADC

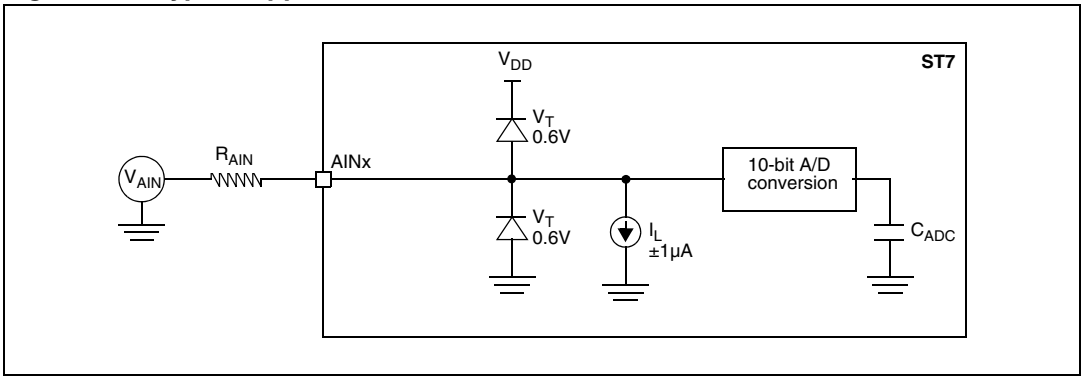


Table 125. ADC accuracy with 4.5 V ≤ V_{DD} ≤ 5.5 V

Symbol	Parameter	Conditions	Typ.	Max.(1)	Unit
$\|E_T\|$	Total unadjusted error	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz(2)(3)	2.0	3.4	LSB
$\|E_O\|$	Offset error		0.4	1.7	
$\|E_G\|$	Gain error		0.4	1.5	
$\|E_D\|$	Differential linearity error		1.9	3.1	
$\|E_L\|$	Integral linearity error		1.8	2.9	

1. Data based on characterization results, monitored in production to guarantee 99.73% within ± max value from -40 °C to +125 °C (± 3σ distribution limits).

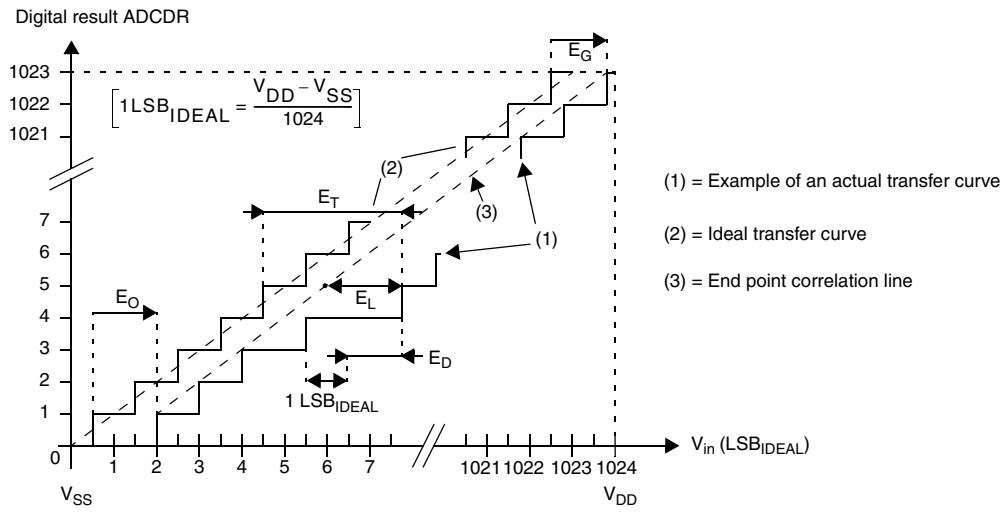
2. Data based on characterization results over the whole temperature range, monitored in production.
3. ADC accuracy vs. negative injection current: Injecting negative current on any of the analog input pins may reduce the accuracy of the conversion being performed on another analog input.
The effect of negative injection current on robust pins is specified in *Section 13.11: 10-bit ADC characteristics on page 210*
Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in *Section 13.8: I/O port pin characteristics on page 199* does not affect the ADC accuracy.

Table 126. ADC accuracy with 3 V ≤ V_{DD} ≤ 3.6 V

Symbol	Parameter	Conditions	Typ.	Max.(1)	Unit
$\|E_T\|$	Total unadjusted error	f_{CPU} = 4 MHz, f_{ADC} = 2 MHz(2)(3)	1.9	3.1	LSB
$\|E_O\|$	Offset error		0.3	1.2	
$\|E_G\|$	Gain error		0.3	1	
$\|E_D\|$	Differential linearity error		1.8	3	
$\|E_L\|$	Integral linearity error		1.7	2.8	

1. Data based on characterization results, monitored in production to guarantee 99.73% within ± max value from -40 °C to +125 °C (± 3σ distribution limits).
2. Data based on characterization results over the whole temperature range, monitored in production.
3. ADC accuracy vs. negative injection current: Injecting negative current on any of the analog input pins may reduce the accuracy of the conversion being performed on another analog input.
The effect of negative injection current on robust pins is specified in *Section 13.11: 10-bit ADC characteristics on page 210*
Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in *Section 13.8: I/O port pin characteristics on page 199* does not affect the ADC accuracy.

Figure 103. ADC accuracy characteristics



1. Legend:
E_T = Total unadjusted error: maximum deviation between the actual and the ideal transfer curves
E_O = Offset error: deviation between the first actual transition and the first ideal one
E_G = Gain error: deviation between the last ideal transition and the last actual one
E_D = Differential linearity error: maximum deviation between actual steps and the ideal one
E_L = Integral linearity error: maximum deviation between any actual transition and the end point correlation line

14 Package characteristics

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.

14.1 Package mechanical data

Figure 104. 20-pin plastic small outline package, 300-mil width

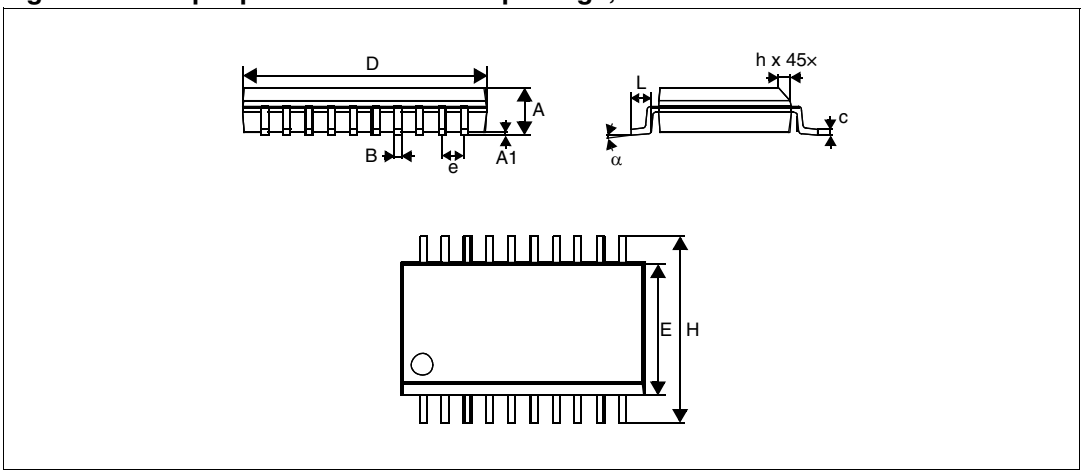


Table 127. 20-pin plastic small outline package, 300-mil width, mechanical data

Dim.	mm			inches		
	Min.	Typ.	Max.	Min.	Typ.	Max.
A	2.35		2.65	0.093		0.104
A1	0.10		0.30	0.004		0.012
B	0.33		0.51	0.013		0.020
C	0.23		0.32	0.009		0.013
D	12.60		13.00	0.496		0.512
E	7.40		7.60	0.291		0.299
e		1.27			0.050	
H	10.00		10.65	0.394		0.419
h	0.25		0.75	0.010		0.030
α	0°		8°	0°		8°
L	0.40		1.27	0.016		0.050

Figure 105. QFN 5x6, 20-terminal very thin fine pitch quad flat no-lead package

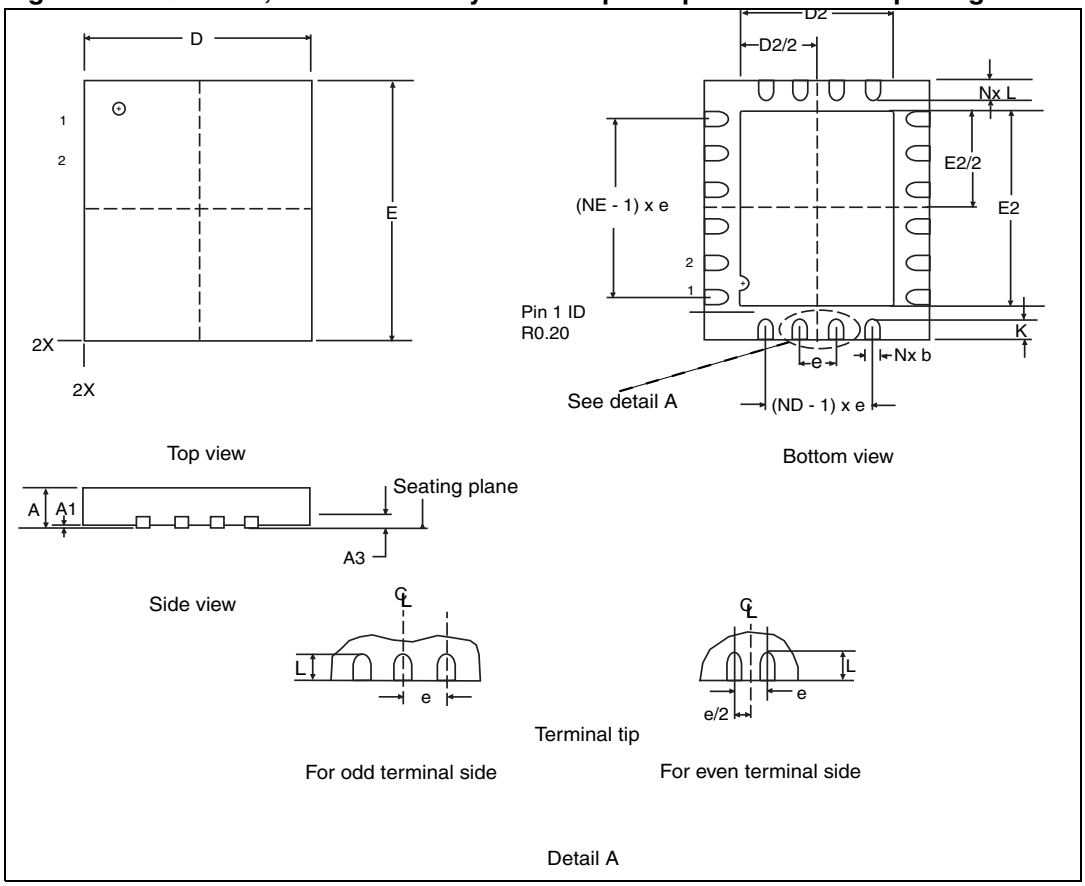


Table 128. QFN 5x6: 20-terminal very thin fine pitch quad flat no-lead package

Dim.	mm			inches		
	Min.	Typ.	Max.	Min.	Typ.	Min.
e		0.80			0.0315	
L	0.45	0.50	0.55	0.0177	0.0197	0.0217
b(1)	0.25	0.30	0.35	0.0098	0.0118	0.0138
D2	3.30	3.40	3.50	0.1299	0.1339	0.1378
E2	4.30	4.40	4.50	0.1693	0.1732	0.1772
D		5.00			0.1969	
E		6.00			0.2362	
A	0.80	0.85	0.90	0.0315	0.0335	0.0354
A1	0.00	0.02	0.05	0.0000	0.0008	0.0020
A3		0.02			0.0008	
K		0.20			0.0079	
N(2)	20					

Table 128. QFN 5x6: 20-terminal very thin fine pitch quad flat no-lead package

ND(3)	4
NE(3)	6

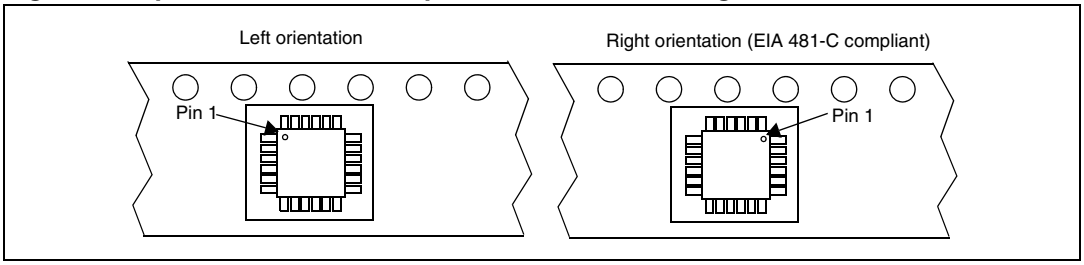
1. Dimension b applies to metallized terminals and is measured between 0.15 and 0.30 mm from terminal TIP. If the terminal has the optional radius on the other end of the terminal the dimension b should not be measured in that radius area.
2. N is the total number of terminals
3. ND and NE refer to the number of terminals on each D and E side respectively

14.2 Packaging for automatic handling

The devices can be supplied in trays or with tape and reel conditioning.

Tape and reel conditioning can be ordered with pin 1 left-oriented or right-oriented when facing the tape sprocket holes as shown in *Figure 106*.

Figure 106. pin 1 orientation in tape and reel conditioning



See also *Figure 107: ST7FL3x Flash commercial product structure on page 220* and *Figure 108: ST7FL3x FASTROM commercial product structure on page 221*.

14.3 Thermal characteristics

Table 129. Thermal characteristics

Symbol	Parameter	Package	Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient)	SO20	70	°C/W
		QFN20	30	
T_{Jmax}	Maximum junction temperature(1)	SO20	150	°C
		QFN20		
P_{Dmax}	Maximum power dissipation(2)	SO20	< 350	mW
		QFN20	< 800	

1. The maximum chip-junction temperature is based on technology characteristics
2. The maximum power dissipation is obtained from the formula $P_D = (T_J - T_A)/R_{thJA}$. The power dissipation of an application can be defined by the user with the formula: $P_D = P_{INT} + P_{PORT}$, where P_{INT} is the chip internal power (I_{DD} x V_{DD}) and P_{PORT} is the port power dissipation depending on the ports used in the application.

15 Device configuration and ordering information

15.1 Introduction

Each device is available for production in user programmable versions (Flash) as well as in factory coded versions (ROM). ST7L3x devices are ROM versions.

ST7PL3x devices are factory advanced service technique ROM (FASTROM) versions: They are factory programmed Flash devices.

ST7FL3 Flash devices are shipped to customers with a default program memory content (FFh), while ROM/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 while the ROM/FASTROM devices are factory-configured.

15.2 Option bytes

The two option bytes allow the hardware configuration of the microcontroller to be selected. Differences in option byte configuration between Flash and ROM devices are presented in *Table 130* and are described in *Section 15.2.1: Flash option bytes on page 216* and *Section 15.2.2: ROM option bytes on page 217*.

Table 130. Flash and ROM option bytes

		Option byte 0								Option byte 1							
		7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	Flash	AWUCK	OSCRANGE 2:0			SEC 1	SEC 0	FM PR	FM PW	Res	PLL OFF	Res	OSC	LVD 1:0		WDG SW	WDG HALT
	ROM					Res		ROP_R	ROP_D								
Default value		1	1	1	1	1	1	0	0	1(1)	1	0(1)	0	1	1	1	1

1. Contact your STMicroelectronics support

15.2.1 Flash option bytes

Table 131. Option byte 0 description

Bit	Bit name	Function
7	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, the internal RC oscillator must be selected (option OSC = 0).*
6:4	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. 000: Typ. frequency range with resonator (LP) = 1~2 MHz 001: Typ. frequency range with resonator (MP) = 2~4 MHz) 010: Typ. frequency range with resonator (MS) = 4~8 MHz) 011: Typ. frequency range with resonator (HS) = 8~16 MHz) 100: Typ. frequency range with resonator (VLP) = 32.768~ kHz) 101: External clock on OSC1 110: Reserved 111: External clock on PB4 *Note: 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.*
3:2	SEC[1:0]	Sector 0 size definition These option bits indicate the size of sector 0 as follows: 00: Sector 0 size = 0.5 Kbytes 01: Sector 0 size = 1 Kbyte 10: Sector 0 size = 2 Kbytes 11: Sector 0 size = 4 Kbytes
1	FMP_R	Readout 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: Memory protection on page 25* for more details. 0: Readout protection off 1: Readout protection on
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 *Note: The option bytes have no address in the memory map and are accessed only in programming mode (for example using a standard ST7 programming tool). The default content of the Flash is fixed to FFh.*

Table 132. Option byte 1 description

Bit	Bit name	Function
7	-	Reserved, must be set to 1(1)
6	PLLOFF	PLL disable This option bit enables or disables the PLL. 0: PLL enabled 1: PLL disabled (bypassed)
5	-	Reserved, must be set to 0(1)
4	OSC	RC oscillator selection This option bit enables selection of the internal RC oscillator. 0: RC oscillator on 1: RC oscillator off *Note: To improve clock stability and frequency accuracy when the RC oscillator is selected, it is recommended to place a decoupling capacitor, typically 100 nF, between the V_{DD} and V_{SS} pins as close as possible to the ST7 device.*
3:2	LVD[1:0]	Low voltage selection These option bits enable the voltage detection block (LVD and AVD) with a selected threshold to the LVD and AVD: 11: LVD off 10: LVD on (highest voltage threshold)
1	WDGSW	Hardware or software watchdog 0: Hardware (watchdog always enabled) 1: Software (watchdog to be enabled by software)
0	WDGHALT	Watchdog reset on halt 0: No reset generation when entering halt mode 1: Reset generation when entering halt mode

1. Contact your local STMicroelectronics sales office.

15.2.2 ROM option bytes

Table 133. Option byte 0 description

Bit	Bit name	Function
7	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, the internal RC oscillator must be selected (option OSC = 0).*

Table 133. Option byte 0 description

Bit	Bit name	Function
6:4	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. 000: Typ. frequency range with resonator (LP) = 1~2 MHz 001: Typ. frequency range with resonator (MP) = 2~4 MHz) 010: Typ. frequency range with resonator (MS) = 4~8 MHz) 011: Typ. frequency range with resonator (HS) = 8~16 MHz) 100: Typ. frequency range with resonator (VLP) = 32.768~ kHz) 101: External clock on OSC1 110: Reserved 111: External clock on PB4 *Note: 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*
3:2	-	Reserved, must be set to 1
1	ROP_R	Readout protection for ROM This option is for read protection of ROM 0: Readout protection off 1: Readout protection on
0	ROP_D	Readout protection for data EEPROM This option is for read protection of EEPROM memory. 0: Readout protection off 1: Readout protection on

Table 134. Option byte 1 description

Bit	Bit name	Function
7	-	Reserved, must be set to 1(1)
6	PLLOFF	PLL disable This option bit enables or disables the PLL. 0: PLL enabled 1: PLL disabled (bypassed)
5	-	Reserved, must be set to 0(1)
4	OSC	RC oscillator selection This option bit enables selection of the internal RC oscillator. 0: RC oscillator on 1: RC oscillator off *Note: To improve clock stability and frequency accuracy when the RC oscillator is selected, it is recommended to place a decoupling capacitor, typically 100 nF, between the V_{DD} and V_{SS} pins as close as possible to the ST7 device.*
3:2	LVD[1:0]	Low voltage selection These option bits enable the voltage detection block (LVD and AVD) with a selected threshold to the LVD and AVD: 11: LVD off 10: LVD on (highest voltage threshold)

Table 134. Option byte 1 description

Bit	Bit name	Function
1	WDGSW	Hardware or software watchdog 0: Hardware (watchdog always enabled) 1: Software (watchdog to be enabled by software)
0	WDGHALT	Watchdog reset on halt 0: No reset generation when entering halt mode 1: Reset generation when entering halt mode

1. Contact your STMicroelectronics support

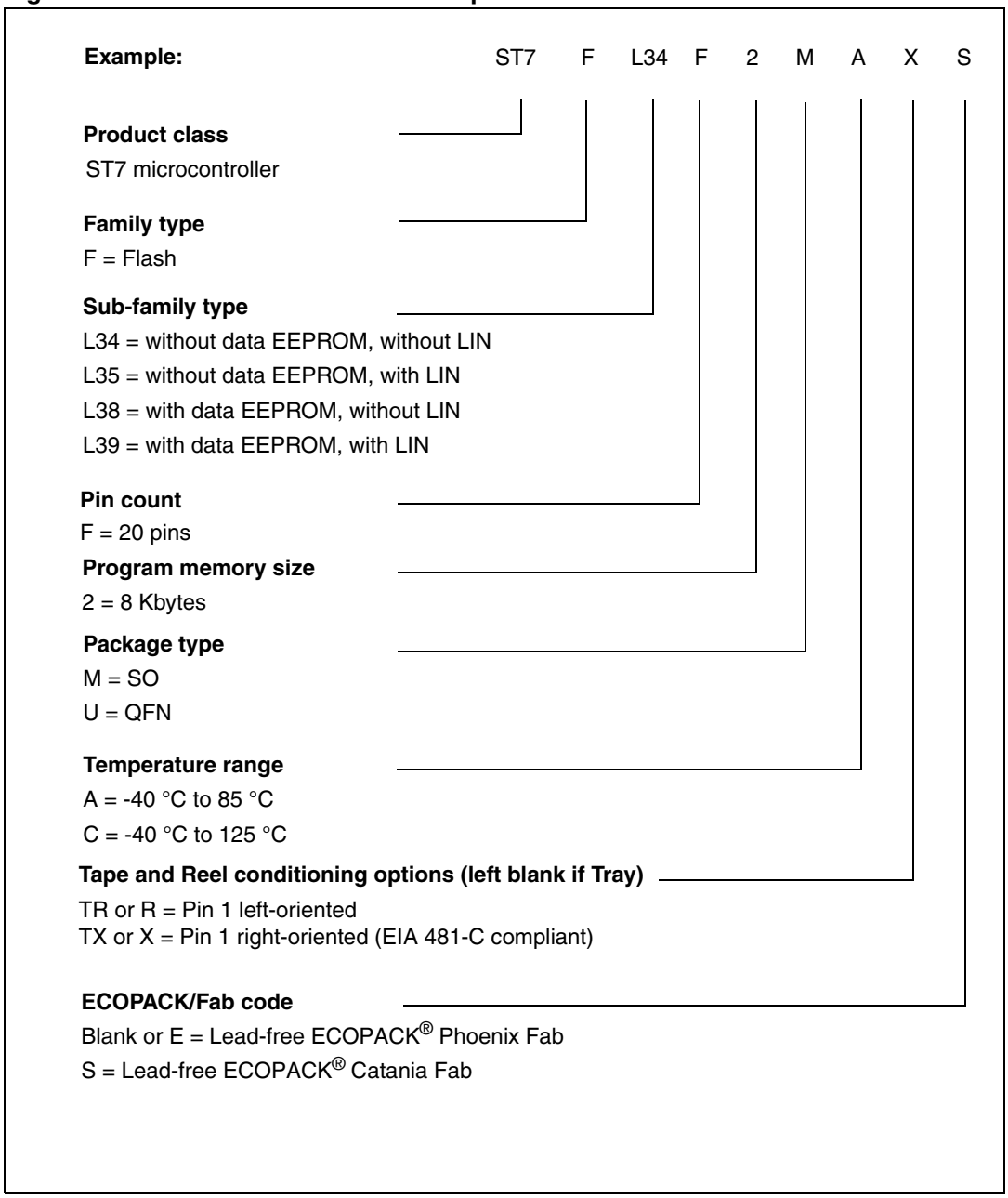
15.3 Device ordering information and transfer of customer code

Customer code is made up of the ROM/FASTROM contents and the list of the selected options (if any). The ROM/FASTROM contents are to be sent on a diskette or by electronic means, with the S19 hexadecimal file generated by the development tool. All unused bytes must be set to FFh. The selected options are communicated to STMicroelectronics using the correctly completed option list appended *on page 223*.

Refer to application note AN1635 for information on the counter listing returned by ST after code has been transferred.

The STMicroelectronics Sales Organization will be pleased to provide detailed information on contractual points.

Figure 107. ST7FL3x Flash commercial product structure



1. For a list of available options (e.g. memory size, package) and orderable part numbers or for further information on any aspect of this device, please go to *www.st.com* or contact the ST Sales Office nearest to you.

Figure 108. ST7FL3x FASTROM commercial product structure

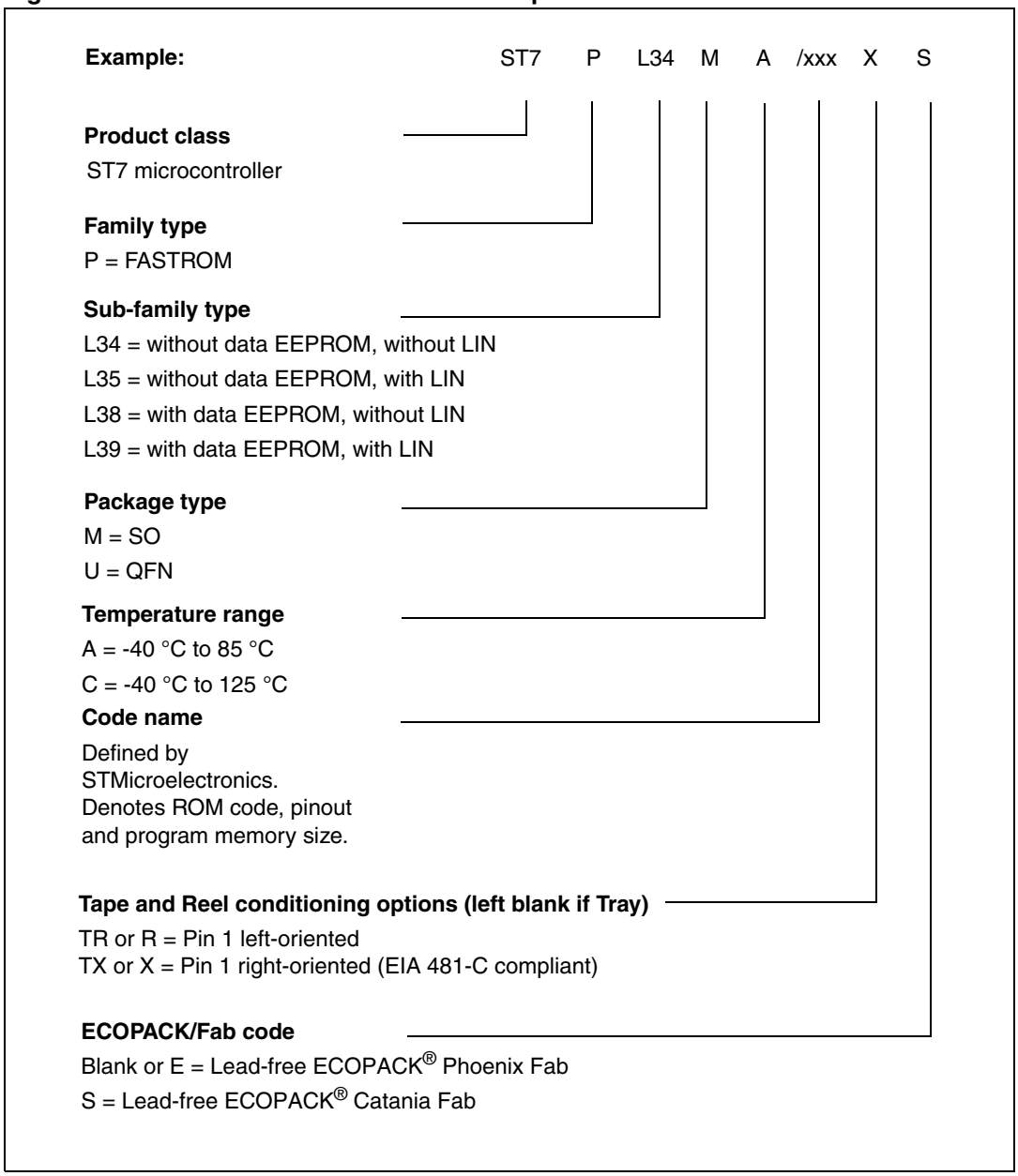
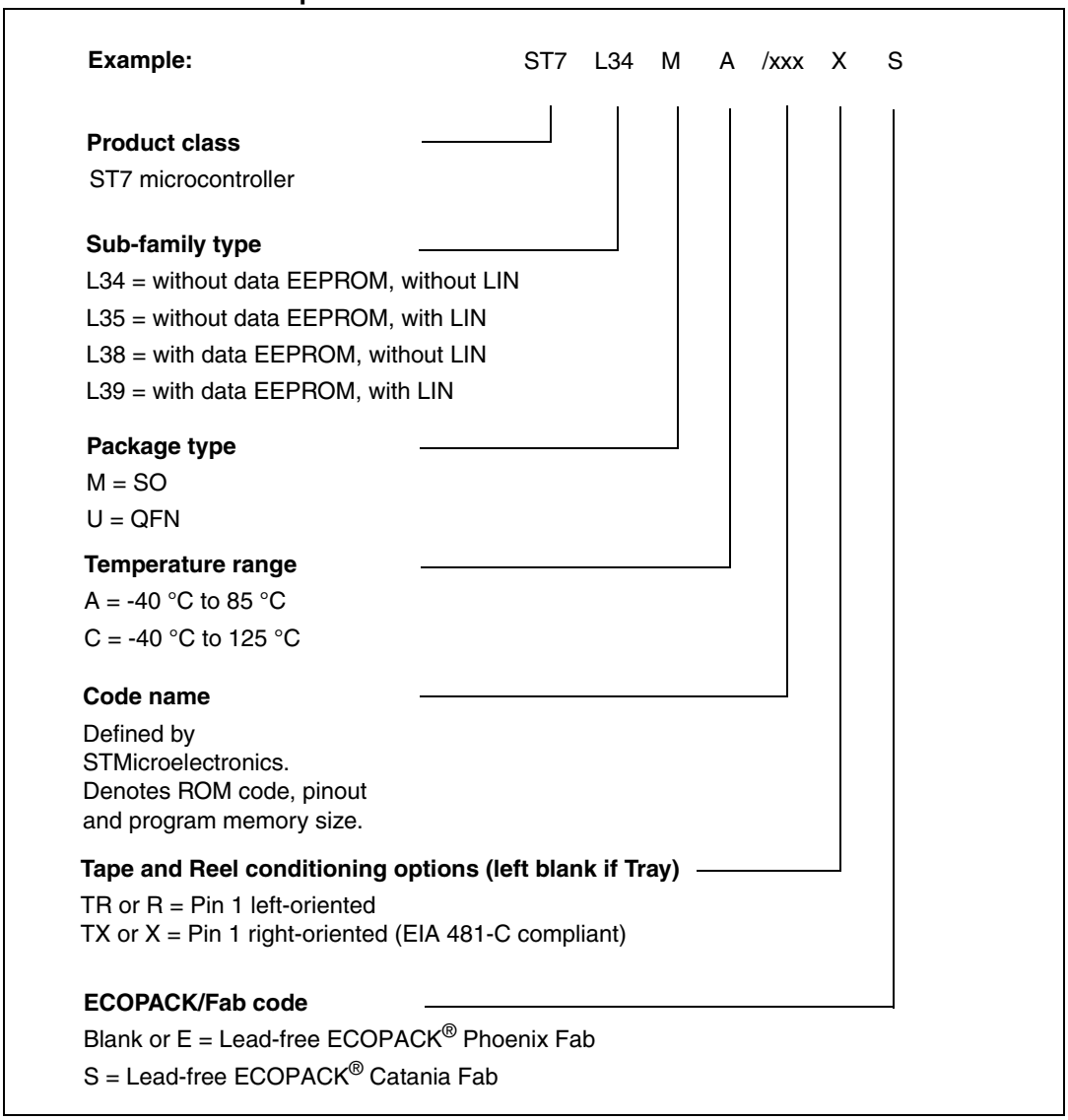


Figure 109. ROM commercial product code structure



ST7L3 FASTROM and ROM microcontroller option list

(Last update: October 2007)

Customer: .....
Address: .....
.....
Contact: .....
Phone No: .....
Reference/FASTROM or ROM code: .....

The FASTROM/ROM code name is assigned by STMicroelectronics.
FASTROM/ROM code must be sent in .S19 format. .Hex extension cannot be processed.

Device type/memory size/package (check only one option):

FASTROM device 8K	SO20	QFN20
	[ ] ST7PL34F2M	[ ] ST7PL34F2U
	[ ] ST7PL35F2M	[ ] ST7PL35F2U
	[ ] ST7PL38F2M	[ ] ST7PL38F2U
	[ ] ST7PL39F2M	[ ] ST7PL39F2U

ROM device 8K	SO20	QFN20
	[ ] ST7L34F2M	[ ] ST7L34F2U
	[ ] ST7L35F2M	[ ] ST7L35F2U
	[ ] ST7L38F2M	[ ] ST7L38F2U
	[ ] ST7L39F2M	[ ] ST7L39F2U

Conditioning: [ ] Tape and reel [ ] Tube
(check only one option)

Special marking: [ ] No [ ] Yes "....."
Authorized characters are letters, digits, '.', '-', '/' and spaces only.
Maximum character count: SO20 (8 char. max): .....
QFN20 (8 char. max): .....

Temperature range: [ ] A (-40 to +85 °C) [ ] C (-40 to +125 °C)
AWUCK selection: [ ] 32 kHz oscillator [ ] AWU RC oscillator
Clock source selection: [ ] Resonator [ ] VLP: Very low power resonator (32 to 100 kHz)
[ ] LP: Low power resonator (1 to 2 MHz)
[ ] MP: Medium power resonator (2 to 4 MHz)
[ ] MS: Medium speed resonator (4 to 8 MHz)
[ ] HS: High speed resonator (8 to 16 MHz)
[ ] External clock [ ] on PB4
[ ] on OSCI
[ ] Internal RC oscillator
PLL: [ ] Disabled [ ] Enabled
LVD reset threshold: [ ] Disabled [ ] Enabled (highest voltage threshold)
Watchdog selection: [ ] Software activation [ ] Hardware activation
Watchdog reset on halt: [ ] Disabled [ ] Enabled

Flash devices only:
Sector 0 size: [ ] 0.5 K [ ] 1 K [ ] 2 K [ ] 4 K
Readout protection: [ ] Disabled [ ] Enabled
Flash write protection: [ ] Disabled [ ] Enabled

ROM devices only
ROM readout protection: [ ] Disabled [ ] Enabled
EEDATA readout protection: [ ] Disabled [ ] Enabled

Comments: .....
Supply operating range in the application: .....
Notes: .....
Date: .....
Signature: .....

1. Not all configurations are available. See *Section 15.2: Option bytes on page 215* for authorized option byte combinations.

15.4 Development tools

15.4.1 Starter Kits

ST offers complete, affordable **starter kits**. Starter kits are complete, affordable hardware/software tool packages that include features and samples to help you quickly start developing your application.

15.4.2 Development and debugging tools

Application development for ST7 is supported by fully optimizing C compilers and the ST7 assembler-linker toolchain, which are all seamlessly integrated in the ST7 integrated development environments in order to facilitate the debugging and fine-tuning of your application. The cosmic C compiler is available in a free version that outputs up to 16 Kbytes of code.

The range of hardware tools includes full featured ST7-EMU3 series emulators, cost effective ST7-DVP3 series emulators and the low-cost RLink in-circuit debugger/programmer. These tools are supported by the ST7 Toolset from STMicroelectronics, which includes the STVD7 integrated development environment (IDE) with high-level language debugger, editor, project manager and integrated programming interface.

15.4.3 Programming tools

During the development cycle, the ST7-DVP3 and ST7-EMU3 series emulators and the RLink provide in-circuit programming capability for programming the Flash microcontroller on your application board.

ST also provides a low-cost dedicated in-circuit programmer, the ST7-STICK, as well as ST7 socket boards which provide all the sockets required for programming any of the devices in a specific ST7 subfamily on a platform that can be used with any tool with in-circuit programming capability for ST7.

For production programming of ST7 devices, ST's third-party tool partners also provide a complete range of gang and automated programming solutions, which are ready to integrate into your production environment.

15.4.4 Order codes for development and programming tools

Table 135 below lists the ordering codes for the ST7L3x development and programming tools. For additional ordering codes for spare parts and accessories, refer to the online product selector at www.st.com/mcu.

Table 135. ST7L3 development and programming tools

Supported products	In-circuit debugger, RLink series(1)		Emulator		Programming tool	
	Starter kit with demo board	Starter kit without demo board	DVP series	EMU series	In-circuit programmer	ST socket boards and EPBs
ST7FL34	ST7FLITE-SK/RAIS(2)(3)	STX-RLINK(2)(3)	ST7MDT10-DVP3(4)	ST7MDT10-EMU3	ST7-STICK STX-RLINK (4)(5)	ST7SB10-123(4)
ST7FL35						
ST7FL38						
ST7FL39						

1. *Available from ST or from Raisonance*
2. *USB connection to PC*
3. Parallel port connection to PC
4. *Add suffix /EU, /UK or /US for the power supply for your region*
5. *Includes connection kit for DIP16/SO16 only. See "How to order an EMU or DVP" in ST product and tool selection guide for connection kit ordering information*

16 Important notes

16.1 Clearing active interrupts outside interrupt routine

When an active interrupt request occurs at the same time as the related flag or interrupt mask is being cleared, the CC register may be corrupted.

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, that is, when:

- The interrupt request is cleared (flag reset or interrupt mask) within its own interrupt routine
- The interrupt request is cleared (flag reset or interrupt mask) within any interrupt routine
- The interrupt request is cleared (flag reset or interrupt mask) in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom is avoided by implementing the following sequence:

Perform SIM and RIM operation before and after resetting an active interrupt request. Example:

- SIM
- Reset flag or interrupt mask
- RIM

16.2 LINSCI limitations

16.2.1 Header time-out does not prevent wake-up from mute mode

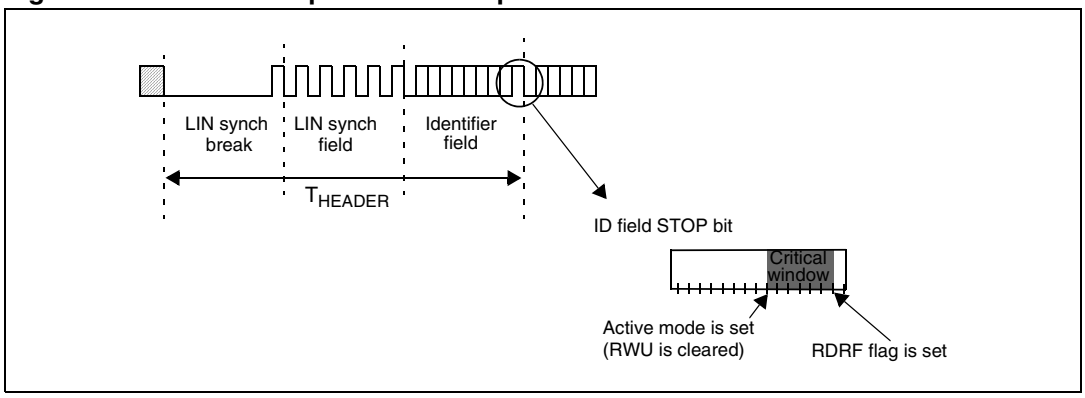
Normally, when LINSCI is configured in LIN slave mode, if a header time-out occurs during a LIN header reception (that is, header length > 57 bits), the LIN header error bit (LHE) is set, an interrupt occurs to inform the application but the LINSCI should stay in mute mode, waiting for the next header reception.

Problem description

The LINSCI sampling period is Tbit/16. If a LIN header time-out occurs between the 9th and the 15th sample of the identifier field stop bit (refer to *Figure 110*), the LINSCI wakes up from mute mode. Nevertheless, LHE is set and LIN header detection flag (LHDF) is kept cleared.

In addition, if LHE is reset by software before this 15th sample (by accessing the SCISR register and reading the SCIDR register in the LINSCI interrupt routine), the LINSCI will generate another LINSCI interrupt (due to the RDRF flag setting).

Figure 110. Header reception event sequence



Impact on application

Software may execute the interrupt routine twice after header reception.

Moreover, in reception mode, as the receiver is no longer in mute mode, an interrupt is generated on each data byte reception.

Workaround

The problem can be detected in the LINSCI interrupt routine. In case of time-out error (LHE is set and LHLR is loaded with 00h), the software can check the RWU bit in the SCICR2 register. If RWU is cleared, it can be set by software (refer to *Figure 111*). The workaround is shown in bold characters.

Figure 111. LINSCI interrupt routine

```
@interrupt void LINSCI_IT ( void ) /* LINSCI interrupt routine */
{
      /* clear flags */
      SCISR_buffer = SCISR;
      SCIDR_buffer = SCIDR;

      if ( SCISR_buffer & LHE )/* header error ? */
      {
            if (!LHLR)/* header time-out? */
            {
                  if ( !(SCICR2 & RWU) )/* active mode ? */
                  {
                        _asm("sim");/* disable interrupts */
                        SCISR;
                        SCIDR;/* Clear RDRF flag */
                        SCICR2 |= RWU;/* set mute mode */
                        SCISR;
                        SCIDR;/* Clear RDRF flag */
                        SCICR2 |= RWU;/* set mute mode */
                        _asm("rim");/* enable interrupts */
                  }
            }
      }
}
```

Example using Cosmic compiler syntax

17 Revision history

Table 136. Revision history

Date	Revision	Changes
Jun-2004	1	First release
23-Dec-2005	2	Changed temperature range (added -40 to +125°C) Removed references to 1% internal RC accuracy; Changed *Figure 4: Memory map on page 19* Removed reference to amplifier for ADCDRL in *Table 3: Hardware register map on page 20* and in *Section 11.6.6: Register description on page 166* and replaced “Data register low” by “Control and data register low”; Changed *Section 4.4: ICC interface on page 23* and added note 6 Modified note on clock stability and on ICC mode in *Section 7.1: Internal RC oscillator adjustment on page 37* Added text in note 1 in *Table 7: RCCR calibration registers on page 37* Added RCCR1 (*Figure 4 on page 19* and *Section 7: Supply, reset and clock management on page 37*) Added note to *Section 7.5: Reset sequence manager (RSM) on page 42*; Added note 3 after *Table 24: I/O port mode options on page 70* Exit from halt mode during an overflow event set to ‘no’ in *Section 11.2.4: Low power modes on page 88* Removed watchdog section in *Section 11.3: Lite timer 2 (LT2) on page 101* *Table 48: Effect of low power modes on lite timer 2 on page 104* and *Table 49: Lite timer 2 interrupt control/wake-up capability on page 104* expanded Added important note in *Master mode operation on page 111* Changed procedure description in *Transmitter on page 126* In *Extended baud rate generation on page 130*: Corrected equation for Rx to read: Rx = f_{CPU}/(16 x ERPR x PR x RR), {instead of Rx = f_{CPU}/(16 x ERPR x PR x TR)} Added note on illegal opcode reset to *Section 12.2.2: Illegal opcode reset on page 175*; Changed *Section 13.1.2: Typical values on page 178* Changed electrical characteristics in the following sections: *Section 13.3: Operating conditions on page 182*, *Section 13.4: Supply current characteristics on page 189*, *Section 13.6: Memory characteristics on page 195*, *Section 13.7.3: Absolute maximum ratings (electrical sensitivity) on page 198*, *Section 13.8: I/O port pin characteristics on page 199*, *Section 13.9: Control pin characteristics on page 205*, *Section 13.10: Communication interface characteristics on page 207* and *Section 13.11: 10-bit ADC characteristics on page 210* Modified *Section 14: Package characteristics on page 212* Changed *Section 15.2: Option bytes on page 215* (OPT 5 of option byte 1), *Section 15.3: Device ordering information and transfer of customer code on page 219* and *Section 15.4: Development tools on page 224* Changed option list, Added *Section 16: Important notes on page 226*

Table 136. Revision history (continued)

Date	Revision	Changes
06-Mar-2006	3	Removed all x4 PLL option references from document Changed Read operation section in *Section 5.3: Memory access on page 28* Changed note *Figure 8: Data EEPROM write operation on page 29*. Changed *Section 5.5: Access error handling on page 30* Replaced 3.3 V with 3.6 V in *Section 7.2: Phase locked loop on page 38* Changed *Master mode operation on page 111*: added important note Changed *Section 13.1.2: Typical values on page 178* Changed *Section 13.3.1: General operating conditions on page 182* and added note on clock stability and frequency accuracy; removed the following figure: PLL $\Delta f_{CPU}/f_{CPU}$ vs time Changed *Section 13.3.2: Operating conditions with low voltage detector (LVD) on page 186* and *Section 13.3.4: Internal RC oscillator and PLL on page 188* Changed *Section 13.4.1: Supply current on page 189* and added notes Changed *Table 115: Characteristics of EEPROM data memory on page 196* Removed note 6 from *Section 13.6: Memory characteristics on page 195* Changed *Section 13.6.2: Flash program memory on page 195*; Changed *Section 13.6.3: EEPROM data memory on page 196* Changed values in *Section 13.7.2: Electromagnetic interference (EMI) on page 197* Changed absolute maximum ratings in *Table 118: ESD absolute maximum ratings on page 198* Changed *Section 13.8.1: General characteristics on page 199* and *Section 13.8.2: Output driving current on page 200* Changed *Section 13.9.1: Asynchronous RESET pin on page 205* (changed values, removed references to 3 V and added note 5) Changed *Section 13.11: 10-bit ADC characteristics on page 210*: changed values in ADC accuracy tables and added note 3 Changed notes in *Section 14.3: Thermal characteristics on page 214* Changed *Section 15: Device configuration and ordering information on page 215* Changed *Table 130: Soldering compatibility (wave and reflow soldering process) on page 208* Added note to OSC option bit in *Section 15.2: Option bytes on page 215* Changed configuration of bit 7, option byte 1, in *Table 130: Flash and ROM option bytes on page 215* Changed device type/memory size/package and PLL options in *ST7L1 FASTROM and ROM microcontroller option list on page 257.*

Table 136. Revision history (continued)

Date	Revision	Changes
17-Mar-2006	4	Changed caution text in *Section 8.2: External interrupts on page 51* Changed external interrupt function in *Section 10.2.1: Input modes on page 68* Changed *Table 101* and *Table 102* Changed *Table 103* and *Table 104* Changed *Figure 98: RESET pin protection when LVD Is enabled on page 206* Removed EMC protective circuitry in *Figure 97: RESET pin protection when LVD is disabled on page 206* (device works correctly without these components) Removed section 'LINSCI wrong break duration' from *Section 16: Important notes on page 226*
20-Dec-2006	5	Replaced 'ST7L3' with 'ST7L34, ST7L35, ST7L38, ST7L39' in document name and added QFN20 package to package outline on cover page. Changed *Section 1: Description on page 14* Transferred device summary table from cover page to *Section 1: Description on page 14* Added QFN20 package to the device summary table in *Section 1: Description on page 14* *Figure 1: General block diagram on page 15*: Replaced autoreload timer 2 with autoreload timer 3 Added *Figure 3: 20-pin QFN package pinout on page 16* to *Section 2* *Table 2: Device pin description on page 17*: - Added QFN20 package pin numbers - Removed caution about PB0 and PB1 negative current injection restriction *Figure 4: Memory map on page 19*: Removed references to note 2 *Table 3: Hardware register map on page 20*: - Changed register name for LTCNTR - Changed reset status of registers LTCSR1, ATCSR and SICSR - Changed note 3 Changed last paragraph of *Section 5.5: Access error handling on page 30* Added caution about avoiding unwanted behavior during reset sequence in *Section 7.5.1: Introduction on page 42* *Figure 17: Reset sequences on page 45*: Replaced 'T_{CPU}' with 't_{CPU}' at bottom of figure Changed notes in *Section 7.6.1: Low voltage detector (LVD) on page 45* *Figure 19: Reset and supply management block diagram on page 47*: Removed names from SICSR bits 7:5 Changed reset value of bits CR0 and CR1 from 0 to 1 in *Section 7.6.4: Register description on page 48* *Table 16: Interrupt sensitivity bits on page 54*: Restored table number (inadvertantly removed in Rev. 3) *Figure 34: Watchdog block diagram on page 75*: Changed register label Changed register name and label in *Section 11.1.6: Register description on page 77* Added note for ROM devices only to *PWM mode on page 79*

Table 136. Revision history (continued)

Date	Revision	Changes
20-Dec-2006	5 cont'd	Replaced bit name OVIE1 with OVFIE1 in *Table 35: AT3 interrupt control/wake-up capability on page 89* Changed description of bits 11:0 in *Counter register 1 high (CNTR1H) on page 90*, *Counter register 1 low (CNTR1L) on page 90* and *Table 37: CNTR1H and CNTR1L register descriptions on page 91* Changed name of register ATR1H and ATR1L in *Autoreload register high (ATR1H)*, *Autoreload register low (ATR1L)* and *Table 38: ATR1H and ATR1L register descriptions on page 92* Changed name of register ATCSR2 in *Timer control register2 (ATCSR2)* and *Table 44: ATCSR2 register description on page 97* Changed name of register ATR2H and ATR2L in *Autoreload register2 high (ATR2H)*, *Autoreload register2 low (ATR2L)* and *Table 45: ATR2H and ATR2L register descriptions on page 98* Changed name of register LTCSR1 in *Lite timer control/status register (LTCSR1)* and *Table 53: LTCSR1 register description on page 106*. Changed names of registers SPIDR, SPICR and SPICSR in *Section 11.4.8: Register description on page 118* *Figure 64: LIN synch field measurement on page 148*: - replaced 't_{CPU}' with 'T_{CPU}' - replaced 't_{BR}' with 'T_{BR}' Modified *Table 98: Current characteristics on page 180*: - Changed I_{IO} values - Removed 'Injected current on PB0 and PB1 pins' from table - Removed note 5 'no negative current injection allowed on PB0 and PB1 pins' Restored symbol for PLL jitter in *Table 102: Operating conditions (tested for TA = -40 to +125 °C) @ VDD = 4.5 to 5.5 V on page 183* (inadvertantly changed in Rev. 4) Added note 5 to *Table 105: Operating conditions with low voltage detector on page 186* Specified applicable T_A in *Table 113: RAM and hardware registers on page 195*, *Table 114: Characteristics of dual voltage HDFlash memory on page 195* and *Table 115: Characteristics of EEPROM data memory on page 196* Changed T_A for programming time for 1~32 bytes and changed T_{PROG} from 125°C to 85°C for write erase cycles in *Table 114: Characteristics of dual voltage HDFlash memory on page 195* *Figure 84: Two typical applications with unused I/O pin on page 199*: Replaced ST7XXX with ST7 *Table 121: Output driving current on page 200*: Added table number and title *Table 122: Asynchronous RESET pin on page 205*: Added table number and title Replaced ST72XXX with ST7 in *Figure 98: RESET pin protection when LVD Is enabled on page 206* and *Figure 97: RESET pin protection when LVD is disabled on page 206* Changed *Section 13.10.1: Serial peripheral interface (SPI) on page 207* *Figure 100: SPI slave timing diagram with CPHA = 1 on page 208*: Replaced CPHA = 0 with CPHA = 1 *Figure 101: SPI master timing diagram on page 209*: Repositioned $t_{v(MO)}$ and $t_{h(MO)}$

Table 136. Revision history (continued)

Date	Revision	Changes
20-Dec-2006	5 cont'd	*Table 124: 10-bit ADC characteristics on page 210*: Added table number and title Changed P_{Dmax} value for SO20 package in *Table 129: Thermal characteristics on page 214* Removed text concerning LQFP, TQFP and SDIP packages from *Section 15: Device configuration and ordering information on page 215*. *Figure 102: Typical application with ADC on page 210*: Replaced ST72XXX with ST7 Changed typical and maximum values and added table number and title to *Table 125: ADC accuracy with 4.5 V < VDD < 5.5 V on page 210* and to *Table 126: ADC accuracy with 3 V < VDD < 3.6 V on page 211* Added *Figure 105: QFN 5x6, 20-terminal very thin fine pitch quad flat no-lead package on page 213* Added QFN20 package to *Table 129: Thermal characteristics on page 214* *Table 130: Soldering compatibility (wave and reflow soldering process) on page 208*: - Changed title of 'Plating material' column - Added QFN package - Removed note concerning Pb-package temperature for leadfree soldering compatibility Changed *Section 15.2: Option bytes on page 215* to add different configurations between Flash and ROM devices for OPTION BYTE 0 Removed 'automotive' from title of *Section 15.3: Device ordering information and transfer of customer code on page 219* Removed *'Supported part numbers'* table from *Section 15.3: Device ordering information and transfer of customer code on page 219* Added *Figure 107: ST7FL3x Flash commercial product structure on page 220* Added *Table 135: Flash user programmable device types on page 213* Added *Figure 108: ST7FL3x FASTROM commercial product structure on page 221* Added *Table 136: FASTROM factory coded device types on page 215* Added *Figure 109: ROM commercial product code structure on page 222* Added *Table 137: ROM factory coded device types on page 216* Updated *ST7L3 FASTROM and ROM microcontroller option list on page 223* Changed *Section 15.4: Development tools on page 224* and *Table 135: ST7L3 development and programming tools on page 225* Updated disclaimer (last page) to include a mention about the use of STproducts in automotive applications
02-Aug-2010	6	Changed the description of internal RC oscillator from 'high precision' to '1% in Clock, reset and supply management *on cover page* and in *Section 7.4.3: Internal RC oscillator on page 41* *Table 7: RCCR calibration registers on page 37*: Added footnote *2* *Section 7.6.1: Low voltage detector (LVD) on page 45*: Changed the sentence about the LVD to read that it can be enabled with 'highest voltage threshold' instead of 'low, medium or high' *Figure 26: Halt mode flowchart on page 60*: Added footnote *5*

Table 136. Revision history (continued)

Date	Revision	Changes
02-Aug-2010	6 cont'd	*Figure 31: AWUFH mode flowchart on page 65*: Added footnote *5* *Table 101: Operating conditions (tested for TA = -40 to +125 °C) @ VDD = 4.5 to 5.5 V on page 183*: Changed f_{RC} and ACC_{RC} Flash values *Table 102: Operating conditions (tested for TA = -40 to +125 °C) @ VDD = 4.5 to 5.5 V on page 183*: Changed ACC_{PLL} 'typical' value *Table 103: Operating conditions (tested for TA = -40 to +125 °C) @ VDD = 3.0 to 3.6 V on page 184*: Changed f_{RC} and ACC_{RC} Flash values *Table 118: ESD absolute maximum ratings on page 198*: Changed values of $V_{ESD(HBM)}$ and $V_{ESD(HBM)}$. *Table 132: Option byte 1 description on page 217*: Added 'must be set to 1' to option bit 7 and 'must be set to 0' to option bit 5 *Table 133: Option byte 0 description on page 217*: Added 'must be set to 1' to option bits 3:2 Updated *Figure 107: ST7FL3x Flash commercial product structure*, *Figure 108: ST7FL3x FASTROM commercial product structure* and *Figure 109: ROM commercial product code structure*
11-Oct-2010	7	Updated fCLKIN test conditions and maximum values in *Table 100: General operating conditions* and *Figure 72: fCLKIN maximum operating frequency vs VDD supply voltage*
18-Nov-2011	8	Updated the maximum value of external clock frequency on CLKIN pin (f_{CLKIN}) when V_{DD} = 3 to 3.3 V in *Table 100: General operating conditions on page 182* and in *Figure 72: fCLKIN maximum operating frequency vs VDD supply voltage on page 182*.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY TWO AUTHORIZED ST REPRESENTATIVES, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com